John S. Gero
Sean Hanna *Editors*

Design Computing and Cognition '14

Springer

Design Computing and Cognition ′14

John S. Gero • Sean Hanna
Editors

Design Computing and Cognition ’14

Springer

Editors
John S. Gero
Department of Computer Science
and School of Architecture
University of North Carolina
Charlotte, NC, USA

Sean Hanna
Bartlett School of Architecture
University College London
London, UK

ISBN 978-3-319-14955-4 ISBN 978-3-319-14956-1 (eBook)
DOI 10.1007/978-3-319-14956-1

Library of Congress Control Number: 2015932650

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Preface

Research develops new knowledge, but design has to act. In science, as in the humanities, we have the luxury of holding our theories as tentative, admitting they will eventually be superseded by new evidence or argument. But design decisions must always be made even though our knowledge of the situation is always incomplete, as design problems are by nature ill-defined, unique, and 'wicked'. Since it also deals in propositions rather than explanations, design itself is at risk of being misunderstood in the context of traditional academic research; yet it is the crucial activity that puts into practice the research and knowledge gained across the arts, humanities, sciences and social sciences. This deep cross-disciplinarity is what makes a conference like *Design Computing and Cognition* so valuable, as it brings together those who study human cognition and those who model it with a machine, those who try to understand what designers do and those who help them to do it better.

The activity of design has always been with us. It features in the earliest surviving literature we have: the *Epic of Gilgamesh*, for example, features 4,000 year old design commentary on cities and a flood-proof ark, and the *Instructions of Shuruppak*, perhaps the oldest text in the world, begins with directions for urban and rural planning. What is being designed is changing drastically. The kinds of objects, systems and environments that are now created by design, from cities to global manufacturing networks and virtual environments, are of unprecedented scale. We are compelled to deal with rapid changes in technologies, and coordinate work across continents and time zones. These are scales at which our normal intuition and experience do not function. As the scale and speed of production of such designed outputs increase, so do their impacts, and along with it the importance of making informed decisions in design.

It is this change that makes research in design both so urgent and so interesting. At the same time that we are required to design entire cities, devices and services virtually overnight, the technology is also emerging to handle the 'big data' through which we might achieve an understanding of the nexus between design and consumer. As designers become more enabled by—and embedded in—computational tools, neuroscience and cognitive science are seeing deeper into their brains. For the past decade, this conference series has provided a bridge between the fields of design computing and design cognition. The confluence of these two fields continues to provide the foundation for further advances in both and to an increased understanding design as an activity whose influence continues to spread.

The papers in this volume are from the *Sixth International Conference on Design Computing and Cognition (DCC'14)* held at University College London, UK. They represent the state-of-the-art of research and development in design computing and design cognition. They are of particular interest to researchers, developers and users of advanced computation in design and those who need to gain a better understanding of designing.

In these proceedings the papers are grouped under the following nine headings, describing both advances in theory and application and demonstrating the depth and breadth of design computing and design cognition:

Design Synthesis
Design Cognition
Design Creativity
Design Processes 1
Design Theory
Design Grammars
Design Support
Design Processes 2
Design Ideation

A total of 131 full papers were submitted to the conference, from which 38 were accepted and 37 appear in these proceedings. Each paper was extensively reviewed by at least three reviewers drawn from the international panel listed on the following pages. The reviewers' recommendations were then assessed before the final decision on each paper was taken, and the authors improved their contributions based on the advice of this community. Thanks go to them, for the quality of these papers depends on their efforts. Thanks also go to Pinelopi Kyriazi for putting the papers together into a single volume.

London, UK — Sean Hanna
Charlotte, NC, USA — John S. Gero

List of Reviewers

Henri Achten, Czech Technical University, Czech Republic
Robin Adams, Purdue University, USA
Saeema Ahmed, Technical University of Denmark, Denmark
Kinda Al Sayed, University College London, UK
Katerina Alexiou, Open University, UK
Janet Allen, University of Oklahoma, USA
Eiji Arai, Osaka University, Japan
Petra Badke-Schaub, Delft University of Technology, The Netherlands
Stefania Bandini, University of Milano-Bicocca, Italy
Can Baykan, Middle East Technical University, Turkey
Peter Bentley, University College London, UK
Eric Blanco, G-SCOP-Grenoble INP
Frances Brazier, TU Delft, Netherlands
David C. Brown, Worcester Polytechnic Institute, USA
Ken Brown, University College Cork, Ireland
Duncan Brumby, University College London, UK
Jonathan Cagan, Carnegie Mellon University, USA
Hernan Casakin, Ariel University, Israel
Gaetano Cascini, Politecnico di Milano, Italy
Gabriela Celani, UNICAMP, Brazil
Amaresh Chakrabarti, Indian Institute of Science, India
Wei Chen, Northwestern University, USA
Per Christiansson, Aalborg University, Denmark
P. John Clarkson, University of Cambridge, UK
Mark Clayton, Texas A&M University, USA
Graham Coates, Durham University, UK
Nathan Crilly, University of Cambridge, UK
Andrew Crooks, George Mason University, USA
Steve Culley, University of Bath, UK
Françoise Darses, LIMSI-CNRS, France

Andy Dong, University of Sydney, Australia
Chris Earl, Open University, UK
Claudia Eckert, Open University, UK
Athanassios Economou, Georgia Institute of Technology, USA
Ozgur Eris, TU Delft, Netherlands
Benoit Eynard, Université de Technologie de Compiègne, France
Georges Fadel, Clemson University, USA
Alexander Felfernig, Graz University of Technology, Austria
Susan Finger, Carnegie Mellon University, USA
Daniel Frey, Massachusetts Institute of Technology, USA
Renate Fruchter, Stanford University, USA
Haruyuki Fujii, Tokyo Institute of Technology, Japan
John Gero, University of North Carolina at Charlotte, USA
Pablo Gervás, Universidad Complutense de Madrid, Spain
Ashok Goel, Georgia Institute of Technology, USA
Gabriela Goldschmidt, Technion – Israel Institute of Technology, Israel
Ewa Grabska, Jagiellonian University, Poland
Kazjon Grace, University of North Carolina at Charlotte, USA
Sean Hanna, Bartlett School of Architecture, University College London, London, UK
Armand Hatchuel, Ecole Nationale Supérieure des Mines de Paris, France
Ann Heylighen, KU Leuven, Belgium
Abi Hird, University of Strathclyde, UK
Yuemin Hou, Tsinghua University, China
Thomas J. Howard, The Technical University of Denmark, Denmark
Hao Jiang, Zhejiang University, China
Yan Jin, University of Southern California, USA
Iestyn Jowers, Open University, UK
Jeff Kan, City University of Hong Kong, China
Udo Kannengiesser, Metasonic, Germany
Nick Kelly, University of Southern Queensland, Australia
Terry Knight, Massachusetts Institute of Technology, USA
Corinna Koenigseder, ETH Zurich, Switzerland
Gül Kremer, Pennsylvania State University, USA
Ramesh Krishnamurti, Carnegie Mellon University, USA
Ehud Kroll, Technion – Israel Institute of Technology, Israel
Djordje Krstic, Hetzel Design Inc.
Bimal Kumar, Glasgow Caledonian University, Scotland
Pascal Le Masson, MINES ParisTech, France
Noel Leon, ITESM, Mexico
Julie Linsey Georgia Institute of Technology, USA
Mary Lou Maher, University of North Carolina at Charlotte, USA
Peter Matthews, Durham University, UK
Janet McDonnell, Central Saint Martins, University of the Arts London, UK

Chris McManus, University College London, UK
Scarlett Miller, Pennsylvania State University, USA
Elena Mulet, Universitat Jaume I, Spain
Clemens Muenzer, ETH Zurich, Switzerland
Jeffrey V. Nickerson, Stevens Institute of Technology, USA
Morteza Pourmohamadi, University of Sydney, Australia
Rabee Reffat, King Fahd University of Petroleum and Minerals, Saudi Arabia
Yoram Reich, Tel-Aviv University, Israel
Luis Romão, Technical University of Lisbon, Portugal
Duska Rosenberg, Royal Holloway University of London, UK
Pertti Saariluoma, University of Jyväskylä, Finland
Somwrita Sarkar, University of Sydney, Sydney, Australia
Jonathan Sauder, University of Southern California, USA
Gerhard Schmidt, ETH Zurich, Switzerland
Jami Shah, Arizona State University, USA
Offer Shai, Tel-Aviv University, Israel
Kristi Shea, ETH Zurich, Switzerland
Li Shu, University of Toronto, Canada
Vishal Singh, Aalto University, Finland
Tim Smithers, TS Research Consulting
Ricardo Sosa, Singapore University of Technology and Design
Jayesh Srivastava, University of Toronto, Canada
Martin Stacey, De Montfort University, UK
Tino Stankovic, ETH Zurich, Switzerland
George Stiny, Massachusetts Institute of Technology, USA
Fritz Stoeckli, ETH Zurich, Swtizerland
Rudi Stoufffs, Delft University of Technology, Netherlands
Joshua Summers, Clemson University, USA
Hsien-Hui Tang, National Taiwan University of Science and Technology, Taiwan
Ming Xi Tang, The Hong Kong Polytechnic University, China
Toshiharu Taura, Kobe University, Japan
Barbara Tversky, Columbia and Stanford, USA
Andrew Vande Moere, K.U. Leuven, Belgium
Noe Vargas Hernandez, University of Texas at El Paso, USA
Pieter Vermaas, TU Delft, Netherlands
Tijana Vuletic, University of Strahclyde, UK
Robert Woodbury, Simon Fraser University, Canada
Meng Xu, Clemson University, USA
Maria Yang, Massachusetts Institute of Technology, USA
Bernard Yannou, Ecole Centrale Paris, France
Seda Yilmaz, Iowa State University, USA
Robert Youmans, George Mason University, USA
Theodore Zamenopoulos, Open University, UK

Contents

Part I
Design Synthesis

Dynamic Structuring in Cellular Self-Organizing Systems

Newsha Khani and Yan Jin

Abstract Conventional mechanical systems composed of various modules and parts are often inherently inadequate for dealing with unforeseeable changing situations. Taking advantage of the flexibility of multi-agent systems, a cellular self-organizing (CSO) systems approach has been proposed, in which mechanical cells or agents self-organize themselves as the environment and tasks change based on a set of rules. To enable CSO systems to deal with more realistic tasks, a two-field mechanism is introduced to describe task and agents complexities and to investigate how social rules among agents can influence CSO system performance with increasing task complexity. The simulation results of case studies based on the proposed mechanism provide insights into task-driven dynamic structures and their effect on the behavior, and consequently the function, of CSO systems.

Introduction

Adaptability is needed for systems to operate in harsh and unpredictable environments where it is impossible for the designer to conceptualize every possible incident and predict details of changing functional requirements. Space and deep sea explorations and rescue missions in hazardous environments are some examples of such variable environments. In most, if not all, engineered systems, the physical components are designed for a limited purpose and restricted operation range, beyond which the behaviors are not predictable.

The existing approach to dealing with changing task environments relies on designers' imagination of a variety of possible situations of the task domain that helps them devise needed responses to the imaginable possibilities. Following the law of requisite variety [1]—i.e., only variety (of the system) can conquer variety (of the task)—this approach increases the system variety by adding more components and therefore enlarging system state space. While the approach has been effective for many complex systems developed to date, as the system components

N. Khani (✉) • Y. Jin
University of Southern California, USA
e-mail: nkhani@usc.edu

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_1

become too many, highly sophisticated and their interactions more intertwined, unintended interactions will ensue, making it difficult for designers to ensure the valid operation range for the system to survive its expected lifecycle.

As an alternative approach to adaptive and complex engineered systems, a cellular self-organizing (CSO) systems approach has been proposed [2–4]. Like many other multi-agent systems, A CSO system is composed of multiple homogeneous or heterogeneous mechanical cells (mCells, i.e., agents) that can be a small functional component or a robot. Each mCell is equipped with needed sensors and actuators and encoded with system design-DNA (dDNA) containing the information that specifies cellular actions and decisions for taking these actions. mCells interact with their task environment and with each other, leading to self-organizing emergent behavior and functions at the system level. To facilitate mCells' interactions with the task environment, a task field-based regulation (FBR) mechanism has been developed [4]. To explore mCell interactions, a COARM (collision, avoidance, alignment, randomness, and momentum) parametric model has been examined [3].

The self-organizing behavior of the current CSO systems is regulated by each mCell transforming the task environment into a task-field in which it finds its "most comfortable place" and moves into it. The task is completed by the collective effort of the mCells making themselves "more comfortable." Each mCell makes their movement decisions completely based on its own sensed information of environment, its own transformation algorithm, and its own decisions for action. mCells collectively perform the task by first "discovering" what the task is (where is the "comfortable place") and then "carrying out" the task (move into the "comfortable place"). This distributed and self-interested approach allows for flexibility to cope with changing tasks, robustness to deal with changing environment, and resilience to still function with system dismemberment.

The current field-based regulation (FBR) approach to self-organization has two problems. First, when the task becomes more complex, both the description and transformation of task-field become highly complicated, potentially becoming a design hurdle. Second, the current approach does not directly address the interaction between mCells with respect to the task, leaving the power of mCells' self-organized structures unutilized. As will be described in the following sections, the problems have become evident when we make the box-moving task to include "rotate" the box in addition to simply "push/move" the box. This increased complexity of the task has made the simple FBR based CSO system incapable of completing the task in most cases, even when the field description is fully supplied. There is a need to incorporate task-driven dynamic structuring among mCells into the self-organizing framework.

Social structures play an important role in solving collective tasks. Many complex systems are hierarchical in structure including social systems, biological systems and physical systems [5]. Structures can be found everywhere in society, such as governments, companies, and universities. Many natural systems, over eons of time, have participated in the evolution process to organize themselves into a more complex and favorable arrangement [6].

In this research, we explore a dynamic social structuring approach to enhance the self-organizing functionality for CSO systems. We attain social structuring among mCells by introducing both general and context-based social rules and devise a social-rule based regulation (SRBR) for mCells to choose their actions. To facilitate SRBR, we introduce the concept of "social field" in addition to the current "task field." In SRBR, mCells' behavior is adjusted through perceived social field to be in harmony with system-wide welfare. Social rules can be designed based on the task definition and resolution of possible occurring conflicts.

In the rest of this paper, we first review the related work in section "Related Work", and then, in section "A Social Rule Based Regulation Approach to Dynamic Social Structuring", introduce our dynamic social structuring concepts and present social rule based behavior regulation (SRBR) approach. In section "Case Study" we demonstrate the effectiveness of our approach through simulation-based case studies. Section "Concluding Remarks" draws conclusions and points to future research directions. In the following, we will use the word "agent" and "mCell" interchangeably and will use the latter only when necessary for emphasizing CSO features.

Related Work

In the field of engineering design, design for adaptability and design of reconfigurable systems have been investigated in the past decade. In their work focusing on vehicle design, Ferguson and Lewis [7] introduced a method of designing effective reconfigurable systems that focuses on determining how the design variables of a system change, as well as investigating the stability of a reconfigurable system through the application of a state-feedback controller. Martin and Ishii [8] proposed a design for variety (DFV) approach that allows quick reconfiguration of products but mainly aims to reduce time to market by addressing generational product variation. Indices have been developed for generational variance to help designers reduce the development time of future evolutionary products. In addition to developing design methods for reconfigurable systems, various reconfigurable robotics have been developed mostly by computer scientists. Unsal et al. [9] focused on creating very simplistic i-Cube systems (with cubes being able to attached to each other) in order to investigate whether they can fully realize the full potential of this class of systems. PolyBot has gone through several updates over the years [10] but acquired notoriety by being the first robot that "demonstrated sequentially two topologically distinct locomotion modes by self-configuration. SuperBot [11] is composed of a series of homogeneous modules each of which has three joints and three points of connection. Control of SuperBot is naturally inspired and achieved through a "hormone" control algorithm.

Despite the implicit and informal nature of some multi-agent relations, all multi-agent systems possess some form of organization. For a distributed system with the purpose of solving a problem or reaching objective functionality, an organized way of sharing information among agents can be very helpful. Organizational oriented design has shown to be effective and is typically used to achieve better communication strategies [12]. It has been proved that the behavior of the system depends on shape, size and characteristics of the organizational structure [13, 14]. Researchers have suggested that there is no single type of organization that is a best match for all circumstances [13].

As an alternative approach to adaptive and complex engineered systems, the previous work on cellular self-organizing systems (CSO) has provided useful insights into understanding necessary characteristics of adaptive systems and introducing nature inspired concepts. The current FBR approach is fully distributed since every mCell works on their own without considering other mCells. From a multi-agent system's perspective, the full distribution represents a level of disorderliness that has two important implications. First, the disorderliness means limited functional capabilities because the system lacks ways to create corresponding sophistication when tasks become more complex. Second, the disorderliness, on the other hand, provides an opportunity for us to infuse order into the system and therefore increase the level of overall system capability. The question is *how can we devise such order so that we can "control" the level of orderliness for best balance of system adaptability and functionality*?

A Social Rule Based Regulation Approach to Dynamic Social Structuring

Basic Idea

As mentioned above, a system needs to possess a certain level of complexity in order to deal with tasks with a corresponding level of complexity [1]. Furthermore, it has been demonstrated that a system with higher physical complexity is more adaptable because the higher-level diversity permits satisfaction of changes of constraints around the system [15]. Although algorithmic information content based complexity measure equals randomness with complexity, from a system design perspective, it is more appropriate to count the complexity of a system based on its physical, structural, and effective features. In this case, pure randomness is discounted and the attention is placed on agent interactions and evolving structures.

Following Huberman and Hogg [15], we consider the complexity spectrum of engineered systems over order and disorder bell shaped, as illustrated in Fig. 1. A single solid object, such as a hammer, has complete order, as indicated in point (a) in Fig. 1; it has close to zero complexity and can deal with very simple tasks, such as punching a nail. By increasing number of dedicated components and introducing interactions between them, the order decreases in the sense that the system can be in various ranges of possible states. Such systems can be a gearbox (simpler) or an internal combustion engine (more complex). Although this "complexity by design" approach (from (a) to (c) in Fig. 1) has been the mainstream approach to complex engineered systems and has been highly effective, the

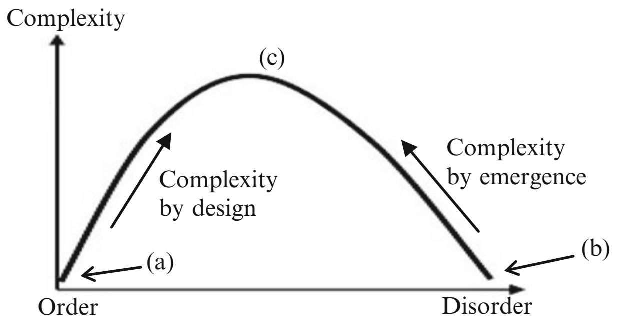


Fig. 1 Hypothetical system complexity over order-disorder spectrum (Adapted from [15])

unintended and unknown interactions among the sophisticated components may potentially become a "showstopper" when the systems demand super complexity for super demanding tasks. Space mission accidents and those of nuclear power plants are examples.

An alternative approach to complex engineered systems is to start from completely disorganized simple agents (or mCells in our CSO term), as indicated by point (b) in Fig. 1. While the completely disordered agents cannot perform any task, not even punching nails, introducing order among the agents can potentially lead to a functional system (moving from (b) to (c) in Fig. 1). Physical materials, biological systems, and ant colonies are examples. The distinctive feature of this approach to complex engineered system is "complexity by emergence." Since "by emergence" does not require explicit knowledge of specific interactions among agents, the "showstopper" mentioned above can be avoided. Furthermore, this approach may fundamentally expand the design of engineered systems by bringing biological developmental concepts into mechanical system development.

Our research on self-organizing systems takes the "by emergence" approach. Besides introducing the concepts of design-DNA (dDNA) and mechanical cell (mCell, i.e., agent), a task field based behavior regulation (FBR) mechanism has been developed to allow agents to self-organize (i.e., introducing order) through each agent seeking attractors of its perceived task field. Based on Fig. 1, previous CSO system designs fall into a cluster of points close to (b). Although this limited orderliness was effective for completing "push box" tasks, it was not enough for "push and rotate box." To further increase the level of orderliness, in this research we introduce the concept of "social structure" to capture explicit interactions among agents and apply "social rules" to facilitate dynamical social structuring among agents.

In the following subsections, we first introduce the measure of task complexity and then describe the models of agents, social structures, and social rules, followed by the social rule based regulation mechanism. The subsequent case study sections will demonstrate how higher level task complexity demands dynamic structuring and how social rule-based regulation can be applied to increase the orderliness, and consequently the capability, of the overall system.

Task Complexity

In the CSO framework, tasks together with environmental situations are presented as “task fields” in which agents seek and move to attractors [4]. Task field is defined as below.

Definition 1 (*Task Field & Field Formation*): $tField = FLD_t(FR, ENV)$ *where*, FLD_t: *field formation operator*, *FR is set of functional requirements of task*, *and ENV is set of environment constraints*.

At a given time, an agent’s behavior can be self-regulated based on its “*field position*” at that moment which is determined by the task requirements and the environmental situation. We assume that each agent is equipped with needed sensors and afield formation operator.

To demonstrate that more complex tasks require more complex systems, a measure of task complexity is needed. Various measures of task complexity have been proposed [16, 17]. We define task complexity to have four components: composite complexity, object complexity, coordination complexity and dynamic complexity.

Tasks are composed of various functions. Typically, a function can be represented as a pair of<verb><object> (e.g., <push><box>). As the number of distinguishable verbs (i.e., actions) of a task increases, agents need to be more knowledgeable in order to perform the task. Therefore, the number of distinct verbs can be used as a measure of an aspect of task complexity, called *composite complexity*. We have,

Definition 2 (*Composite Complexity*): $CoC = \sum_{i=1}^{|V|} 1 + \sum_{i=1}^{|0|} 1$ *where*, $V = \{v_1, \ldots, v_n\}$ *is the set of all distinguished actions and* $O = \{o_1, \ldots, o_m\}$ *is the number of all distinguished objects*; $|V| = n$ *and* $|O| = m$.

In addition to the number of objects, the properties of the objects involved in a task, such as shape, dimension, and mass, also contribute to the task complexity. Therefore the number of parameters used to describe the distinctive objects can be used to define the *object complexity* of the task. The more the parameters are, the higher the complexity level is. We define object complexity of a task as,

Definition 3 (*Object Complexity*): $ObC = \sum_{i}^{N} P_1$ *where*, *N is number of unique objects involved in the task*, P_i *is number of parameters for describing object i*.

For a given task, in addition to the number of actions, there can be various relationships between these actions that must be maintained for the completion of the task. Examples include timing between actions (e.g., parallel, sequential, or specific delay) and number of relative occurrences. The existence of these relationships requires coordination of actions within an agent and between multiple agents. This change in phase of agent action can add a fair amount of *coordination complexity* to the system.

Definition 4 (*Coordination Complexity*): $CoC = \sum_{i \in V} \sum_{j \in V} r_{ij}$ *where, r is action relations between action (verb) i and j, which can be sequential or reciprocal.*

Another kind of task complexity deals with the changing environment. When environment changes, task field will vary. Depending on the degree of variation, an agent's behavior for action and coordination should be adjusted. We can capture such *dynamic complexity* by the sum of differences across a certain time period for the above mentioned three complexity components, as described in [16]. We have,

Definition 5 (*Dynamic Complexity*): $DyC = \left|CpC_{(t+1)} - CpC_{(t)}\right| + \left|ObC_{(t+1)} - ObC_{(t)}\right| + \left|CoC_{(t+1)} - CoC_{(t)}\right|$

The overall task complexity is the weighted sum of the abovementioned complexities:

Definition 6 (*Task Complexity*): $TC = W_{cp}CpC + W_{ob}ObC + W_{co}CoC + W_{dy}DyC$ *where,* W_{cp}, W_{ob}, W_{co}, W_{dy} *are the weights assigned to each complexity measure.*

Examples of how these complexity measures are applied and computed are given in the case study section.

Agent and Social Structures

In the CSO framework, we treat mechanical components as mechanical cells (mCell, i.e., agents). Following our previous work [4], we have following definitions:

Definition 7 (*Mechanical Cell*): $mCell = \{C_u, S, A, B\}$ *where Cu: control unit;* $S = \{s_1, s_2, \ldots\}$: *sensors/sensory information;* $A = \{a_1, a_2, \ldots\}$: *actuators/actions; B: designed behavior, or design information (see definition 4 below).*

Mechanical Cell is the smallest structural and functional unit of a CSO system. Although for a CSO system design, the appearance or the structure of its *mCells* may be different, a *mCell* should be able to sense the environment and process material, energy and/or information as their actions.

Definition 8 (*State*): $State = \{S_C, A_C\}$ *where* $S_C \subset S$ *and* $A_C \subset A$ *are currently sensory information and actions, respectively.*

State is used to represent the situation. It is the combination of the current sensor information *Sc* and current actions *Ac*.

Definition 9 (Behavior): $b = \{S_E, A_E\} \rightarrow A_N$ *where* $S_E \subset S$ *and* $A_E \subset A$ *are existing sensor information and actions, respectively; and* $A_N \subset A$ *are next step actions.*

A behavior *b* is the designed action for given situations or states. The *Cu* of the *mCell* should be able to judge the situation and make decisions on next actions. The design information of a CSO system is the fully developed behaviors for each *mCell.*

One important feature of a CSO system is that each agent is self-interested; they always seek attractions (i.e., attractors) that make them "happier". It is this self-organizing behavior that makes the overall system robust and adaptive to change. However, such self-organizing behavior must be effectively guided so that structures, and therefore complexity, can emerge and the overall system can be functional. In this research, the notion of *satisfaction* is used to capture the happiness of an agent in choosing their actions. For a single agent without considering the existence of other agents, its satisfaction can be defined as below:

Definition 10 (Agent Satisfaction): $Sat_{Beh_i} = Eff(Beh_i, tField)$ *where,* $Beh_i = \{beh_1,\ldots,beh_n\}$ *set of behaviors available to agenti, Eff returns effectiveness profile of* Beh_i *in the current t Field.*

An individual agent's satisfaction is a function that maps the all available behaviors to the effectiveness with respect to its task field. The values of an agent's satisfaction for each possible behavior in the current task field constitute a profile of probabilities of executing for all possible behaviors. This is identical with the FBR described in [4].

Along the similar line of thinking about task complexities mentioned above and by focusing on the physical and effective features [15, 18], we consider the complexity of an agent in terms of the agent's number of actions, number of behaviors and communication capacity (e.g., range and number of channels). We have,

Definition 11 (Individual Agent Complexity): $C_{agent_i} = N_a + N_b + C_{Com}$ *where,* N_a *is the number of actions,* N_b *is the number of behaviors,* C_{com} *is the communication capacity.*

Definition 12 (System's Agents Complexity): $C_{agents} = \sum_{i=1}^{N} C_{agent_i}$ *where, N is the number of agents.*

To increase the emergent complexity and level of sophistication of a multi-agent system requires devising order into the system, as indicated in Fig. 1. In this research, we devise order by introducing social structures among agents. More specifically, we apply the graph theory principles to capture the interactions among agents.

Assume G is a set of all possible graphs that can be formed by N agents $Ag = \{a_1, a_2,\ldots,a_N\}$. Then we define

Definition 13 (Social Structure): $G(t) = (N, E(t))$, *where, N is the number of agents, N is the number of agents, E(t) is the links of interactions/relations between agents at time t.*

As shown above, social structure $G(t)$ is a function of time and is directly dependent on the evolution of agents' interactions. For simplicity, we assume agents are constant nodes in the graph while edges between the nodes changes over time resulting in a dynamic structure.

In CSO systems, the social structure represented as connectivity graph is realized by defining social rules that specify how agents interact with each other. These social rules can be general (e.g., "move to similar direction with neighbors) or task specific (e.g., "move closer with neighbors in on the edge of a box"). We define social complexity measure of agents based on their connectivity graph that originates from social rules. This type of graph complexity is notably similar to the complexity measures defined in molecular chemists [19]. The vertex degree magnitude-based information content, I_{vd} is based on Shannon entropy and defines information as the reduced entropy of the system relative to the maximum entropy that can exist in a system with the same number of elements. The analysis has shown that the I_{vd} index satisfies the criteria for a measure of network complexity. It increases with the connectivity and other complexity factors, such as the number of branches, cycles, cliques, etc.

Definition 14 (Social Complexity): $SC = \sum_{i=1}^{N} d_i \log(d_i)/N$ *where,* d_i *is the degree of each node i (how many other agents are communicating with agent i).*

Social Rule Based Behavior Regulation

The main objective of this research is to explore ways to facilitate emergence of order and therefore complexity so that a CSO system can deal with more complex tasks. We want to devise dynamic structuring methods that can help guide agents to self-organize. We take a social rule based behavior regulation approach and explore various local and bottom up social relations to achieve dynamic social structuring.

Generally speaking, the deficiency of disorderliness or disorganization can be divided into two categories. One is "conflict deficiency" and the other "opportunity-loss deficiency." For simple tasks (e.g., push a box to a destination in an open space) where individual agent's "goal" is mostly consistent with the system goal, the agents' effort can additively contribute to the system overall function. When tasks become more complex, conflicts between agents' actions (e.g., push box in opposite directions due to space constraints) may occur and cooperation opportunities may be lost.

In order to minimize the conflict between agents and exploit cooperation opportunities, social rules and social relations can play an important role. A social rule is a description of behavioral relationship between two encountering agents that can be used by the agents to modify their otherwise individually, rather than socially, determined actions. Two agents acting on a give social rule are said to be engaged in a social relationship. Based on *definition 13* mentioned above, when agents are engaged in social relations by following social rules, social structures emerge, leading to more order and higher complexity of the system.

To avoid conflicts and promote cooperation, social rules can be defined to specify which actions should be avoided and which actions are recommended for given conditions. The conditions are often task domain dependent, although they can also be general. We have,

Definition 15 (*Social Rule*): *sRule* $= < C, ForA, RecA >$ *where C is a condition specifying a set of states*; *ForA*: *forbidden actions for states specified by*; *RecA*: *suggested action.*

Social rules defined above introduce relations among encountering agents. It is conceivable that when an agent encounter neighbors and neighbors encounter their neighbors the cascading effect may lead to a large scale network structure with varying densities. The distribution of such densities can be defined as a *social field* in which every agent has its own position and the awareness of the social field allows agent to reach (i.e., be aware of) beyond the encountering neighbor agents. We have,

Definition 16 (*Social Field*): $sField = FLD_S\,(sRule)$ *Where FLDs is the field formation operator*; *sRule is a social rule.*

Social field adds another layer to the design of CSO systems as a helpful mechanism to secure unity in the system. We will explore its effect in future research. In this research, the focus is put on allowing agents to adjust their otherwise individual satisfaction behavior (see *definition 10*), based on applying social rules to the encountering neighbor agents. This social rule based behavior regulation (SRBR) can be defined as follows.

Definition 17 (*Social Rule Based Behavior Regulation*): $SocSat_{Beh_i} = SRBR(Sat_{beh_i}; SR_i, NA_i)$ *Where*, *SRBR is social field based regulation operator for behavior correction*; Sat_{beh_i} *is tField based behavior satisfaction* (*see definition 10*); SR_i *is set of social rules*; NA_i *is set of encountering neighbor agents*; $SocSat_{Beh_i}$ *is socially regulated behavior satisfaction.*

The above is a general definition. To apply SRBR, an agent needs to (1) generate its independent satisfaction profile through FBR (see *definition 10*), (2) identify and communicate with its neighbors, (3) possess social rules, (4) know which rule to apply for a given situation, and (5) know how to generate new social satisfaction behavior. Each of the five steps can be task domain dependent. In the following section, we discuss how these steps can be implemented and the above mentioned concepts be applied.

Case Study

The objective of case study is to explore and demonstrate how social rule based behavior regulation can increase the order, and therefore the complexity, of the overall system and how this increased order is essential for dealing with more complex tasks. To pursue this objective, we developed a multi-agent simulation system based on the Net Logo platform [20], a popular tool used by researchers of various disciplines.

Figure 2 illustrates the design of simulation based experiment. As independent variable, two strategies were explored, with social structuring (SRBR), and without social structuring (FBR). Control variables are used to test different task and agent situations. Two tasks were tested, pushing a box without an obstacle (simple) and pushing a box with an obstacle (complex). For all settings, we measure success rate, time duration (number of steps) and total effort (total distance the agents traveled) as dependent variables.

Tasks

The box-moving task used for the cases study is illustrated in Fig. 3. Multiple agents intend to move the box to the goal “G”. Given that the canal becomes narrower, the agents must rotate the box to horizontal as it gets closer to the entrance of the narrowing part. Furthermore, there can be an obstacle “obs” on the way.

The specific tasks can be expresses as follows.

T1 = *<Orient><Goal>*
T2 = *<Move><Box>to<Goal>*
T3 = *<Rotate><Box>*
T4 = *<Move><Box>away from<Wall>*
T5 = *<Sense><Social Field>*

The task fields include the attraction field from the “goal” and the repulsion fields from the walls as well as the obstacle if present, as indicated in Fig. 3. For the “goal” field, a gravity-like field is applied, and for the “walls” and the “obstacle”, a gradient based repulsion distribution is introduced to provide “warnings” of

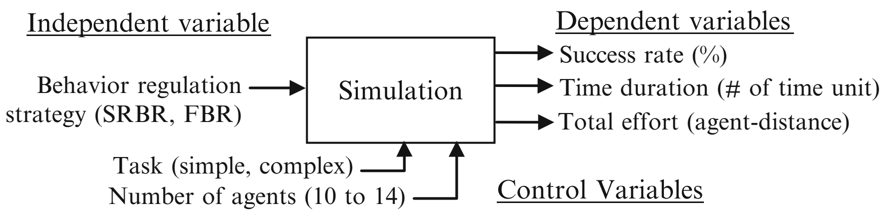

Independent variable
Behavior regulation strategy (SRBR, FBR)
Simulation
Dependent variables
Success rate (%)
Time duration (# of time unit)
Total effort (agent-distance)
Task (simple, complex)
Number of agents (10 to 14)
Control Variables

Fig. 2 Experiment design

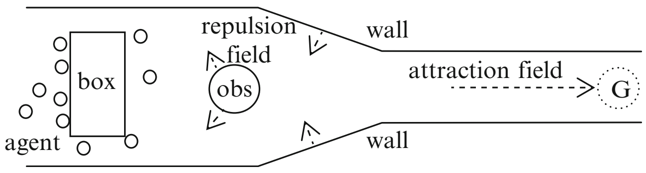


Fig. 3 Box-moving task used in case studies

Table 1 Complexity measures of various box-moving situations

Situation	0: open space	1: with wall	2: with wall+obs
Complexity	7	12	13

collision as agents get closer to them. The gradient distribution of the constraints (i.e., walls and obs) together with the sensory range of agents determine how much in advance agents can predict the collision and find ways to avoid it. In this simulation study, *higher positions* (i.e., *higher value*) in the field are more desirable to agents. For moving the box, an agent always tries to find a “low field position” around the box and from there to move the box toward a “high field position” which is often, but not always, the “goal” position.

We calculated the *object complexity* for the box which is the main object. The characteristics of the box include its dimensions *width* and *length* and its *orientation* angle. For simplicity, we consider the angle to be 90°. Thus the objective complexity sums up to two for this item. There is one reciprocal (i.e., move and rotate) and two sequential activities (i.e., direct→move, and direct→rotate) that are interacting with each other. Therefore, the coordination complexity consists of three interconnected actions resulting in having the complexity of 3 for this portion leading to the total complexity of 12 for “with wall” situation, as indicated in Table 1. The based on the similar calculation, the “open space” and “wall+obs” situations have complexity value of 7 and 13, respectively, shown in Table 1.

The system is composed of n agents: $A = \{a_i\}(i = 1, \ldots, n)$. The initial positions of agents are randomly assigned but are always on the left side of the box. Guided by the task-field of attraction and repulsion, each agent contributes to the correct movement of the box in a way that the emergent movement of the box is toward the goal. Although this strategy (i.e., “non-social”) works well for “open-space with a few obstacles” [4] when more constraints, such as “wall” and more “obs”, are added, new strategies (e.g., “social structuring”) are needed.

Social Rules

As mentioned above, social rules usually are designed to allow agents to avoid conflicts and/or to promote cooperation. In this case study, the social rules are set to

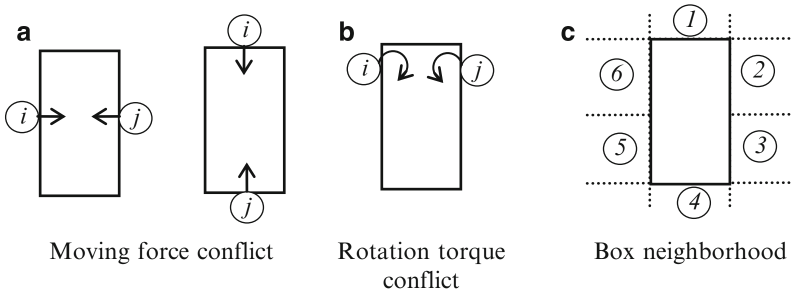


Fig. 4 Possible conflicts of agents *i* and *j*; and box neighborhood

provide guidance for agents to become aware of, and subsequently avoid, potential conflicts. Figure 4(a) and (b) illustrate possible force and torque conflicts between agents *i* and *j*, respectively.

To facilitate description of rules, we introduce the "box neighborhood" by defining six zones, as indicated in Fig. 4c. Agents are aware of their location, i.e., their zone. Furthermore, they can broadcast their location and field density value to neighbor agents. The communication rule follows:

Social rule 1 (*communication rule*): <*condition*: *enter box neighborhood*><*recommended action*: *broadcast* [*location*] *and* [*field strength*]>

When an agent receives broadcast from an agent in the neighborhood, it will attempt to determine if a force conflict or a torque conflict exists and then decide if it will take the recommended actions provided by the following conflicting avoidance rules:

Social rule 2 (*force conflict rule*): <*condition*: *force conflict*><*forbidden action*: *push in opposite-direction in opposite zone*><*recommended action*: *find a new location*>

Social rule 3 (*torque conflict rule*): <*condition*: *torque conflict*><*forbidden action*: *push in opposite-direction in opposite zone*><*recommended action*: *move to next neighbor zone*>

Agents have the option to ignore any or all of the above three rules. When the probability for agents to follow the rules decreases, we say that the system is less socially active, and otherwise more socially active.

Results

Figure 5 illustrates a series of screenshots of a typical simulation run. The large box appeared to be a collection of small boxes because of an implementation difficulty. It should be considered as a single large box.

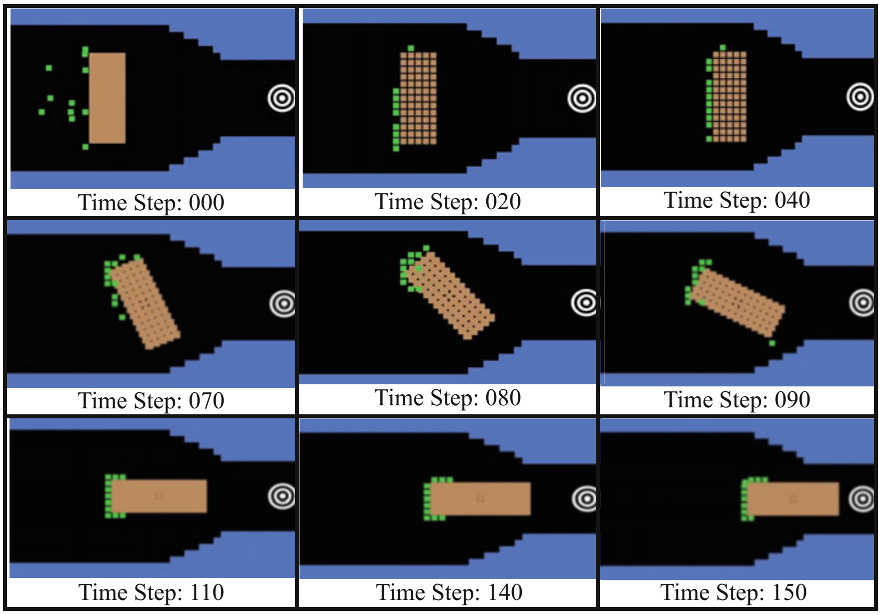


Fig. 5 Screenshots of a typical simulation run

Figure 6 illustrates the comparison of success rate results for social (SRBR) and non-social (FBR) strategies for the “with wall” situation with varying number of agents. All results indicated in the graphs are averages of 100 simulation-runs for that specific setting. For non-social strategy (i.e., no social rule and no structuring), the success rate for 10-agent case is 0; no simulation runs could complete the task. Adding more agents increases the overall system complexity, resulting in better success rate. It can be seen that for this specific case study, 11-agent and 13-agent appear to be *critical* numbers by which the success rate jumps. The social structuring approach proves to be more reliable. The success rate remains 100 % for all agent number settings. The increase of system complexity from adding social rules, and consequently social structures, has made the system more effective to deal with complex tasks.

Figure 7 shows the comparison of total effort and time duration for completed (i.e., successful) simulation runs for non-social and social strategies. For non-social strategy, because no simulation run was successful for ten-agent or less case, there was no data for comparison. It is interesting to see that for the 11-agent case, the absence of social rules and structuring has made the system more efficient. This means that for the 60 % completed runs (see Fig. 6), no social rule is more efficient than having social rules. This is because social structuring incurs “over-head” in task processing. However, the cost for this added efficiency is the 40 % failed runs.

For cases where agent number is larger than 11, the social and non-social have the comparable success rates (see Fig. 6), but the social structuring strategy appears to be more efficient in terms of both effort and time duration. The implication of

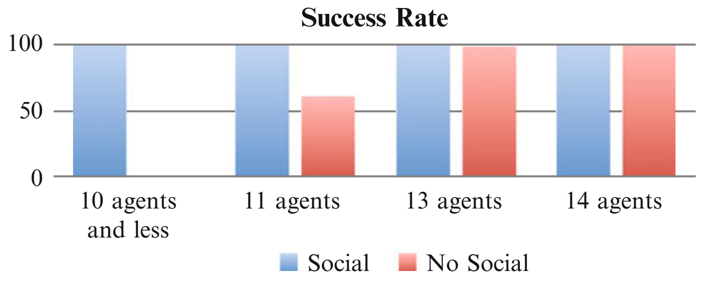


Fig. 6 Success rate comparison for social (SRBR) and non-social (FBR) strategies for the "with wall" situation with varying number of agents

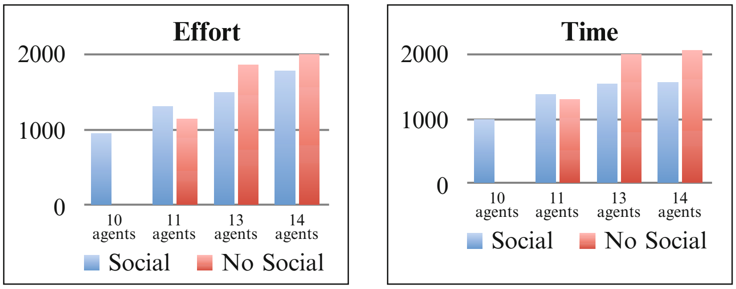


Fig. 7 Effort and duration time comparison for social (SRBR) and non-social (FBR) strategies for the "with wall" situation with varying number of agents

these results is important: while increasing complexity from (b) to (c) in Fig. 1 can be realized by either social structuring or adding more agents, the "impact" of them is different. Adding not-enough agent-power may run risk of failures and adding too much agent-power may lead to waste of time and effort. On the other hand, adding proper social structuring may remove the failure risk and maintain an adequate level of efficiency.

The change of social complexity over simulation time in a typical simulation run with social structuring strategy and 12 agents are shown in Fig. 8. As shown in the figure, social complexity increases when agents start to communicate with each other by following *social rule 1* and "help" each other by following social *2* and *3* when rotating the box in the middle of the process. Social complexity through social structuring varies over time; it increases when needed by the task situation (rotate the box) and decreases when the situation is resolved. This task driven variability is the key difference from the agent complexity obtained through adding more agent-power. While adding more agents somehow relies on "randomness" to increase the success rate and consequently looses efficiency, social rule based self-organization builds competence through local, bottom-up but explicit structuring efforts.

To further explore how more complex tasks demand social structuring, we carried out simulations for the "with wall+obs" situation. The task complexity

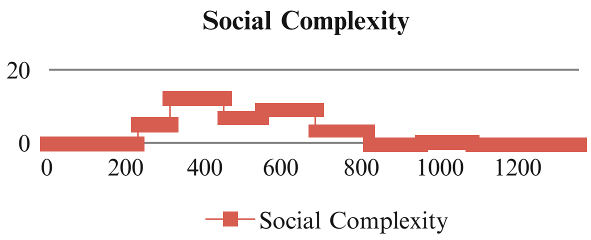


Fig. 8 Social complexity during the process of moving box towards goal with SRBR strategy and 12 agents

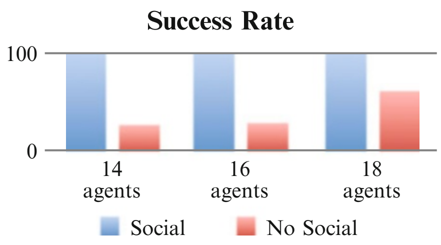


Fig. 9 Success rate comparison for social (SRBR) and non-social (FBR) strategies for the "with wall+obs" situation with varying number of agents

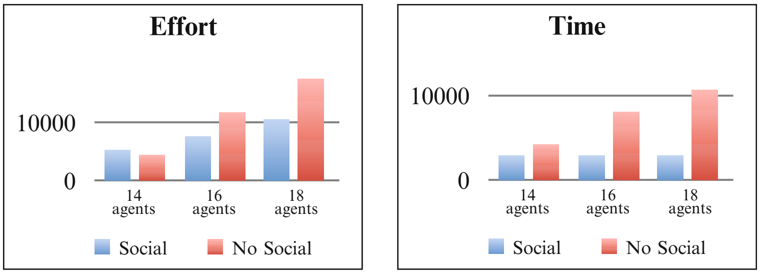


Fig. 10 Effort and time duration comparison for social (SRBR) and non-social (FBR) strategies for the "with wall+obs" situation with varying number of agents

measure for this situation is 13 (see Table 1), more complex than the "with wall" situation. For this situation we only explored the cases for 14–18 agents. Figure 9 shows the success rate comparison of two strategies and Fig. 10 the comparison of effort and time duration.

It can be seen from Fig. 9 that the success rate for non-social strategy decreased dramatically even with more agents (see Fig. 6). However, the social rule based structuring strategy remains to be 100 % successful.

For effort comparison, the non-social strategy is again more efficient than the social 1 for the 14-agent case, as shown in Fig. 10. However, its success rate is only 23 % (see Fig. 9). Overall, the efficiency for non-social strategy is much worse than that for the social strategy. By comparing Fig. 10 with Fig. 8, it can be seen that the more complex task "with wall+obs" is more in need for emergent structural complexity of the system. However, when more agents are added into the already social-rule based system, there is only increase of effort and no improvement of time duration, as shown in Fig. 10. From the above results, it can be seen that devising proper social rules and adequate number of agents is important for designing CSO systems.

Concluding Remarks

As domain tasks become more complex, the engineered systems become more complex by moving from rigid and tightly organized formations into those of more components and more interactions. A potential issue with this top-down or ordered-to-disorder approach is the unintended and unknown interactions that may cause failure of the whole system. An alternative approach is to start with simple and disorganized agents and then move bottom-up and disordered-to-ordered by devising dynamic structures through self-organization. In this research, we explored the sources of task complexity by defining various complexity types and investigated how social rule based behavior regulation can be applied to allow dynamic structures, hence system complexity, emerge from self-interested agents. The case study results have demonstrated the potential of effectiveness of our proposed approach and shed some useful insights.

- Increasing complexity from disorder can be achieved through adding more agents or devising structures. However, the former only has limited effect. When tasks become more complex, adding agents can hardly reach 100 % success rate and the efficiency for the successful runs is low. On the other hand, devising dynamic structures can make the system more adaptable. Not only the success rate is always 100 % but the efficiency is well maintained with changing task complexity (from "with wall" to "with wall+obs") and varying number of agents (from 8 to 18). This result is consistent with Huberman and Hogg's [15] conjecture that higher structural complexity makes system more adaptable.
- When a relatively disordered system can complete a task by a certain probability, for this completed task, its efficiency can be better than structured systems; and this happens only for a small window of number of agents. The reason behind can be that dynamic structuring incurs over-head. However, the efficiency gain of the disorderliness is based on the high risk of failures.
- There can be tipping points of matching between the task complexity and system complexity. Adding one more agent from 10 to 11 can increase the success rate from 0 % to 60 % (Fig. 6), from 16 to 18 causes change from 25 % to 60 % (Fig. 9). This tipping point phenomenon can be due to the lack of social structuring of the system or it may be a result of mismatch between the highly complex task and not-so-complex system. Future work is needed to understand the real causal relations.

Our ongoing work explores the properties of various types of task complexity and their demands for corresponding types of structural complexity of the CSO system. Along the way, we will include more close-to-real engineering tasks and gradually make our CSO systems more real and practically functional.

This paper is based on the work supported in part by the National Science Foundation under Grants No. CMMI-0943997 and No. CMMI-1201107. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Ashby WR (1956) An introduction to cybernetics. Taylor & Francis, London
2. Zouein G, Chen C, Jin Y (2010) Create adaptive systems through 'DNA' guided cellular formation. Des Creat 2010:149
3. Chiang W, Jin Y (2011) Toward a meta-model of behavioral interaction for designing complex adaptive systems. IDETC2011-48821, 29–31 Aug
4. Chen C, Jin Y (2011) A behavior based approach to cellular self-organizing sys. design. IDETC2011-48833, 29–31 Aug, Washington, DC
5. Simon HA (1962) The architecture of complexity. Proc Am Philos Soc 106:467–482
6. Williams EL (1981) Thermodynamics and the development of order. Creation research society books
7. Ferguson S, Lewis K (2006) Effective development of reconfigurable systems using linear state-feedback control. AIAA J 44(4):868–878
8. Martin MV, Ishii K (2002) Design for variety: developing standardized & modularized product platform architectures. Res Eng Design 13(4):213–235
9. Unsal C, Kilic H, Khosla P (2001) A modular self-reconfigurable bipartite robotic system: implementation and motion planning, Kluwer Autonom. Robots 10:23–40
10. Yim M, Zhang Y, Duff D (2002) Modular robots. IEEE Spectrum 39(2):30–34
11. Shen WM, Krivokon M, Chiu H, Everist J, Rubenstein M, Venkatesh J (2006) Multimode locomotion for reconfigurable robots. Auton Robot 20(2):165–177
12. Horling B, Lesser V (2004) A survey of multi-agent organizational paradigms. Knowl Eng Rev 19(4):281–316
13. Galbraith JR (1977) Organization design. Addison-Wesley, Reading
14. Brooks CH, Durfee EH (2003) Congregation formation in multi-agent systems. Auton Agent Multi-Agent Syst 7(1):145–170
15. Huberman B, Hogg T (1986) Complexity and adaptation. Physica D 22(3):376–384
16. Wood RE (1986) Task complexity: definition of the construct. Organ Behav Hum Decis Process 37(1):60–82
17. Campbell DJ (1988) Task complexity: a review and analysis. Acad Manag Rev 13(1):40–52
18. Gell-Mann M (1995) What is complexity? Complexity 1(1):16–19
19. Bonchev D (2004) Complexity analysis of yeast proteome network. Chem Biodivers 1 (2):312–326
20. Wilensky U (2001). Modeling nature's emergent patterns with multi-agent languages. Paper presented at Euro Logo 2001. Linz, Austria

Computational Design Synthesis of Aircraft Configurations with Shape Grammars

Matthias Oberhauser, Sky Sartorius, Thomas Gmeiner, and Kristina Shea

Abstract Today, a majority of the produced aircraft share a common baseline design although unconventional designs have shown to be advantageous in certain cases. Using computational design synthesis a constrained solution space can be automatically generated and a high number of design candidates can be quickly explored without fixation on common designs. In this paper, the authors present and discuss a formal, three-dimensional spatial grammar to create different aircraft configurations that adhere to given design constraints. The grammar is built around a case study, the recreation of historic design proposals for a commercially successful, unconventionally configured aircraft, the Lockheed P-38. The results show that the developed 3D spatial grammar is not only able to computationally recreate the six design candidates that were part of the historic designs proposal but to generate further 14 feasible configurations and parametric variants by exploring the solution space more freely and thoroughly. Further, the use of the grammar for computational synthesis of other aircraft designs is discussed.

Introduction

Considering fixed-wing aircraft today, one will see that most have a similar basis when it comes to their design. The configuration of an airplane, meaning roughly the spatial arrangement and number of wings, structural parts, engines, and body parts, is very similar for almost all aircraft that have been produced to date. For multi-engine aircraft, this standard configuration, which will be referred to as the baseline design, consists of a forward wing with nacelle-mounted engines and a single aft tail. Leonard conducted a study covering 292 twin-engine propeller-driven aircraft that had at least a flying prototype. He concluded that this baseline

M. Oberhauser (✉) • S. Sartorius • T. Gmeiner
Technische Universität München, München (Munich), Germany
e-mail: matthias.oberhauser@gmail.com

K. Shea
Swiss Federal Institute of Technology, Zurich, Switzerland

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_2

design accounts for 66 % of all aircraft configurations [1]. In addition, half of the remaining, unconventional designs do not have a production representative, which means that in absolute numbers of built aircraft, the fraction of non-baseline designs is significantly lower.

Searching for reasons for this lack of unconventional designs, some factors emerge: In preliminary aircraft design, engineers often rely on historical data for their first estimation of the vessel's performance, as analytic calculations of aircraft characteristics are often hard to calculate [2]. As seen before, this historical data mainly consists of baseline designs.

In addition, the lack of historical examples of choosing unconventional designs are a reason that aircraft designers rather stick to an evolutionary design approach, i.e., starting from the last design, rather than a revolutionary one. This bias towards known designs is a common phenomenon called design fixation [3].

Yet, if an unconventional design proposal is considered, further issues emerge like the need for research in aerodynamics, propulsion, and structural analysis, with an uncertain outcome [4]. Furthermore, strict safety regulations in commercial aviation have to be met, which could be a great hurdle for such designs as well. However, unconventional configurations could play a critical role in solving current and future environmental challenges of aviation such as fuel and CO_2 reduction or noise abatement. Further, advances in technologies like unmanned aerial systems that remove the requirements for crew compartments and life support systems, thus leading to different safety regulations, enable new concepts [5].

The necessary outside-the-box thinking to come up with unconventional aircraft designs has to start in the early phases of the design process. In these phases, the creation of candidate solutions that explore the solution space, a process called design synthesis, is a key activity. In recent years, approaches and methods to formalize the design synthesis process have been developed that allow for the fast and unbiased computational generation of candidate solutions. This paper presents the use of such a computational design synthesis (CDS) approach using shape grammars in the area of aircraft configuration and discusses how it can help the engineer explore design alternatives and eventually come up with more unconventional designs that fulfill or exceed the design requirements.

Background

The Aircraft Design Process

The conceptional design process in aeronautics consists of *creative synthesis*, *analysis* and *decision making* [6]. In contrast to the other phases, design synthesis focuses on creating alternatives that fit to predefined criteria, i.e. requirements, yet without dominant consideration of analytic aspects, e.g. simulation. These very first design sketches can be merely "back-of-a-napkin" drawings like the sketches in

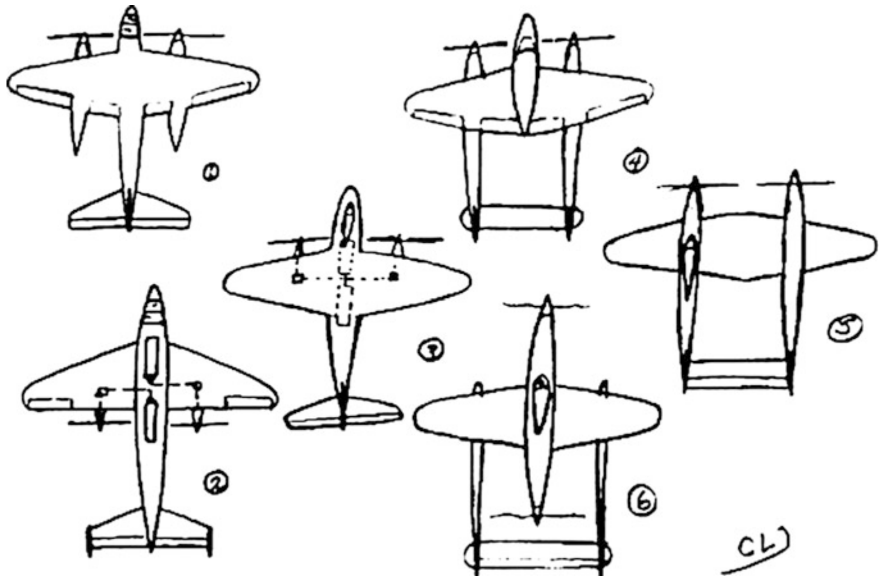


Fig. 1 P-38 conceptual sketches (From [6])

Fig. 1. These simple drawings contain comprehensive details of the desired design as they clearly communicate most of the aspects of the proposed configuration [6].

In the later *analysis* phase, parameters are derived from the conceptual configurations and in the decision making phase, one or more designs have to be chosen or dismissed. Unlike the *synthesis* phase, these two phases require a highly structured and methodical thinking [6].

Shape Grammars

Shape grammars were introduced by Stiny and Gips nearly 40 years ago and soon became popular among architects and urban planners [7]. The basic concept of shape grammars is to define a finite set of grammar rules that operate over a set of labeled shapes, also called a vocabulary. A rule consists of a left hand side (LHS) and a right hand side (RHS) and is applied to a current working shape (CWS). If a labeled shape defined in a rule's LHS is found in the CWS, it will be replaced (in the CWS) with the shape in the rule's RHS [7]. If a set of rules is applied to a CWS in a certain order, given an initial shape, complex shapes can emerge.

Spatial grammars is a general term that includes different grammar representations that define languages of shape, for example, string grammars, set grammars, graph grammars, and shape grammars. This paper uses a set grammar formalism. The process of shape recognition and shape transformation can be done "paper based" as Stiny and Gips used in their initial work or with the use of computer-based implementations. An overview and review of systems was carried out by Gips in 1999 [8] and more recently by McKay et al. in 2012 [9] and Chakrabarti et al. in 2011 [10].

Computational Design Synthesis in Aerospace

There are already some examples of computational design synthesis, or CDS, in aeronautics. A probabilistic model of creating any sort of new objects from a set of given objects was developed by Kalogerakis et al. [11]. Although not specifically developed for aeronautic applications, one presented use case is the synthesis of new aircraft models from an existing set of aircraft. This model does have potential to assist the aircraft design synthesis. Yet, as it covers only morphology, it might be difficult to set constraints for the designs.

A commercial application of shape grammars can be seen at Boeing. Since 1997, a tool called Genisis is used to synthesize and compare solutions for the tubing systems in aircraft [12].

Arnold and Rudolph use a graph-based design language to transform a design model to a manufacturing model automatically [13]. As an example for this process, a computer generated aircraft fuselage panel is transferred to a digital factory.

Another graph-based approach is the research on aircraft cabin seat layouts conducted by Helms [14]. In this project, the arrangement of seats, aisles, lavatories and service areas was automatically synthesized to meet defined constraints like legroom or number of seats. This resulted in a large variety of cabin configurations meeting different performance requirements, e.g. maximum comfort or maximum number of seats.

Another application of CDS in aeronautics can be seen in Stanford et al. [15]. Here the structure of a nature inspired skeleton enforcement for a flapping wing is generated. Therefore, a cellular division method driven by a genetic algorithm is used to create an optimal skeleton wing structure.

There is also research in aeronautics from Stavrev who developed a shape grammar for creating pressurized parts for space stations. In the light of inflatable space habitats, more architectural freedom might be possible which is discovered using shape grammars [16]. However, what has not been investigated so far is the applicability of CDS to aircraft configuration, which is presented in this paper.

Method

In this research, a shape grammar approach for the generation of aircraft configurations represented as three-dimensional models is pursued. To implement the process of shape grammar rule development and rule application, a variety of software is available. For this research *Spapper*, presented by Hoisl [17], is used.

Spapper is an interactive 3D spatial grammar interpreter that is embedded in the open source computer-aided design (CAD) software *FreeCAD*. It uses a set grammar approach that defines a vocabulary of parameterized solid primitives in algebra U_{33}. Both parametric and non-parametric rules can be developed using a GUI, making use of common CAD functions in *FreeCAD* for creating and editing geometric objects [18]. Further, 3D spatial labels can be defined to simplify LHS rule matching, define spatial and state constraints and encode non-geometric information [19]. 3D labels carry information about location and rotation in three-dimensional space. The 3D environment, the support for spatial labels and parametric shape rules as well as the automatic application of grammar rules, including automatic LHS rule recognition under rotation and translation transformations, makes it a good choice for this research. First, a two-dimensional paper-based grammar is developed. This initial step offers the possibility to explore and test the mechanisms for creating aircraft configurations using a grammatical approach with reduced complexity. The gained knowledge is then used in the development of a 3D grammar presented here. Although a two-dimensional shape grammar interpreter would be suitable for creating conceptual design sketches, the integration of *Spapper* in an engineering CAD environment like *FreeCAD* offers a variety of benefits, such as the possibility to create and modify shapes with exact dimensions, to position them spatially, and also to use equations to set proportions or placement locations. Finally, *FreeCAD*'s open *Python* interface makes the system ideal for potential pre- or post processing steps.

As mentioned in previous papers on the development of specific shape grammars, a common method for the evaluation and testing of the expressivity of the grammar is its ability to recreate existing, feasible designs [20, 21]. The grammar can then be used to explore the design space further and generate novel designs. As the aim of the presented research is the generation of unconventional aircraft configurations, a successful example from this class of aircraft was chosen as a case study.

Use Case: P-38 Lightning Designed by Kelly Johnson

In response to a U.S. Air Force (USAF) proposal for a twin engine fighter aircraft in 1937, Clarence "Kelly" Johnson sketched out six possible configurations, as shown in Fig. 1. This design synthesis led to an unconventional design that became one of the most successful USAF aircraft during WWII with over 10,000 units built and Kelly Johnson became one of the most influential aircraft designers of his time [22]. Figure 2 shows the P-38. It has a double tail boom containing two propeller engines and a central nacelle containing armament and pilot.

As this design, in contrast to other unconventional designs, was an outstanding success, and the conceptional design process is well documented, it makes a good case study for demonstrating the potential of spatial grammars in aircraft design.

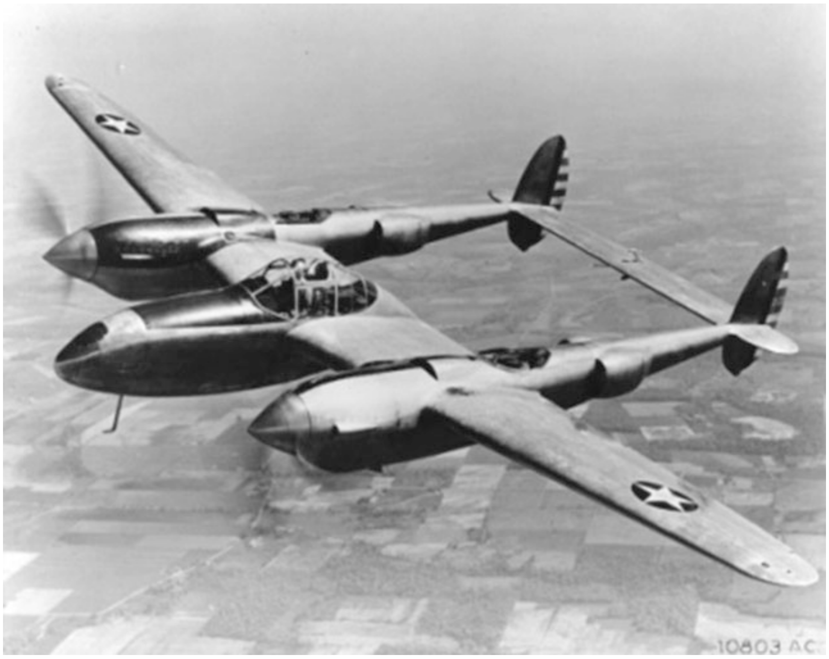


Fig. 2 The YP-38, a pre-series model of the P-38 [23]

Grammar Rule Set Summary

The P-38 has a "tube-wing" arrangement meaning that the aircraft is built from tube-like parts for the fuselage and a number of attached aerodynamic surfaces for wings. As this is the design of the vast majority of flying aircraft [24] it will be the basis for the development of the formal grammar. This decision is made to constrain the general design space and to increase the percentage of feasible solutions while still allowing for unconventional designs. Based on the parts of the aircraft, the rules can be grouped into three categories:

- **Structural Parts** and elements for containing payload, occupants and systems
- **Propulsion Elements**, their position and number
- **Aerodynamic Surfaces** like wings and tails

Consequently, the vocabulary of the aircraft grammar for the P-38 case consists of fuselage parts, structure, engine parts, wings, a cockpit and propellers. With this vocabulary as shown in Fig. 3 and their permutation, a number of configurations can be generated.

All these parts are single parameterized primitives, or Boolean combinations of these primitives, as these are supported well by *Spapper*. Some of these solids, such as the wedge used to represent a wing, do not exactly fit the real physical structure of the respective aircraft part. Still, the representations of the parts are unambiguous

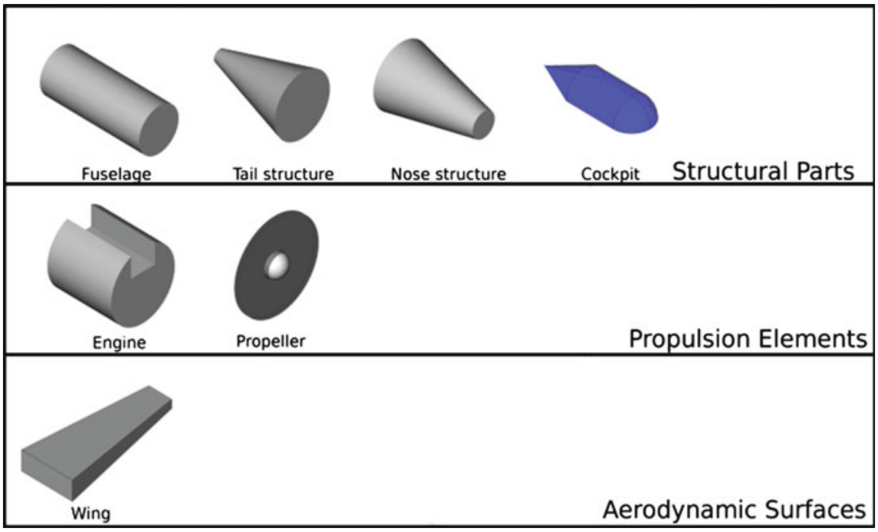


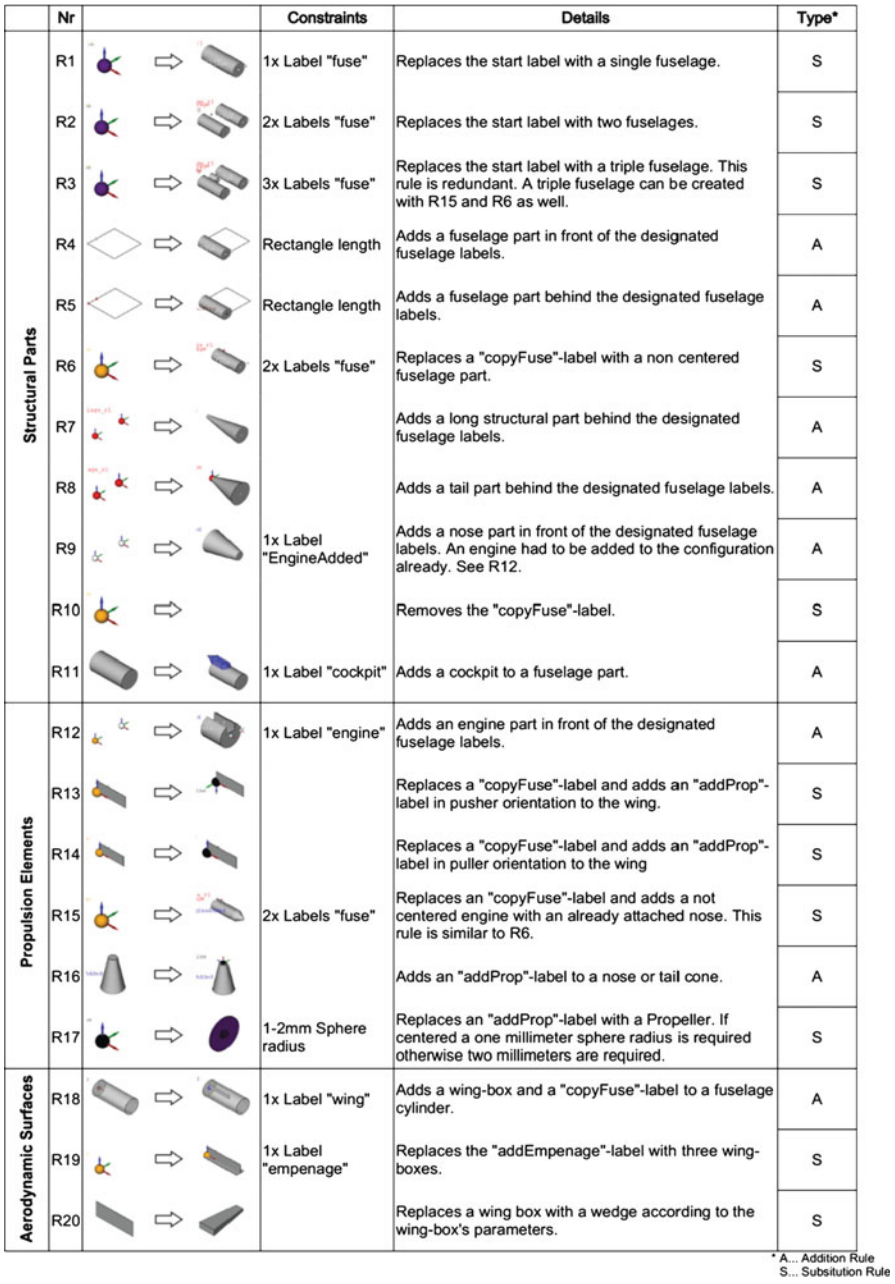
Fig. 3 The vocabulary used for the aircraft grammar

enough for a designer to interpret the resulting designs. And as the aim is not to generate finished, production-ready results but rather to explore the design space and spark a designer's creativity by generating possible configurations, this simplification is valid. Figure 4 contains the complete rule set with a brief description of each rule. In the following sections, not all 20 developed rules for this particular case are discussed, but rather an overview of the design process is given.

Fuselage

The fuselage parts consist of parameterized cylinders. As these could also be of different geometries, the connecting points will be marked with labels. The deviation of the two labels determine the radius of the added fuselage part. The cylinder height and radius are set to be randomly chosen by *Spapper* but within user-given ranges. Using these free parameters, a greater variety of designs emerge without the need to define more rules. Regarding the maximum length of the fuselage, it is important to restrict the design space. This was done using a rectangle whose length restricts the maximum length of the combined cylinders. Figure 5 shows the application of Rule 4.

The use of discrete fuselage parts was chosen to give added wings and cockpits a distinct longitudinal position as seen in Fig. 7. To generate more than one fuselage like the P-38's twin boom design, Rule 6 and Rule 15 copy a fuselage or an engine part laterally. To assure that the copied parts are connected to each other, these copying operations can only take place at fuselage parts chosen to hold the main



	Nr		Constraints	Details	Type*
Structural Parts	R1		1x Label "fuse"	Replaces the start label with a single fuselage.	S
	R2		2x Labels "fuse"	Replaces the start label with two fuselages.	S
	R3		3x Labels "fuse"	Replaces the start label with a triple fuselage. This rule is redundant. A triple fuselage can be created with R15 and R6 as well.	S
	R4		Rectangle length	Adds a fuselage part in front of the designated fuselage labels.	A
	R5		Rectangle length	Adds a fuselage part behind the designated fuselage labels.	A
	R6		2x Labels "fuse"	Replaces a "copyFuse"-label with a non centered fuselage part.	S
	R7			Adds a long structural part behind the designated fuselage labels.	A
	R8			Adds a tail part behind the designated fuselage labels.	A
	R9		1x Label "EngineAdded"	Adds a nose part in front of the designated fuselage labels. An engine had to be added to the configuration already. See R12.	A
	R10			Removes the "copyFuse"-label.	S
	R11		1x Label "cockpit"	Adds a cockpit to a fuselage part.	A
Propulsion Elements	R12		1x Label "engine"	Adds an engine part in front of the designated fuselage labels.	A
	R13			Replaces a "copyFuse"-label and adds an "addProp"-label in pusher orientation to the wing.	S
	R14			Replaces a "copyFuse"-label and adds an "addProp"-label in puller orientation to the wing	S
	R15		2x Labels "fuse"	Replaces an "copyFuse"-label and adds a not centered engine with an already attached nose. This rule is similar to R6.	S
	R16			Adds an "addProp"-label to a nose or tail cone.	A
	R17		1-2mm Sphere radius	Replaces an "addProp"-label with a Propeller. If centered a one millimeter sphere radius is required otherwise two millimeters are required.	S
Aerodynamic Surfaces	R18		1x Label "wing"	Adds a wing-box and a "copyFuse"-label to a fuselage cylinder.	A
	R19		1x Label "empenage"	Replaces the "addEmpenage"-label with three wing-boxes.	S
	R20			Replaces a wing box with a wedge according to the wing-box's parameters.	S

* A... Addition Rule
S... Subsitution Rule

Fig. 4 The set of rules

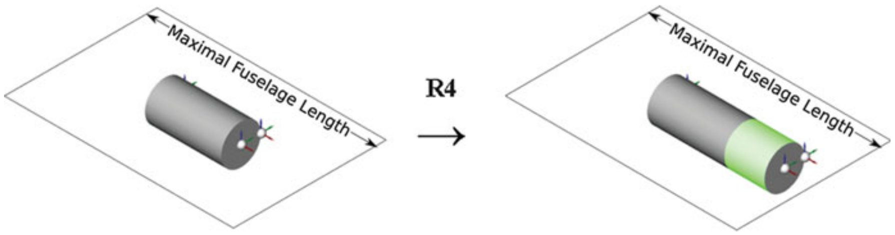


Fig. 5 Rule application for adding a fuselage part

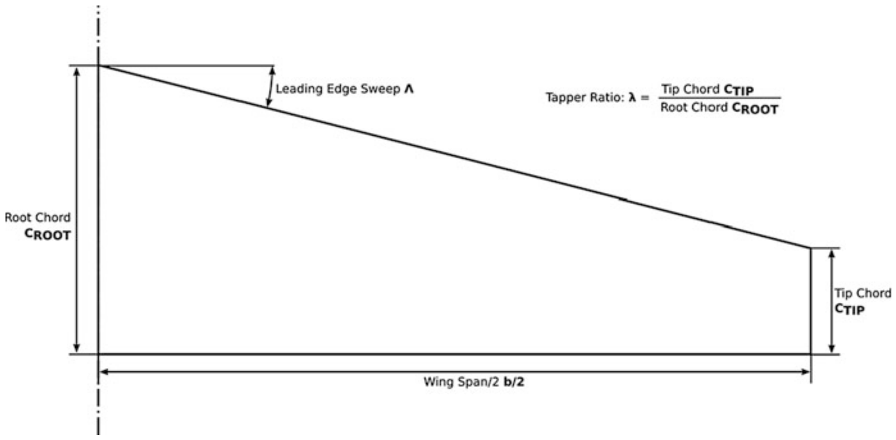


Fig. 6 Parameters of the planform of a wing panel

wing. This is implemented by adding a specific label if the wing is added. Rule 10 removes that label in order to generate single fuselage configurations. The maximum number of fuselages is controlled through a distinct number of labels in the initial shape. Given the P-38 case, a maximum number of three independent fuselages has been chosen.

Wing Panels

The wing panels consist of the primitive wedge, giving a sufficient representation for a conceptional design sketch as in this phase the wing's planform is especially relevant. Consequently, the wedge is parameterized using the parameters described in Fig. 6.

These parameters are set to fixed values for the main wing (Rule 18) and the horizontal and vertical stabilizers (Rule 19) suitable for the use case. Choosing these parameters randomly would result in a high degree of variation but would also

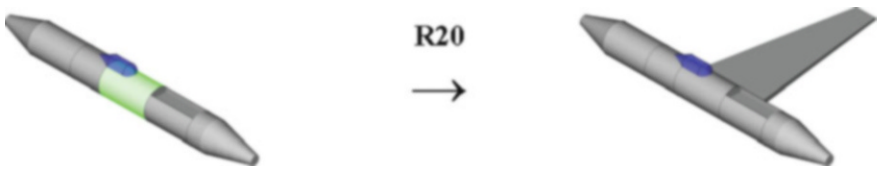


Fig. 7 Application of a wing panel to a fuselage section

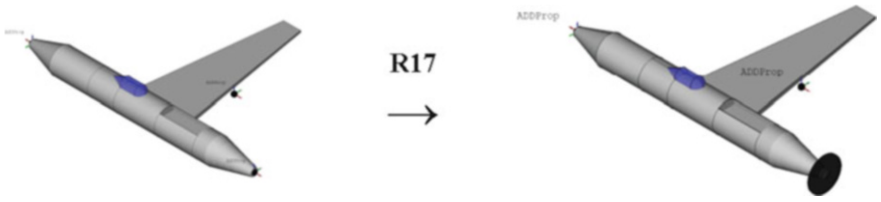


Fig. 8 Application of a propeller

result in a greater portion of the resulting designs being infeasible. The fixed values are stored in the initial shape in the form of a parameterized object, in this case a simple circle. Its' parameters represent the wing's variables, for example the circle's radius equals the wing span. Figure 7 illustrates the application of a wing panel to the fuselage.

As another constraint, both the main wing and the empennage can only be applied once. This is controlled through labels supplied in the initial shape and is discussed in more detail in the results section.

Propulsion

The propulsion consists of a representation of an engine part and propellers, which can be attached either to a wing (Rule 13 and Rule 14) or a fuselage (Rule 16). The set of rules will provide multiple possible positions for the propellers using labels (see Fig. 8). In this use case the maximum number of propellers, which is represented by a sphere's radius in the initial shape, is limited to two. Depending on the position of the propeller, the sphere's radius will be reduced by one (centerline) or two (not centered) millimeters. Consequently, not all possible positions will be used in the design process.

In order to fulfill the requirements of the P-38 case study, the propellers could also be located remotely from the engines driving them, for example, propellers mounted to the wings driven via a chain by central fuselage-mounted engine(s). Although not very common today, the very first powered aircraft, the Wright Flyer and two of the six P-38 design proposals use this configuration.

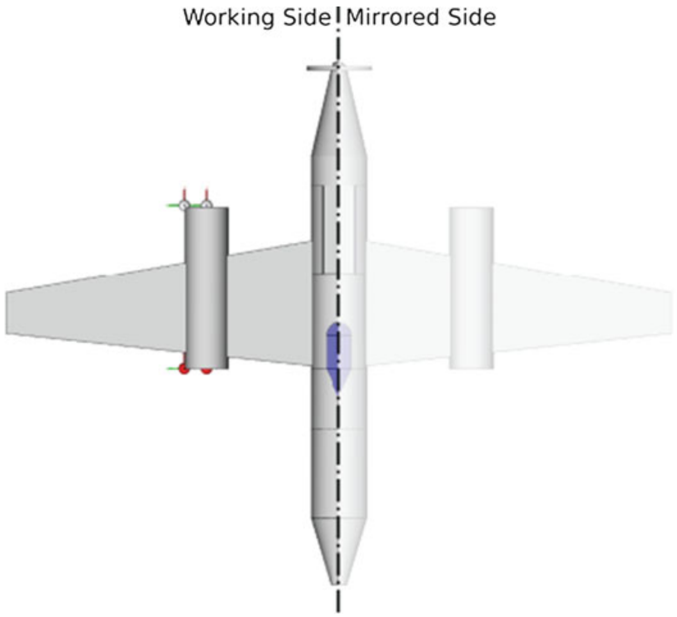


Fig. 9 Working side (*left*) and the aircrafts' mirrored side (*right*)

Symmetry

As presented, several design constraints, like the number of propellers or the geometry of the wings and fuselages, have been considered in the grammar. This is necessary to increase the percentage of feasible designs created. Another important constraint to achieve feasible conceptual designs is symmetry. All airplanes but a few rare exceptions, like the Rutan Boomerang or oblique wing designs, have a symmetrical configuration. Although there might be some advantages, such as the enhanced safety regarding a single engine failure provided by the asymmetric Rutan Boomerang [25], most of the parts of this rule set will be mirrored to generate a symmetric design. The mirroring is conducted when a rule is applied. Therefore, the current working shape (CWS) is divided into a working side and a mirrored side. Figure 9 shows an unfinished design and the two areas.

For all the objects added to the working side, identical objects are added as well. These identical objects are (if necessary) mirrored and moved from the working side to the mirrored side. As there is no mirror functionality that can be used with *Spapper*, the mirroring for non-rotationally symmetric objects is done by changing the objects parameters accordingly. Whereas rotationally symmetric objects can simply be moved to the working side. This process is applied to geometry only. Labels are not affected as they should appear in the working side exclusively where the rules are applied. The mirroring operation takes places automatically during the application of a rule. Therefore a symmetric design is produced after every rule application step. This operation can be omitted if a component should be asymmetric intentionally. In this case study, this applies to the cockpit for twin fuselage designs as it should be placed on one side only.

Results

For the P-38 case study, an initial shape has to be prepared to impose the required constraints and to increase the probability that the historic design proposals seen in Fig. 1 will be generated. As discussed, the constraints can be set using a number of labels and objects in the initial set of shapes. The variables that can be influenced by changing the initial shape are listed in Table 1. These values do not specifically fit the P-38 data but were chosen to generate feasible designs and fit the proposal's proportions.

Figure 10 shows the initial shape based on the variables described in Table 1. For better visibility, some of the objects have been moved and/or scaled.

Another important factor for the outcome is the order of the rule application. In this case, the 20 rules defined will be applied randomly. This means that *Spapper* goes through the list of rules in a random order trying to match the chosen rule's LHS with the working shape. When a match is found, the rule is applied, if not

Table 1 Constraints set in this use case

Constraint	Value	Constraint	Value wing	Value tail
$n_{ENGINES}$	1–2	b	15 m	5.5 m
$n_{COCKPITS}$	1	c_{ROOT}	3 m	1 m
$n_{PROPELLERS}$	2	λ	0.3	0.6
$n_{FUSELAGES}$	1–3	Λ	10°	7°
n_{WINGS}	1			
$L_{FUSELAGE}$	<10 m			

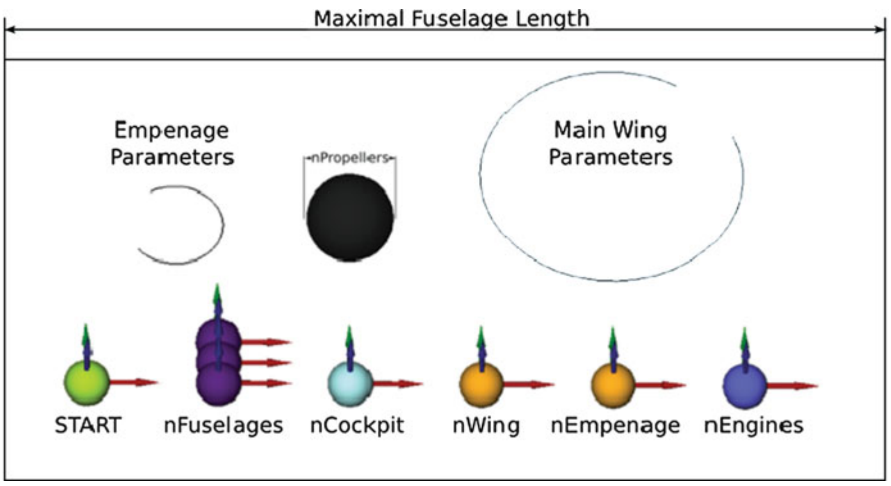


Fig. 10 The initial shape

another rule is chosen. If there are multiple shapes in the current working shape that match a rule's LHS, *Spapper* is set to choose one of the matching shapes randomly as well. This way, once the rule application process is initiated, the designs are created automatically without requiring further user interaction. Each successful rule match is regarded as one application, whereas unsuccessful attempts do not count. The rule application process continues until a user-defined total number of applications is reached or there are no further matches found. The resulting design is one solution. The total number of solutions again can be user-defined, such that more than one solution can be generated by *Spapper* automatically.

As the rules are designed to result in a complete configuration, or design, the rule application has to be continued until there are no further matches to be found. At that point, all necessary rule applications to generate a feasible design have been performed. Consequently, it is ensured that all obligatory elements like wings, an empennage, engines, and propellers are part of the design. The number of applications to return a complete design is variable, but 15–20 successful applications are enough to create all presented historic P-38 designs.

Figure 11 illustrates a whole design generation process. The initial shape (IS) consists of a number of labels, a sphere, two circles and the rectangle that restricts the design space. All these objects store the information that is described in Table 1. For this P-38 design candidate, 18 rule applications are applied. Due to the additive design process, there are numerous rule sequences to reach this outcome.

Using the above constraints, rule set and generative process, a variety of configurations can be created. The design variations stem from the random order of applying the rules, the random definition of variable values, if they are set to be unrestricted or within certain ranges, and the random selection of the matched labeled shape in the current working shape in case multiple labeled shapes match the LHS of a rule. For this use case *Spapper* was set to generate and store 100 solutions automatically, which took approximately 2 h on a quad core 2.7 GHz notebook. Figure 12 shows these configurations in groups based on the classification cycle proposed by Leonard [1]. Here, one representative design for each configuration is illustrated. Both the classification and the choice of the representative design have been carried out manually. As Fig. 12 shows, 19 alternative and distinct configurations are generated by the grammar besides the baseline design. At this point, no assertions on the quality of the generated designs are made as this is not part of the design *synthesis* phase but part of the *analysis* phase.

In Fig. 13, the total number of generated configurations is illustrated. The numbers in the pie chart reference the related configuration numbers of Fig. 12 The green pie sections (aircraft numbers 1–6) show that 22 out of the 100 generated designs fit one of the known P-38 design proposals. This verifies that the grammar meets the goal of being able to generate these known configurations. Further, for 27 of the additional configurations generated, at least one flying prototype exists or existed. This underlines that these designs have already proven to be airworthy, and are hence valid. For the remaining configurations, which account for 72 %, no

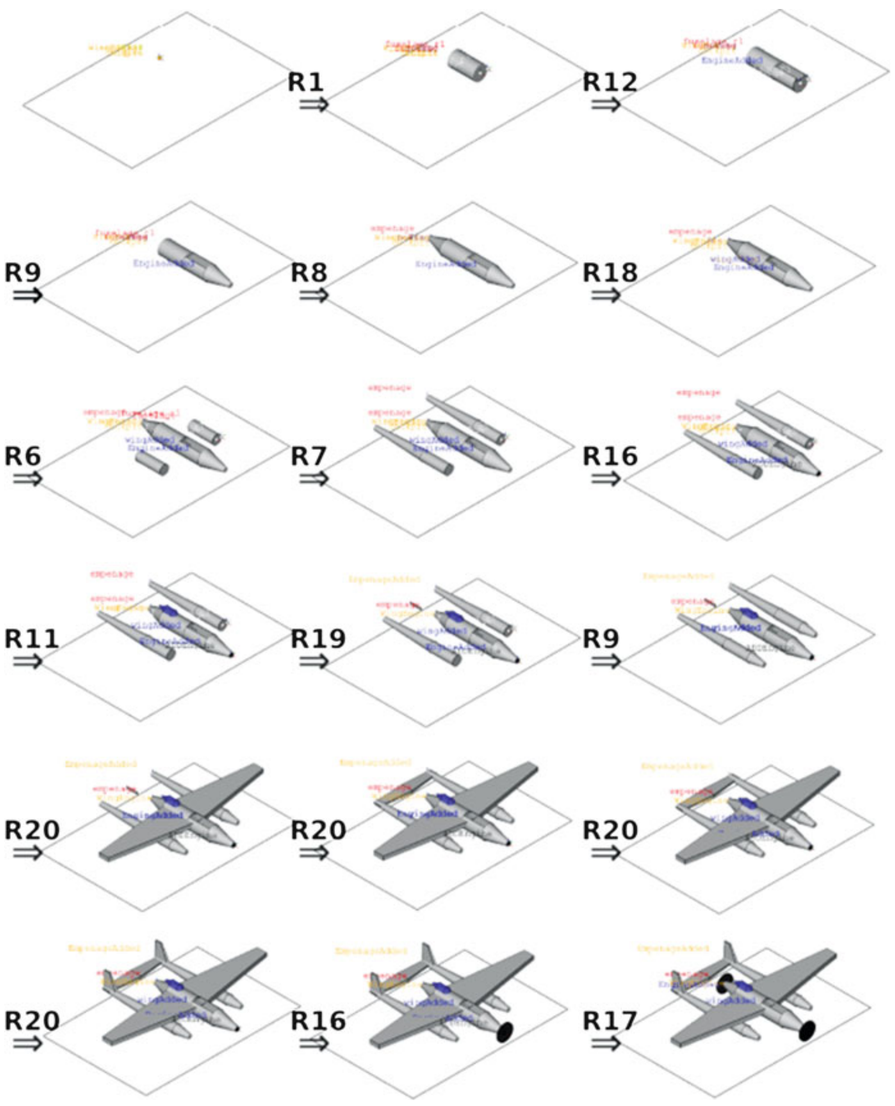


Fig. 11 Step-by-step rule application

known flying counterpart exists. Only one created solution had to be rejected because of a geometry collision. This very low rate of rejected designs is a consequence of the rules' strict constraints. Overall, this illustrates that the grammar is capable of generating both known and new designs.

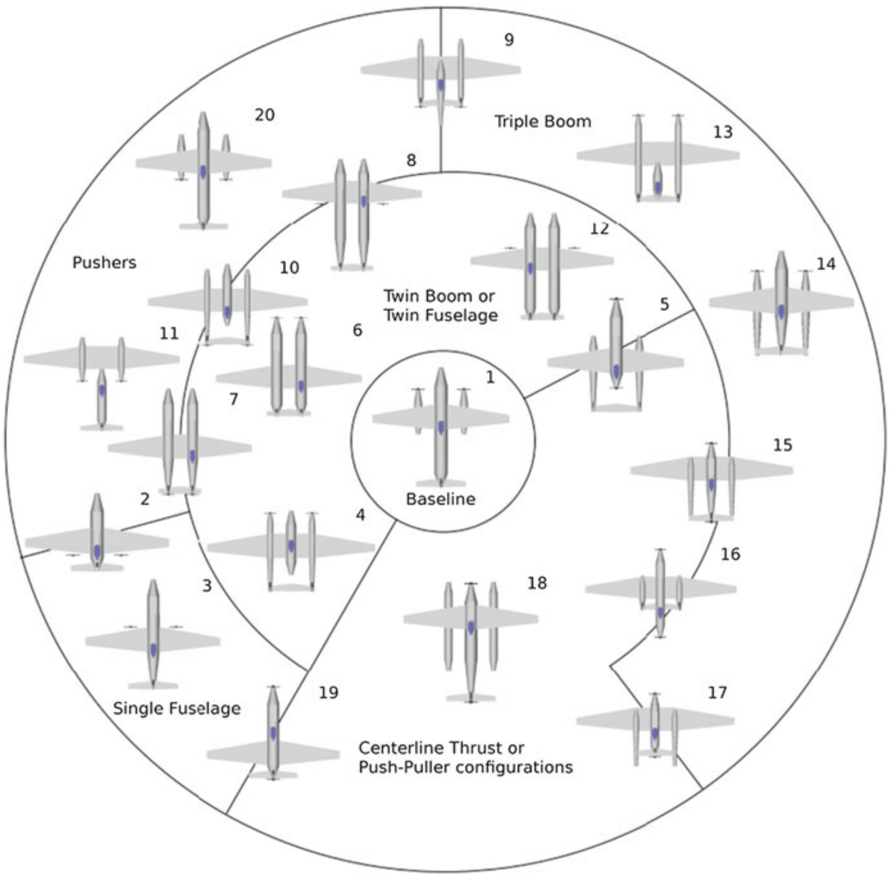


Fig. 12 Emerged aircraft configurations. Classification based on Leonard [1]

Application to Further Case Studies

In the initial shape, requirements on specific parameters of the final design can be set. This controls the outcome of the rule application process. By changing the Initial Shape and these parameters, the grammar can be applied to other case studies. In Fig. 14, two design candidates were generated with a modified initial shape. The third design resembling an X-Fighter needs also modifications in the rule set itself. Here, the dihedral angle of the wings and the length of the nose are modified. For different scenarios more modifications might be needed depending on the requirements. For example, the propeller geometry could be replaced by a jet engine to create a modern fighter design. The vocabulary could also be extended with more propulsion or structure elements, such as engines, inlets or aerodynamic surfaces, to create more diversity.

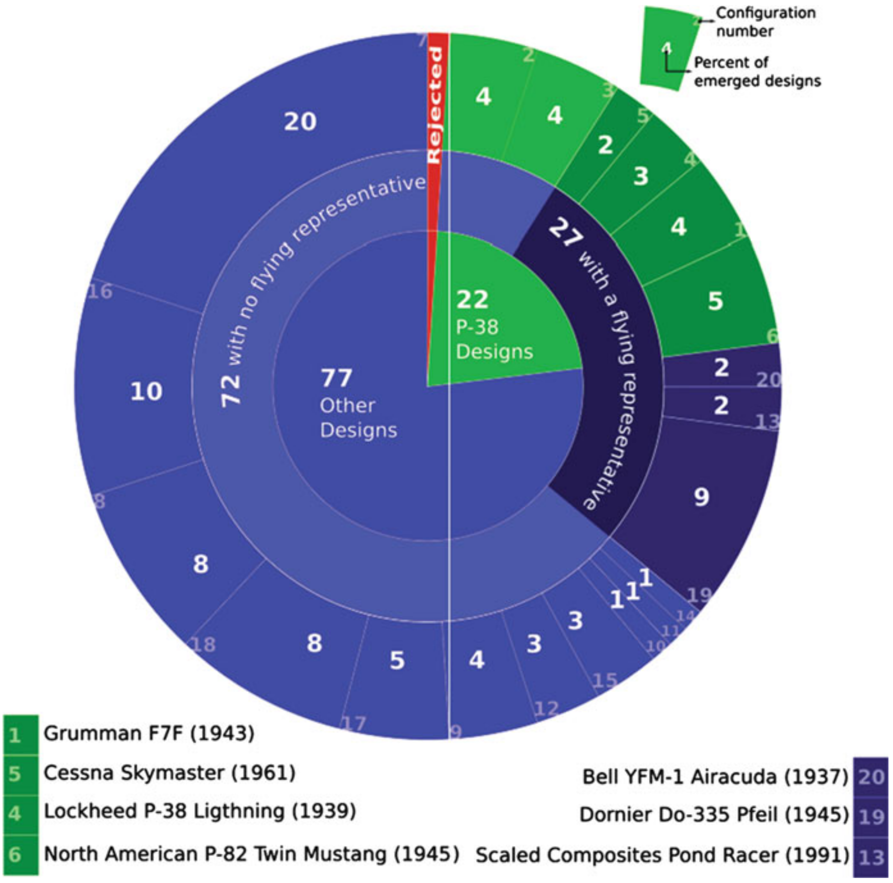


Fig. 13 Quantitative analysis of the emerged configurations

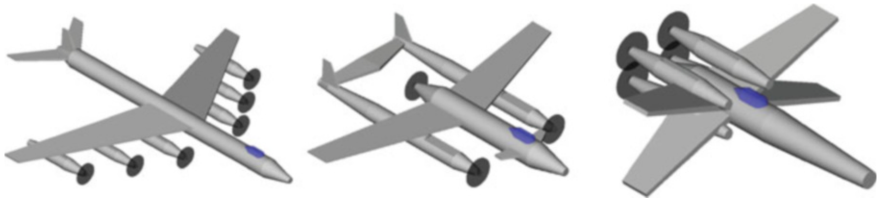


Fig. 14 Designs with a different initial shape or modified rules

In these examples, the rule application process was conducted manually, i.e. the selection of the rules and which rule match to apply in the design was carried out by a human designer. Automatic rule application is also possible but might result in a higher fraction of infeasible designs as the constraints in these designs have been relaxed. Still, a subsequent analysis of automatically generated designs can be carried out to identify infeasible designs.

Discussion

This paper presents a shape grammar that is capable of creating a variety of three-dimensional twin-engine aircraft designs with a focus on unconventional tube-and-wing configurations. The grammar is developed and applied using the open source software tool *Spapper*. As an initial case study, the grammar should be able to generate the designs proposed by Kelly Johnson for the P-38 aircraft (see Fig. 1), an example of a commercially successful unconventional tube-wing-style aircraft. The grammar development results in a set of 20 rules that can be applied to a defined initial shape. The rule set and the initial shape, by using three-dimensional labels and parameterized objects, incorporate several design requirements and constraints that are necessary to yield feasible aircraft designs. This also demonstrates the generality and effectiveness of *Spapper* for spatial grammar rule development and application. When testing the developed grammar with a set of constraints stemming from the P-38 case study, for example the number of propellers or wing geometries, it is not only able to repeat the design synthesis conducted by Kelly Johnson, but also to generate a variety of other configurations. Among these solutions are several designs with a flying representative, meaning that these designs have already proven to be airworthy. This shows that the grammar is able to generate feasible aircraft concept designs, which suggests that there are more feasible solutions among the created novel designs that have not yet been produced. The grammar was not developed to generate final and production-ready designs but to computationally create aircraft configurations and to explore a given design space. It allows for the automatic creation of a high number of candidate solutions without a bias towards known solutions as is common with human designers [3]. Hence, using the grammar, the design synthesis process can not only be formalized and automated but novel yet feasible configurations can be generated, that can, for example, help a designer in breaking existing mental bias.

The outcome of the rule application process highly depends on the information given in the initial set of shapes where some of the constraints are defined in the form of labels or the parameters of other geometric objects. Therefore, with some limitations, the set of rules can also be used to create configurations with different requirements than the P-38, i.e. a different number of engines, wings or alike. Fulfilling these given requirements with the use of shape grammars is rather challenging as an aircraft configuration highly depends on certain conditions and restrictions and is not just a combination of shapes. *Spapper* supported this with its ability to define free parameter ranges and mathematical equations for setting conditional parameters or implementing symmetry. Further, the use of three-dimensional labels not only for spatial orientation but as a state variable, i.e. for controlling the number of a certain rule's application, was necessary.

Considering the results of the P-38 case study, the created designs do not just illustrate the aircraft's configuration and shape. As *Spapper* is embedded in a CAD environment, it is also possible to obtain quantitative data. The extractable data include volume, area or lengths of parts or the whole configuration. In the early design stage such values are rough estimations only but are sufficient to rule out infeasible designs, not only because of their configuration but also based on calculated parameters. Given that objective values to rate the configurations can be found and constraints like in Table 1 (e.g. number of engines, wing geometry etc.) can be defined in the initial shape, CDS could be embedded in an iterative optimization process in future projects [26].

Conclusion

The developed spatial grammar for creating aircraft configurations proved its ability to create feasible designs in the historic P-38 case study and showed the generality and effectiveness of *Spapper* as a spatial grammar interpreter. It is not only capable of recreating the historic conceptual design candidates but generated even more designs that fit the requirements for this particular case study. Some of the additionally discovered configurations have never been built so far and may be competitive design candidates. In this stage, the grammar could help aircraft designers to think outside-the-box and explore the design space by presenting alternatives that differ from the common baseline design. The spatial grammar interpreter *Spapper* in *FreeCAD* offers potential for further development and integration with other tools like using quantitative data from the generated designs to evaluate them and consequently embedding the rule-set in an iterative optimization process.

References

1. Leonard J (2001) Twin-engine propeller-driven aircraft configurations. Gateway News November 2001, American Institute of Aeronautics and Astronautics, St. Louis Section
2. Raymer DP (1992) Aircraft design: a conceptual approach, 2nd edn. American Institute of Aeronautics and Astronautics, Reston
3. Youmans RJ, Arciszewski T (2012) Design fixation: a cloak of many colors. In: Gero JS (ed) Design computing and cognition '12 (to appear)
4. Lange RH (1988) Review of unconventional aircraft design concepts. J Aircr 25:385–392. doi:10.2514/3.45592
5. Kroo I (2004) Innovations in aeronautics. 42nd AIAA Aerospace Sciences Meeting and Exhibit. doi: 10.2514/6.2004-1
6. Brandt SA, Whitford R, Stiles RJ, Bertin JJ (2004) Introduction to aeronautics: a design perspective, 2nd edn. AIAA, Cranfield Institute of Technology, Reston, Virgina
7. Stiny G, Gips J (1972) Shape grammars and the generative specification of painting and sculpture. In: Freiman CV (ed) Information processing 71. North Holland Publishing Co, Amsterdam, pp 1460–1465
8. Gips J (1999) Computer implementation of shape grammars. NSF/MIT Workshop on Shape Computation
9. McKay A, Chase S, Shea K, Chau HH (2012) Spatial grammar implementation: from theory to useable software. AI EDAM 26:143–159. doi:10.1017/S0890060412000042

10. Chakrabarti A, Shea K, Stone R et al (2011) Computer-based design synthesis research: an overview. J Comput Inf Sci Eng 11:021003–021003. doi:10.1115/1.3593409
11. Kalogerakis E, Chaudhuri S, Koller D, Koltun V (2012) A probabilistic model for component-based shape synthesis. ACM Trans Graph 31(55):1–55. doi:10.1145/2185520.2185551
12. Heisserman J, Mattikalli R, Callahan S (2004) A grammatical approach to design generation and its application to aircraft systems. In Proceedings of Generative CAD Systems Symposium. pp 12–14
13. Arnold P, Rudolph S (2012) Bridging the gap between product design and product manufacturing by means of graph-based design languages. In Proceedings 9th Tools and Methods of Competitive Engineering
14. Helms B (2013) Object-oriented graph grammars for computational design synthesis. Technische Universität München, http://mediatum.ub.tum.de/node?id=1113665
15. Stanford B, Beran P, Kobayashi M (2012) Aeroelastic optimization of flapping wing venation: a cellular division approach. AIAA J 50:938–951. doi:10.2514/1.J051443
16. Stavrev V (2011) A shape grammar for space architecture – a shape grammar for space architecture – part II. 3D graph grammar approach. 41st International Conference on Environmental Systems. doi: 10.2514/6.2011-5042
17. Hoisl F (2012) Visual, interactive 3D spatial grammars in CAD for computational design synthesis. Universitätsbibliothek der TU München, http://mediatum.ub.tum.de/node?id=1096886
18. Hoisl F, Shea K (2013) Three-dimensional labels: a unified approach to labels for a general spatial grammar interpreter. Artif Intell Eng Des Anal Manuf 27(4):359–375. doi:10.1017/S0890060413000188
19. Hoisl F, Shea K (2011) An interactive, visual approach to developing and applying parametric three-dimensional spatial grammars. Artif Intell Eng Des Anal Manuf 25:333–356
20. Agarwal M, Cagan J (1996) A blend of different tastes: the language of coffee makers. Environ Plan B Plan Des 25(2):205–226
21. McCormack JP, Cagan J, Vogel CM (2004) Speaking the Buick language: capturing, understanding, and exploring brand identity with shape grammars. Des Stud 25:1–29
22. Callaway T (2012) Aviation classics 14 – lockheed P38 lightning. Mortons Media Group, Horncastle
23. US Air Force (1943) Lockheed YP-38. In: US Air Force Photos. http://www.af.mil/News/Photos.aspx?igphoto=2000595321. Accessed 1 Dec 2013
24. Chambers JR (2005) Innovation in flight: research of the NASA Langley Research Center on revolutionary advanced concepts for aeronautics. National Aeronautics and Space Administration, Washington, DC
25. Paur J (2011) Burt Rutan's boomerang: safety through asymmetry. In: Wired AUTOPIA. http://www.wired.com/autopia/2011/07/burt-rutans-boomerang-safety-through-asymmetry/. Accessed 15 Sep 2013
26. Shea K, Aish R, Gourtovaia M (2005) Towards integrated performance-driven generative design tools. Autom Constr 14:253–264. doi:10.1016/j.autcon.2004.07.002

Spectral (Re)construction of Urban Street Networks: Generative Design Using Global Information from Structure

Somwrita Sarkar

Abstract Modeling and analysis of urban form is typically performed using local generative design techniques, (e.g., shape grammars), with closed sets of local rules operating on elements. While this approach is powerful, the open variety of possible non-unique choices over the element and rule sets does not answer an important *closure* question: How much information, i.e., how many elements and rules, exhaustively capture all the information on structure? This paper investigates the inverted principle: *using global system information to reconstruct a design*. We show that orthogonal eigenmodes of a street network's adjacency matrix capture global system information, and can be used to *exactly* reconstruct these networks. Further, by randomly perturbing the eigenmodes, new street networks of similar typology are generated. Thus, eigenmodes are global generators of structure. Outcomes provide new mechanisms for measuring and describing typology, morphology, and urban structure, and new future directions for generative design using global system information.

Introduction

As long as cities have existed, we have tried to understand the principles behind urban structure and form, in the hope that we will be able to design cities better. Cities are very special examples of complex systems that are *both* self-organized and designed at the same time [1]. It is insufficient to understand urban structure using physical design principles alone, because a large part of the physical structure is generated by the action of multiple, distributed, socio-economic processes that act simultaneously with centralized planning and design [2, 3]. Spatial structure implicitly encodes both the actions of planners and designers, as well as the results of the distributed socio-economic processes. Thus, it is important to develop a deep

S. Sarkar (✉)
University of Sydney, Sydney, Australia
e-mail: somwrita@gmail.com

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_3

understanding of the physical layer as an encoder of all the other layers, demographic, socio-economic, and political.

The formal study of city structure and street networks has historically being amongst the richest areas of inquiry in urban design and planning. Traditional approaches have been largely qualitative, and have derived from historical and cultural studies [4–6]. The understanding of formal "type" has been defined on the basis of design movements, specific stylistic contributions of designers, cultural or art specific eras, socio-political forces, and geographic factors. Other prevalent approaches have been based on quantitative modeling. From Christopher Alexander's Pattern Language [7] and comparison of tree and semi-lattice structures [8], to the Space Syntax approach [9], to work on modeling the fractal structure of cities using cellular automata and agent based models [1, 10, 11], to using shape grammars to model urban typologies and styles [12–14], to traditional approaches in quantitative and economic geography, economics, and transportation research [15], all quantitative modeling approaches attempt to derive pattern based, semi-mathematical, or mathematical first-principles based ways to characterize urban structure.

Much of the preceding research described above is about generative design: a set of elements and a set of "growth" or "modification" rules to operate on these elements are identified, abstracted, and defined, starting from an original design. The resulting corpus of designs or structures is produced by repeated application of the rules to the elements. The microstructures generate the macrostructure, the local generates the global, and higher-level stylistic or typology-based forms are emergent. The basic idea behind generative design is that if we are able to encode and generate a particular typology of urban form by abstract entities and operations, then, then this process can help us to understand the "deep structure" and design accordingly.

While these efforts have been quite powerful, they are, by definition, not exhaustive or complete. That is, once a set of elements and rules is defined, whether the sets are deterministically or stochastically defined, the solution space of the designs that can be generated may be very large, but in principle, countable. However, it is impossible to prove that a defined set of elements and rules will capture all the structural characteristics of a given system completely. Secondly, there is no unique way of defining elements and rules; the same system can in principle be described by starting from a different set of elements and rules.

Aims

The aim of this paper is to address the above and explore the possibility of *generative design using global system information*, *where the generation is encoded by a unique set of entities and operations that*, *as global generators*, *exhaustively and completely capture all the information about structure*. We use a specific network representation of modeling city structure and employ network spectra (eigenvalues and the corresponding set of orthogonal eigenvectors) to extract information from this structure that allows us to generatively reconstruct the structure.

Background

Recently, there has been a surge in research modeling the spatial structure of cities as networks [1–3, 16–18]. This research typically models cities, and more specifically street networks, as complex physical systems represented as graphs, and has shown that graph representations can provide analytical insights into city structure [16, 18, 19], processes that govern its growth [3], patterns of such growth [2], and how social, economic, or transport processes interact with spatial structure [17]. In this paper, we use the network perspective to bring out a new approach, unexplored in [2, 16–18]: *that network modeling, and in particular spectral approaches, can be used not just for analytical purposes, but also as a generative design and reconstruction mechanism to better understand topology and morphology of cities.*

Previous research has looked at using spectral methods to better understand the questions of how different cities may be classified based on their structural characteristics [20, 21], and to generate plan forms in architectural plan design using space syntax approaches [22]. However, there are important differences of approach in how spectral information is used in this paper as compared to previous work. First, previous research has typically adopted a pattern recognition and classification perspective demonstrating the use of spectra for analytical purposes, rather than a generative design perspective [20, 21]. Second, when used for design generation of architectural plan forms, previous work has typically focused on the use of only the largest (positive) few eigenmodes (eigenvectors and eigenvalues) or principal components for generation [22], while in this paper we look at the full spectrum (including positive and negative eigenvalues) and corresponding eigenvector bases for using global structural information to reconstruct urban street networks. Third, previous research has looked at standard approaches to generate new plans: start from a given network and modify this network with local changes to network structure under optimality conditions (say using genetic algorithms [22]) to ensure that the spectral deviation of the new network from the old one is minimized. In this paper, we look at directly perturbing the eigenmodes (the eigenvalues and eigenvector basis) to generate similar networks matching a given network, which is a more direct mechanism of generation.

Significance

We start with an analogy from the physics of waveforms. In Fourier Analysis, any wave can be described by an infinite series of pure sine and cosine components. Starting from a complex waveform, these pure components are derived with each component contributing to the waveform depending on the magnitude of the coefficients that define “how much” of each pure component exists in the complex wave. What is special about this representation is that the components are *orthogonal*; i.e., they are uncorrelated and independent of each other, and therefore they can be used in a combinatorial manner to reconstruct not just the original wave, but other approximations of the wave in lower dimensions.

Exactly the same idea is employed in this paper in making use of graph spectra to understand the structure of networks. The eigenvalue decomposition of a network shares, in discrete graph or matrix space, this property of *orthogonality*. Spectral methods have a rich history of being used to understand a host of global structural features about graphs and matrices, such as graph partitioning [23], modularity [23–25], hierarchical modularity [24–26], system decomposition and design reformulations [27, 28], and properties such as dynamics of information flow or synchronization in networks [29], to name just a few.

In this paper, we use the spectrum to explore the following specific questions: Given a full set of eigenmodes of a system, how many are needed to *exactly* reconstruct the system in question [30]? Further, can we relax the exact reconstruction idea to generate a similar family of plans starting from the adjacency matrix of an original matrix, using the orthogonal or perturbed orthogonal components in a combinatorial manner to generate different structures, which are similar to the original structure?

If we are able to do this, then the following important insights about how design knowledge is encoded in the structure can be drawn:

1. *Design knowledge is "distributed" in the global structure of the design.*
   Much of qualitative research based on historical, cultural, and demographic perspectives [4–6], tells us that for self-organized systems such as cities, design knowledge is encoded in a "distributed" manner in the structure of a city. That is, design knowledge results from "collective memory" or shared memories and common understanding of culture, history, and traditions. Such an understanding of the distributed nature of design knowledge sits in contrast to the traditional AI perspective of knowledge encoding by symbol and grammar definition and subsequent generation. When we abstract elementary symbols, shapes, or entities, and define elementary rules to generate structure, the nature of this distributed-ness cannot be fully captured: an elementary entity or shape sitting in relationship with other entities or shapes takes on new meanings defined by the context in which it sits embedded, in interaction with other entities [28]. This is the idea that a network representation and spectral information about the network can capture, in a global, distributed sense.
2. *Design generation can be automated using such "distributed" knowledge*
   We show in this paper that it is possible to automate design reconstruction using such distributed knowledge of structure. Instead of a collection of local generators, the spectral information provides global generators for a system. Very importantly, we find that the generation process occurs in a spatially distributed manner, rather than starting from a centralized growth root.
3. *Design information can be compressed in a much lower number of dimensions than the number of total dimensions in which the design is expressed.*
   We show in this paper that usually, a much lower number of dimensions than the number of nodes and links in a network can capture the full structure of the

network. Thus, along with encoding design knowledge and design generation, spectral information can also be used to compress design information. In fact similar mechanisms for information compression are regularly used in many other domains such as image and video compression.

Methods

Data Representation

A street network is represented as a graph $G = (V,E)$, where the graph G is defined by the two sets V and E. The set V is a set of n *nodes* or *vertices*, and the set E is a set of m unordered pairs drawn from V, each pair representing a *link* or an *edge*. That is, the set V has n nodes, $1 \le i \le n$, and the set E has m edges, $1 \le i \le m$. This graph G is represented in matrix form by an *adjacency matrix* A, whose rows and columns represent the n nodes, and matrix entry $A_{ij} = 1$ if there is a link between nodes i and j, and $A_{ij} = 0$ otherwise. In a weighted graph, if a link exists between the nodes i and j the entry A_{ij} represents the length of the link between nodes i and j. That is, $\mathrm{A}_{ij} = d_{ij}$, where d_{ij} is the metric distance between the nodes and measures the length of the link. Further, these street networks are undirected; i.e., the edges do not have a direction. Thus, the adjacency matrix A is symmetric, with $A_{ij} = A_{ij}$ This representation of a street network is referred to as the primal graph representation in research literature (Barthelemy [18]; Porta et al. [16, 17]), as opposed to the dual graph, where the streets are nodes, and share a link if they share an intersection. In this paper we focus only on the topological connectivity, i.e., unweighted binary matrices, for the primary reason that it is important to understand topological and geometrical effects in network structure separately. In this first paper on spectral reconstruction of urban street networks, we focus on the topology aspects. Future work will focus on geometry, i.e., the lengths of connections between nodes, angles between streets, etc. We note, however, that the spatial location of nodes (the actual latitude longitude positions) has been considered in the work, and not neglected. The networks considered are primarily spatial networks.

Spectral Reconstruction of Networks

The eigenvalue decomposition (EVD) of a network adjacency matrix is defined as follows:

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^{\boldsymbol{T}},$$

where $\boldsymbol{V}$ is an orthogonal matrix with

$$\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{V} = \boldsymbol{V}\boldsymbol{V}^{\boldsymbol{T}} = \boldsymbol{I},$$

where $\boldsymbol{I}$ is the identity matrix, and the columns of $\boldsymbol{V}$ are the normalized orthogonal set of eigenvectors, corresponding to eigenvalues

$$\boldsymbol{\lambda}_1 \geq \boldsymbol{\lambda}_2 \geq \cdots \boldsymbol{\lambda}_{n-1} \geq \boldsymbol{\lambda}_n,$$

arranged in descending order, expressed in the $\boldsymbol{n} \times \boldsymbol{n}$ diagonal matrix $\boldsymbol{D}$. The eigenvalues are called the *spectrum* of a network, and there is an eigenvector corresponding to each eigenvalue. The main point in this decomposition is the idea of *orthogonality*: the decomposition transforms the dependent correlated rows and columns of $\boldsymbol{A}$ into a set of perpendicular, uncorrelated, independent eigenvectors. The corresponding eigenvalues then capture by how much an eigenvector will "shrink" or "stretch" when transformed by $\boldsymbol{A}$. In simple words, the magnitudes of the eigenvalues capture the relative importance of the corresponding eigenvectors.

In this paper, using the result in [30], we investigate the specific question: what is the least number of eigenvalues and eigenvectors we can use to reconstruct the given urban network exactly? In other words, what is the maximum number of eigenvalues and eigenvectors we can discard as containing irrelevant or redundant information?

There are many ways to decide which eigenvalues and eigenvectors to preserve. For example, previous research has looked at the information contained in the highest magnitude positive eigenvalues and corresponding eigenvectors only [21]. Standard approaches such as Principal Component Analysis also work in this way, preserving the largest eigenvalues and corresponding eigenvectors. However, since the trace of the adjacency matrix is 0 (i.e., sum of diagonal entries), on average, the adjacency matrix will have equal numbers of positive and negative eigenvalues. Therefore, negative eigenvalues would appear to play an equally important role in encoding information as the positive ones. Therefore, we reorder the eigenvalues based on their absolute values, as follows:

$$|\lambda_{(1)}| \leq |\lambda_{(2)}| \leq \cdots \leq |\lambda_n|,$$

such that $\lambda_{(j)}$ is the j^{th} smallest eigenvalue, in absolute value, with corresponding eigenvector Vj. Now, setting the smallest j eigenvalues to 0, we have a new diagonal matrix

$$D' = diag\big(0, 0, \ldots, 0, \lambda_{(j+1)}, \lambda_{(j+2)}, \ldots, \lambda_{(n)}\big).$$

The eigenvectors are correspondingly reordered to produce a new matrix of eigenvectors V'. Now we define the new matrix

$$B = V'D'^{T},$$

where we use these reordered matrices to produce an approximation of the original adjacency matrix A by omitting the j smallest eigenvalues (in absolute value) to reconstruct the matrix.

The approximated matrix B will have all real entries b_{ij}. Thus, using the Heaviside function defined in [30], we threshold the real entries in B to produce a binary reconstructed matrix C, with entries c_{ij} as follows:

$$c_{ij} = \begin{cases} 1, & if \;\; b_{ij} \geq 0.5 \\ 0, & if \;\; b_{ij} < 0.5 \end{cases}$$

We note here, however, that the numerical choice of this threshold for design purposes can also be left as a parameter. Lowering the threshold will usually generate denser networks and lower numbers of eigenvalues and eigenvectors will be needed to reconstruct the network, while a higher threshold will generate sparser networks, and a higher number of eigenvalues and eigenvectors to reconstruct the network.

The operations described above provide us with binary adjacency matrices that are reconstructed starting from the original adjacency matrices, that we have produced by considering the largest absolute value eigenvalues and corresponding eigenvectors as *generators*. The principal idea in this reconstruction is that, as generators, the eigenvalues and eigenvectors capture global information about form of the network, as compared to local motif based generators. Figure 1 shows the reconstruction of a lattice network, and Fig. 2 shows the reconstruction of a binary tree network. Note, in both cases, that because of the regular structure of the networks, the reconstruction occurs in a more or less regular manner. In the next section, we will apply the same process to real city networks that are not as regular as perfect lattices and trees.

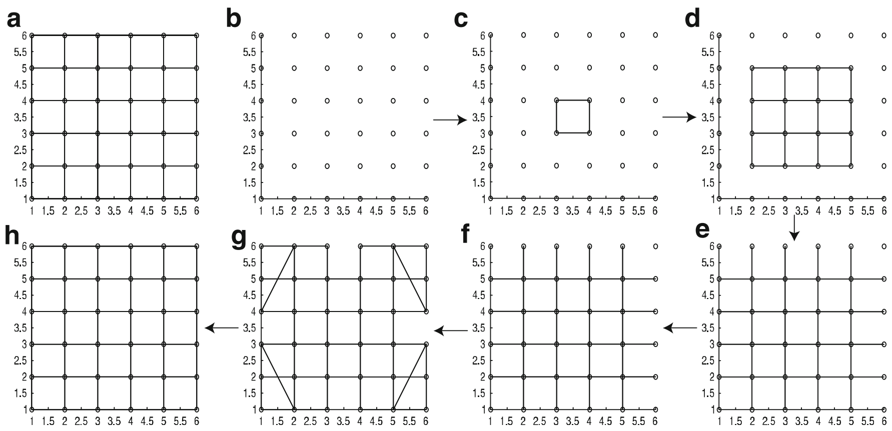


Fig. 1 Spectral reconstruction of lattice networks: (**a**) Original 6 by 6 lattice network, (**b**)–(**h**) networks reconstructed using 2, 4, 6, 8, 10, 12, and 14 largest magnitude eigenvalues and corresponding eigenvectors. The figure shows that out of the 36 modes, only 14 largest modes are needed to reconstruct the original network *exactly*

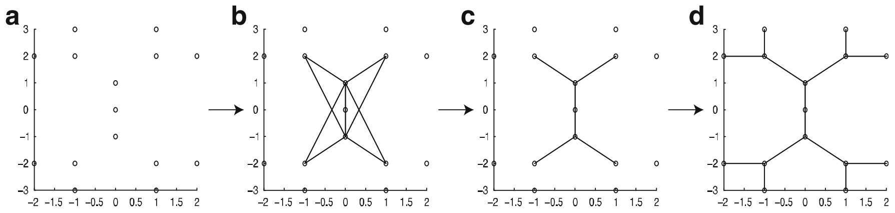


Fig. 2 Spectral reconstruction of tree networks: (**a**)–(**d**) Networks reconstructed using 2, 4, and 6 largest magnitude eigenvalues and corresponding eigenvectors, with (**d**) also showing original network. The figure shows that out of the 15 modes, only 6 largest modes are needed to reconstruct the original network *exactly*

Results

In this section, we first apply the spectral reconstruction technique to real city networks. Further, we also perturb the eigenvector basis randomly to generate networks of similar types as a given original network, therefore showing that small perturbations to the spectra can produce networks of similar typologies.

We have applied the above to a range of different local neighborhoods (for example, exploring the differences between differences neighborhoods in the same city [19]), however, for brevity, we have chosen to present two extreme examples here: (a) the CBD of Sydney, which is understood to be a planned city, and represents a deformed lattice-like network and (b) the historic precinct of Tajgunj, behind the Taj Mahal in Agra, India. The interesting aspect about Tajgunj is that it started as a perfect lattice, a planned settlement, at the time of the construction of the Taj Mahal. Over time, however, self-organization has taken over, and the structure has evolved into tree-like structures growing inside the original lattice-like planned settlement.

Spectral Re-construction of City Networks

Figure 3 shows the spectral reconstruction of the Sydney City network. By preserving different numbers of eigenvalues and eigenvectors, the evolution of the generation is shown. The network has 257 nodes. Thus, we have 257 original eigenmodes. However, we found that only 68 of these were enough to exactly reconstruct the original network. Note the spatially distributed nature of the generation. Also note that at each step, the reconstruction or generation process brings out network forms that share the topological and typological qualities of the original network.

Figure 4 shows the spectral reconstruction of the Tajgunj network. Again, we find that only 50 of the total original 173 eigenmodes are needed to exactly capture the full reconstruction of the original network.

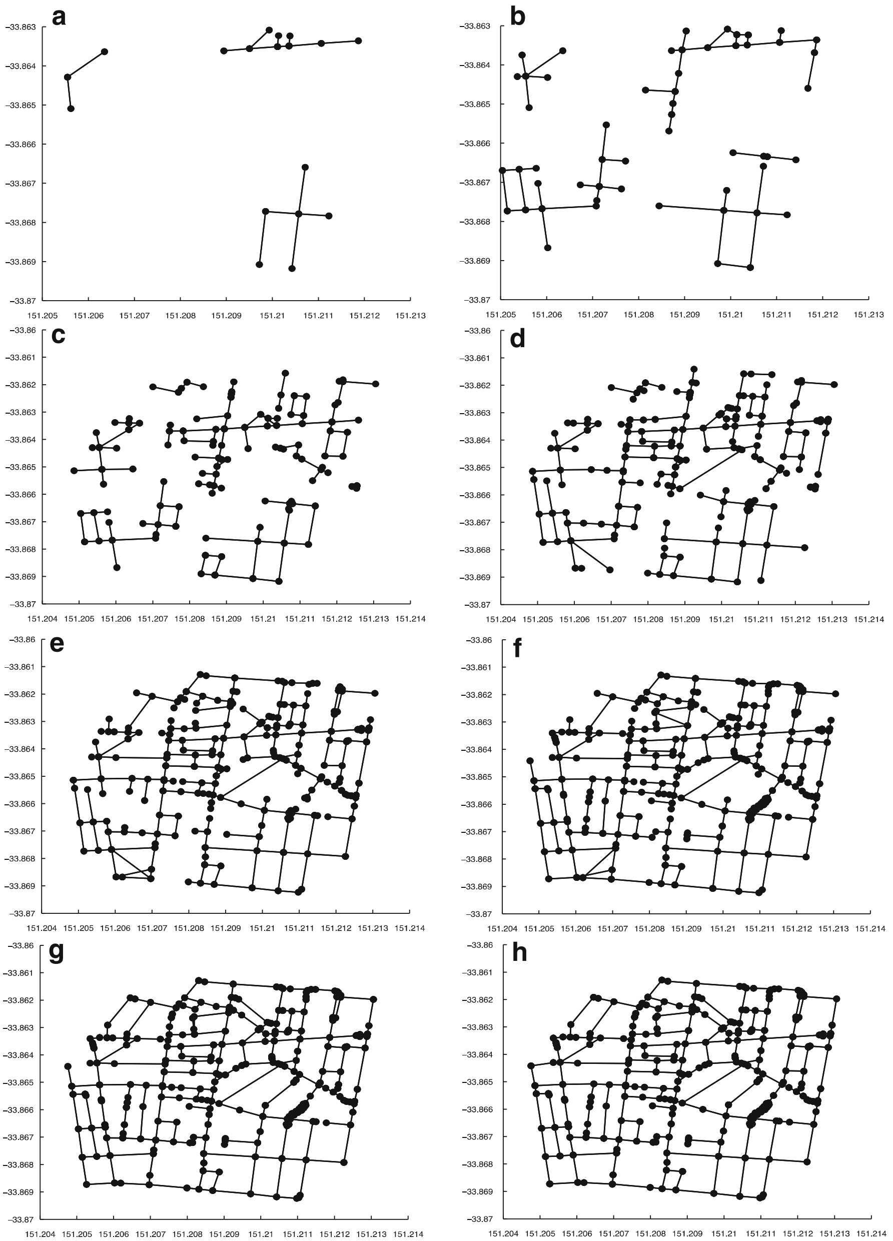


Fig. 3 Spectral reconstruction of Sydney CBD network: (**a**)–(**h**) Networks reconstructed using 5, 10, 20, 30, 40, 50, 60 and 68 largest magnitude eigenvalues and corresponding eigenvectors, with (**h**) also showing original network. The figure shows that out of the 257 modes, only 68 largest modes are needed to reconstruct the original network *exactly*

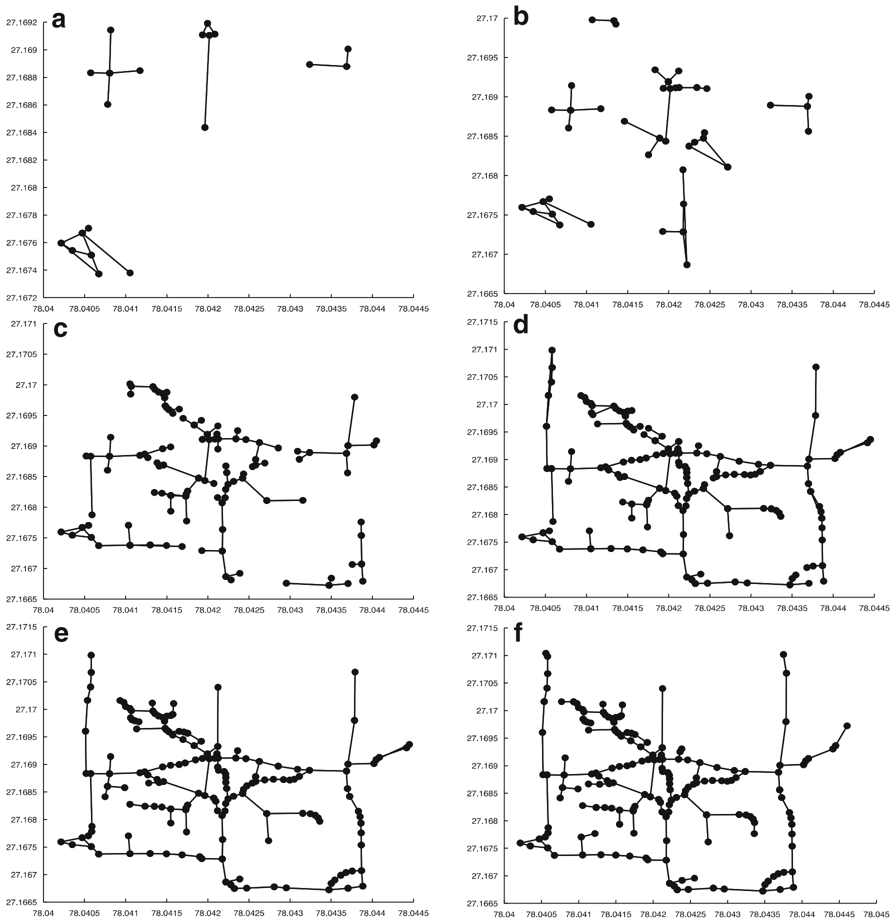


Fig. 4 Spectral reconstruction of Tajgunj, TajMahal, Agra network: (**a**)–(**f**) Networks reconstructed using 5, 10, 20, 30, 40, and 50 largest magnitude eigenvalues and corresponding eigenvectors, with (**f**) also showing original network. The figure shows that out of the 173 modes, only 50 largest modes are needed to reconstruct the original network *exactly*

Generating Similar Networks Using Perturbations

In the previous step, we demonstrated exact reconstruction of the original network. We also wanted to explore the idea of generating similar types of networks like the original network. When we mean similar networks, we mean that the spectral, as well as other local properties such as degree distributions, number of links and link density, number of cycles and cycle density, etc. to be similar to the original network.

We perturbed the eigenvector basis as follows. From the previous section, we know that the eigenvectors form the matrix V, corresponding to eigenvalues in the diagonal matrix D. We now define a random matrix R, with a defined percentage of the entries set to 1 and the rest set to 0. This percentage is a parameter that can be varied to produce different types of networks. We now define

$$V' = V + \in R,$$

where ε is any small number varying from 0 to 1. Now, the matrix V' is used to recompute

$$A' = V'DV'^{T}.$$

Now, A' represents a new network derived from the eigenvector basis of the old network. Figures 5 and 6 show the reconstructed Sydney-like and Agra-like networks generated using 30 % random perturbation, with the value of ε set to 0.1. Note that new links have appeared, and some of the old links have disappeared, but in general, the overall structure and "type" stays the same.

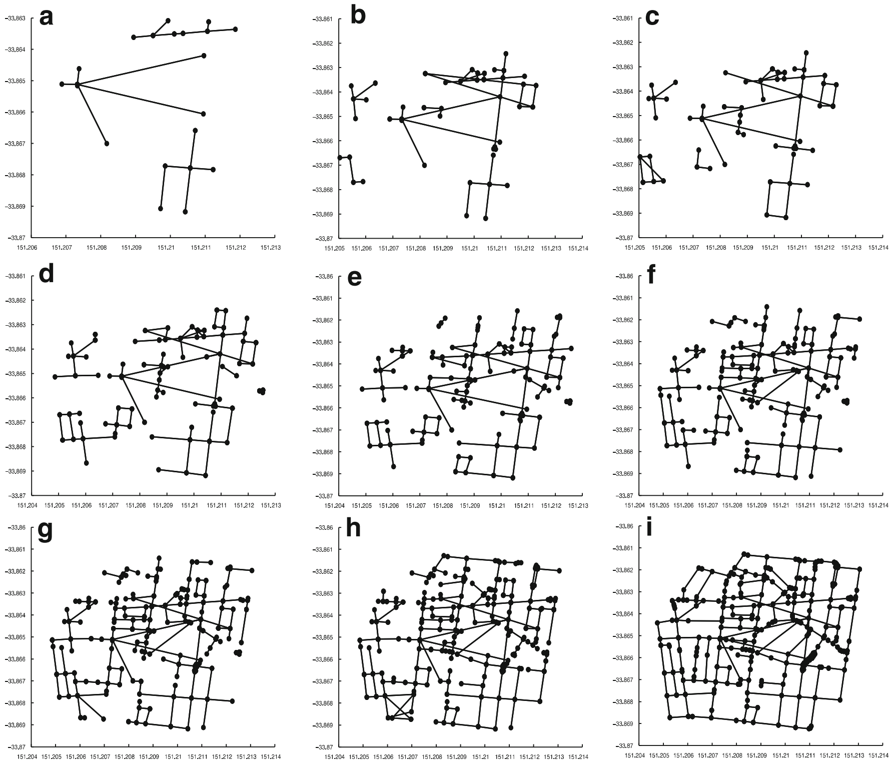


Fig. 5 Spectral reconstruction of Sydney-like networks by eigenvector perturbation by using (**a**)–(**i**): 5, 8, 10, 15, 20, 25, 30, 35, 40 largest magnitude eigenvalues and their corresponding perturbed eigenvectors

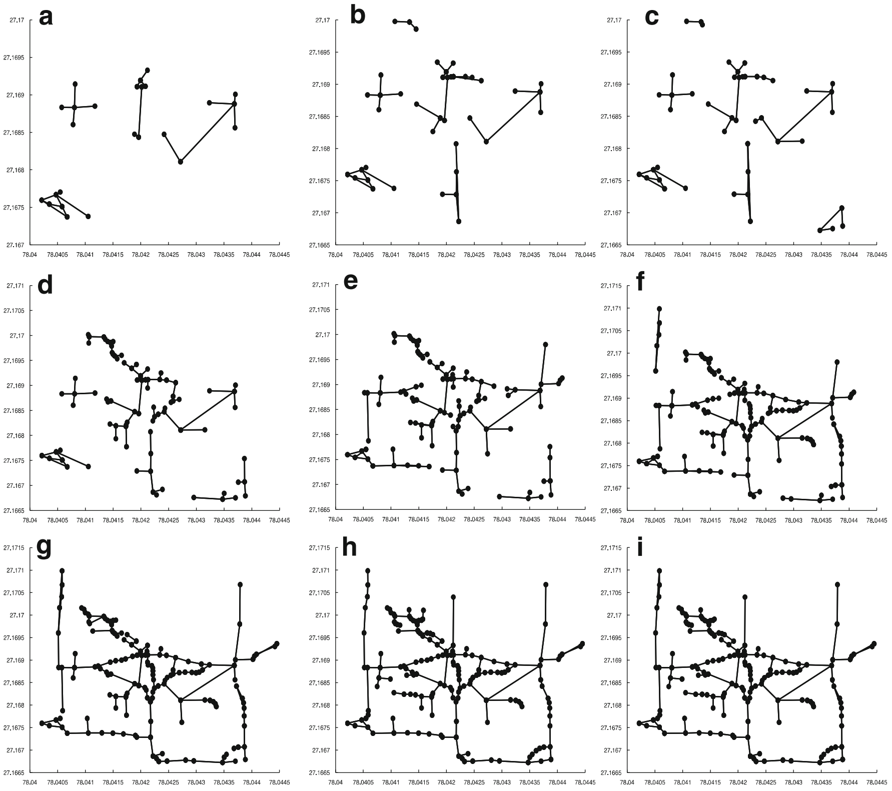


Fig. 6 Spectral reconstruction of Tajgunj-like networks by eigenvector perturbation by using (**a**)–(**i**): 5, 8, 10, 15, 20, 25, 30, 35, 40 largest magnitude eigenvalues and their corresponding perturbed eigenvectors

We also noticed, empirically, that by perturbing the eigenvector basis with about 20 % random perturbation, the original network was still reconstructed perfectly. This shows that the eigenvectors are robust to perturbations, such that a perturbation larger than a certain threshold is needed to corrupt the information contained in them.

Note also that the columns of V' won't remain orthogonal after the perturbation is introduced, so A' will have different eigenvalue decomposition than A. We recompute this as $A' = X\Lambda X^T$ and then compare the spectrum of the new reconstructed network to the spectrum of the original one. Figure 7 shows the eigenvalue spectra of the original and perturbed networks superimposed for Sydney. Note that the deviation is very slight, ensuring that the new networks are very similar to the original network.

A current limitation of the method in this preliminary version is that we have only considered topology, and have therefore, not imposed any geometric or planarity constraints in the generation of the new networks. The perturbations, therefore, have been kept to a minimum, since increasing the perturbations seem to result in non-planar new links being introduced that violate the typical

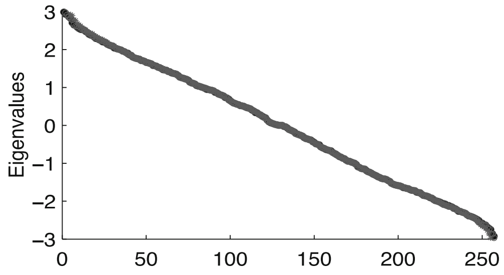


Fig. 7 Comparing the spectrum of the original Sydney city network with spectra of Sydney-like networks produced by perturbation; the black line represents the eigenvalues of the original network, and the grey lines are eigenvalues of the perturbed networks produced using 25–30 % random perturbations introduced into the eigenvector basis

"typology". However, in future versions of the work, this can be easily remedied by explicitly considering geometric and planarity information (for example, length distributions of links, angles between streets [20, 21]: this would simply imply that we run the same method but with a weighted adjacency matrix that incorporates geometric information, such as the positions of nodes, lengths of links, angles between streets, and other such local properties, rather than the binary topology based adjacency matrix. Perturbations will then include perturbations over all these properties to generate new networks with similar typologies.

Conclusions

This paper presented an approach for generative design using global system information, where the generation is encoded by a uniquely defined set of spectral generators that exhaustively and completely capture all the information about the structure of the system. Primal graph representations of urban street networks and their spectra (eigenvalues and the corresponding set of orthogonal eigenvectors) were used to extract information from this structure that allows us to generatively reconstruct the structure. Using the method, we demonstrated the reconstruction of two very different city types and neighborhoods. By perturbing the eigenvector basis, we also generated networks that share similar structural and typological properties as the original networks. A typical property of the generation is that it occurs in a spatially distributed manner, rather than starting from a central root and spreading out in space.

We discussed principal differences between this approach and other locally generative design approaches that employ sets of entities and operations to generate global design structure using local information about structure. We found that global information about design can be encoded in a distributed manner, and can also be used to automate the reconstruction of the design. Further, we found that this idea can also be used as a design information compression idea, since a typically, a lower number of spectral dimensions can be used to reconstruct a network exactly.

We also note here an important point about quantification of the *global information content* of a particular design. Our results show that in general, the greater the order in the network (e.g., lattice-like networks), the greater the symmetry in the eigenmodes, and lesser or more regular will be eigen-information content needed to characterize a design. In contrast, the more disordered the network (e.g., Tajgunj), the greater the asymmetry in the eigenmodes, and more the amount of information content needed to characterize the design. It would be very interesting to bring this into the information theory/entropy computation area to test the hypothesis: How does the entropy or information content needed to characterize a design vary with the specific structural characteristics of regularity or irregularity in the design? For example, will planned and unplanned cities show different informational principles of organization? Answers to these questions will need the development and computation of such metrics that convert the eigen-information to an entropy/information content measurement. We will pursue this in our future work. As a consequence, along with, and separate from, network based metrics such as centrality or betweenness, it will become possible to have quantifiable metrics to look at classifications of city structures, typology classification, and quantifiable morphological analysis (by tracing how eigen-information, entropy, or information content changes as a city structure evolves).

The physical planning of a city, and urban design, are intertwined with self-organization that occurs over time. Thus, for any planning or design intervention, it is essential to deeply understand "what is" before we can begin to define "what will be", or the set of actions to change what is. Tracking changes to global and local network properties as defined and measured by changes to the spectra, in conjunction with changes to any of the local properties such as degree distributions, clustering, centralities of links and nodes, and link and cycle densities, can help us to understand whether the changes we are proposing to the structure of the network will drastically alter global network properties and therefore the typology of the structure.

Thus, in future work, we will analyze how the approach presented in this paper can be linked to local network properties such as mesh-like or lattice-like and tree-like behavior, degrees, clustering, centrality metrics, and link and cycle densities. By doing this, it will be possible to define new metrics for measuring "typology" of urban structure in more quantitative ways, and understanding and measuring the effects of any proposed design interventions. Further, by tracking changes in spectral properties and local properties of city structure networks changing over time, it will also be possible to understand morphology of urban structure in new quantitative ways.

References

1. Batty M (2013) The new science of cities. MIT Press, Cambridge, MA
2. Barthelemy M et al (2013) Self-organization versus top-down planning in the evolution of a city. Nat Sci Rep 3:Article no. 2153

3. Strano E et al (2013) Elementary processes governing the evolution of road networks. Nat Sci Rep 2:Article no. 296
4. Kostof S (1999) The city shaped: urban patterns and meanings through history. Thames and Hudson, New York
5. Kostof S (2005) The city assembled: elements of urban form through history. Thames and Hudson, New York
6. Rossi A (1984) The architecture of the city. MIT Press, Cambridge, MA
7. Alexander C (1977) A pattern language: towns, buildings, construction. Oxford University Press, New York
8. Alexander C (1966) A city is not a tree. Design, London, Council of Industrial Design:No. 206
9. Hillier B, Hanson J (1984) The social logic of space. Cambridge University Press, Cambridge
10. Batty M (2007) Cities and complexity: understanding cities with cellular automata, agent based models, and fractals. MIT Press, Cambridge, MA
11. Batty M, Longley P (1994) Fractal cities: a geometry of form and function. Academic, San Diego
12. Stiny G (1980) Introduction to shape and shape grammars. Environ Plan B: Plan Des 7:343–351
13. Stiny G, Mitchell WJ (1978) The Palladian grammar. Environ Plan B: Plan Des 5:5–18
14. Duarte JP, Rocha JM, Soares GD (2007) Unveiling the structure of the Marrakech Medina: a shape grammar and an interpreter for generating urban form. Artif Intell Eng Des Anal Manuf 21:317–349
15. Haggett P, Chorley RJ (1969) Network analysis in geography. Edward Arnold, London
16. Porta S, Crucitti P, Latora V (2006) The network analysis of urban streets: a primal approach. Environ Plan B: Plan Des 33(5):705–725
17. Porta S et al (2009) Street centralities and densities of retail and services in Bologna. Environ Plan B: Plan Des 36:450–465
18. Barthelemy M (2011) Spatial networks. Phys Rep 499:1–101
19. Sarkar S (2013) Street network analysis for understanding typology in cities: case study on Sydney CBD and suburbs. In: State of Australian Cities (SOAC) Conference. Sydney, Australia
20. Hanna S (2012) A representational scheme for the extraction of urban genotypes. In: Gero JS (ed) Design computing and cognition. Springer
21. Hanna S (2012) Comparative analysis of neighborhoods using local graph spectra. In: Eighth International Space Syntax Symposium. Santiago, Chile
22. Hanna S (2007) Representation and generation of plans using graph spectra. In: Sixth International Space Syntax Symposium, Istanbul, Turkey
23. Newman MEJ (2010) Networks: an introduction. Oxford University Press, Oxford
24. Sarkar S, Dong A (2011) Community detection in graphs using singular value decomposition. Phys Rev E 83(4):046114
25. Sarkar S, Henderson JA, Robinson PA (2013) Spectral characterization of hierarchical network modularity and limits of modularity detection. PLoS One 8(1):e54383
26. Sarkar S et al (2013) Spectral characterization of hierarchical modularity in product architectures. J Mech Des 136(1), 011006
27. Sarkar S, Dong A, Gero JS (2009) Design optimization problem reformulation using singular value decomposition. J Mech Des 131(8), 081006
28. Sarkar S, Dong A, Gero JS (2009) Learning symbolic formulations in design: syntax, semantics, and knowledge reification. Artif Intell Eng Des Anal Manuf 24(1):63–85
29. Arenas A, Diaz-Guilera AA, Perez-Vicente CJ (2006) Synchronization reveals topological scales in complex networks. Phys Rev Lett 96(11):114102
30. Liu D, Wang H, Van Mieghem P (2010) Spectral perturbation and reconstructability of complex networks. Phys Rev E 81, 016101

A Spatiotemporal Mereotopology-Based Theory for Qualitative Description in Assembly Design and Sequence Planning

Elise Gruhier, Frédéric Demoly, Said Abboudi, and Samuel Gomes

Abstract This paper presents a novel qualitative theory in the context of assembly-oriented design, which integrates assembly sequence planning in the early product design stages. Based on a brief literature review of current assembly design approaches and mereotopology-based theories, the authors propose to go beyond by defining their own mereotopological theory, therefore enabling the qualitative description of product-process information and knowledge. The proposed mereotopological theory provides a strong basis for describing spatial entities (product parts) changes over time and space by considering a region-based theory linking spatial, temporal and spatiotemporal dimensions. The main objective of such an approach is to provide a product design description by proactively considering its assembly sequence as early as possible in the product development so as to ensure information and knowledge consistency with preliminary information and later introduce a spatiotemporal reasoning layer.

Introduction

The current competitive industrial context has raised needs in computational intelligence in design and manufacturing so as to ensure awareness and understanding of product architects and designers in the product design process [1]. The goal of delivering engineering models in line with the real world over time creates new research challenges, related to spatiotemporal description and representation, which actually cover many research areas such as philosophical investigations, mathematics, artificial intelligence and engineering to name a few. In general, product designers can be seen as actors with specific and sharper knowledge on a specific domain, where customer/user needs are generally

E. Gruhier • F. Demoly (✉) • S. Abboudi • S. Gomes
Université de Technologie de Belfort-Montbéliard, Montbéliard, France
e-mail: frederic.demoly@utbm.fr

© Springer International Publishing Switzerland 2015
J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_4

considered as the main constraints in the product design process. This lead to engineering product definitions mistakes and generally require revisions, due to its functional, geometrical, physical, etc. complexity and multiple temporal configurations, when the product information passes from design to manufacturing engineering and later to production, etc. This statement is not directly addressed to product architects and designers but highlights a lack of awareness and understanding of product lifecycle information and knowledge flows. A remaining challenge [1] consists in considering and integrating knowledge of products' lifecycle, such as knowledge from process planning, assembly planning, etc., at the products' earliest design stages.

As such, Demoly et al. [1] have described the current stakes in engineering design, which highlight emerging needs in proactive engineering based on qualitative description of lifecycle knowledge, The product-process engineering is currently changing from an informal approach, based on experience, to a science-based approach [2]. Moreover the dynamic aspect of the design process has not been yet taken into account in an appropriate manner [3]. Indeed the object evolves and changes over its design process. Previous engineering philosophies, such as sequential and concurrent engineering [4–6], have provided successful results with the support of quantitative data and heuristic rules respectively, but do not cover current Product Lifecycle Management (PLM) requirements. In such a context, everything needs to be under control, interpretable and understandable, especially information and knowledge flows.

Based on these research stakes, new research efforts have to be addressed on formalism and theory in engineering design in order to represent product and lifecycle knowledge in a qualitative and machine-interpretable manner [2]. Here, the major objective is to propose and describe a region-based theory called mereotopology for assembly-oriented design process. With such a theory, the formal representation of the relational information of the product with its parts and its lifecycle operations (i.e., here assembly operations) at various abstraction levels, becomes vital in the product design process (Fig. 1). Indeed, product design

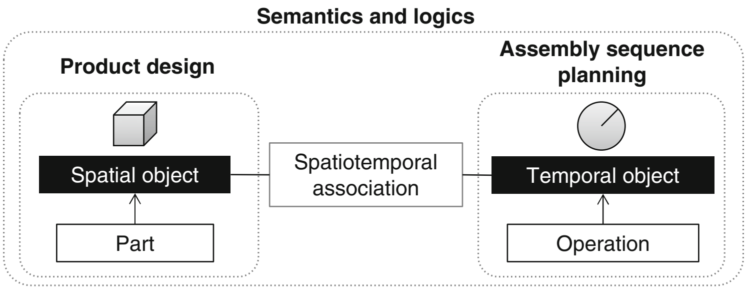


Fig. 1 Understanding product-process integration with semantics and logics

is composed of spatial objects (such as parts) and assembly sequence planning is composed of temporal objects (such as assembly operations). Here, proactive design tackles current engineering issues related to the integration of product's lifecycle constraints and knowledge by considering lifecycle sequence planning issues as early as possible in product design. With such a point of view, product architects and designers have a role to play within the following dimensions: spatial, temporal and spatiotemporal (four-dimensionalism) [7].

Firstly, the paper presents, in section "Literature Review", a brief literature review on assembly design approaches and mereotopology-based theory domains. Then section "Mereotopological Design of Assembly Design Evolution" describes the proposed theory, which provides product-process associations through semantics and logics. Section "Mereotopological Design of Assembly Design Evolution" introduces a mechanical assembly to illustrate the relevance of the proposed theory. In section "Discussions" the limits and the advantages of such a theory will be addressed. Finally, in section "Conclusions and Future Work", conclusions and future work are given.

Literature Review

Assembly-Oriented Design (AOD)

The concept of integrating ASP (Assembly Sequence Planning) with product design was introduced at the beginning of the previous decade in order to overcome the current limitations of Design for Assembly (DFA) and ASP approaches. Based on detailed product geometry, and a part-to-part oriented evaluation, DFA approaches lead most of the time to a redesign of products. In such a context, the issue of concurrent product design and ASP [8], also called AOD [9], has received much attention in research work during the last decade [10]. Product design and assembly sequence planning phases are normally undertaken separately and sequentially, which results in missing the true integration between these both phases. Here, the assembly-oriented practice of product development can be considered as a top-down approach by proactively considering the assembly related product design and its relationship issues in the early phases of the product development process. Among the models and ontology in the literature, relevant work has been done by focusing on assembly formalisms and semantics, in a collaborative product development context [11]. In addition, NIST has proposed generic models, such as CPM (Core Product Model) [12], CPM2 [13] and OAM (Open Assembly Model) [14] that are particularly relevant to represent the product in an integrated manner. As a result of these efforts, the introduction of semantics and knowledge in assembly models proved to provide a better interaction and understanding of the product architect's and assembly planner's intents. These representations facilitate reasoning about the assembly relationships network for assembly process planning and assembly line balancing issues as well, in a more efficient and formal way, at a high abstraction level of the product.

Mereotopology-Based Theories

Lesniewski [15] was the first to attempt to describe a mereology-based theory including theorems and axioms, in order to develop the parthood relation in a formal manner [16]. A key characteristic of mereology is the use of *part of* primitive [17], denoted ***P***, therefore representing part-whole relationship. Several researchers have highlighted limits and basic problems in mereology [18, 19]. As an automotive example, the speedometer is *part of* the dashboard, the dashboard is *part of* the car and the car is *part of* the garage. This seems to be logical sentences, but in no case the speedometer is *part of* the garage. The problem encountered here is that parthood relation is transitive (i.e., parthood means only one thing at once) in the first two statements and intransitive (i.e., parthood means something else) in the last one [20]. To overcome this paradox, the notion of topology has been incorporated and considered together with mereology so as to initiate mereotopology. This theory enables the qualitative formalisation of two fundamental predicates: parthood (i.e., one entity is *part of* another) and connection (i.e., an entity is *connected to* another, denoted ***C***) [19]. The current challenge of the mereotopology-based theory in engineering design is to consider the product as it is perceived in the real world [16]. In other words, mereotopology describes relationships between parts with an "engineering sense". Here, the engineering sense requires a more accurate and more structured prospect [19]. So, such theory enables the description of the engineering sense actually required in AOD. A proposed classification of the existing theories in product design is presented in Table 1.

Table 1 Mereotopological theories and their related primitives in assembly design

Author	Dimension			
	Spatial		Temporal	Spatiotemporal
	First primitive	Developed primitive		
Demoly et al. [16]	***P***	***O; IP; D; Point; X; St; B; T; IB***	$<$	O_t; $*$; $\subseteq_t$;
Kim et al. [11]	***P; IP***	***O; D; Point; X; St; B; T***		
Salustri [19]	***P; C***	***PP; O; EC; TPP; NTPP; SC; E***		

$<$ Precedence, $\subseteq_t$ temporal inclusion, $*$ temporal connection, ***B*** Boundary, ***C*** Connected, ***D*** Discrete, ***E*** Enclosure, ***EC*** Externally Connected, ***IB*** Internal Boundary, ***IP*** Interior Part, ***NTPP*** Non Tangential Proper Part, ***O*** Overlap, $\boldsymbol{O_t}$ temporal overlap, ***P*** Part of, ***Point, X*** Cross, ***PP*** Proper Part, ***SC*** Self-Connection, ***St*** Straddle, ***T*** Tangent, ***TPP*** Tangential Proper Part

Mereotopological Design of Assembly Design Evolution

The current section describes the proposed theory, called JANUS (Joined Aware-Ness and Understanding in assembly-oriented deSign with mereotopology) in the field of product-process design, and particularly focused on product design and assembly sequence planning. We propose to define the theory foundation based on the related objects and primitives to be used for each aspect (i.e., spatial, temporal and spatiotemporal).

Overall Description of the Proposed Theory

General Philosophy of the Theory

The proposed theory foundation is inspired from Bergson's idea [21], in which three temporal situations are considered: the present (i.e., the existence), the past (i.e., the essence) and the future [22]. In this context it is stated that the existence of an object precedes and leads to its essence: an object exists before being defined by concepts. The object cannot be first defined, as at the beginning the object does not exist (e.g., the assembly does not begin with all parts but just with few of them). The first principle of existentialism [23] states that the object is designed after the existence. The object exists only during its project (i.e., its lifecycle) when it is useful and in relation with others. It represents the same idea as the sentence from Descartes "I think, therefore I am". The object is not present as long as it is not recognized by the others as useful (i.e., so until it is not in relation with others). The others are therefore its condition of existence [23]. When the object is in relation with others, it realizes its function (i.e., the reason why this object has been designed). Without its function the object is useless and so does not exist [23]. For instance, sometimes when parts, involved in the functional flow, are added or moved, functions of the entire product can change.

The three temporal situations of Bergson can be adapted to AOD (Fig. 2). The purpose of the AOD approach is to take into account the assembly sequence as early as possible in the product development process. As such, by considering the assembly sequence and the assembly (i.e., part-to-part) relationships, product

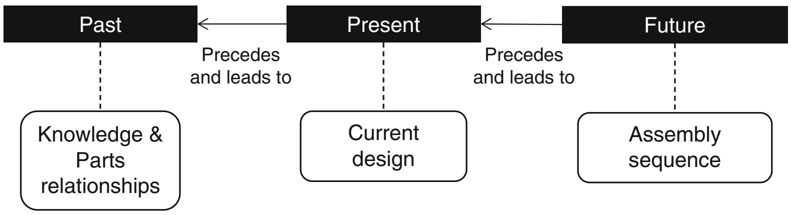


Fig. 2 The three temporal situations of Bergson applied to assembly design

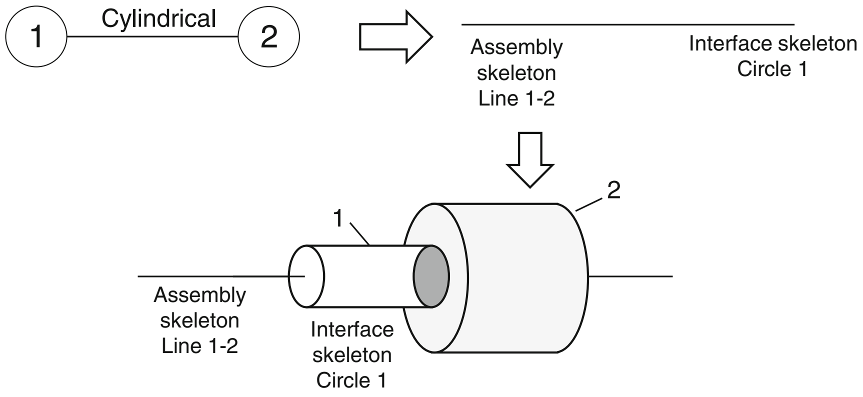

1
Cylindrical
2
Assembly skeleton Line 1-2
Interface skeleton Circle 1
1
2
Assembly skeleton Line 1-2
Interface skeleton Circle 1

Fig. 3 Example of cylindrical pair assembly and related skeletons

architects and designers can work with consistent product-process information knowledge and are also aware of the temporal and spatiotemporal aspects product-process definitions.

Object Change Definition

The theory frame, used to describe the object evolution during the AOD, is inspired from Le Moigne [24]. The product is composed of spatial parts assembled over time. Objects differ from others according to:

- Their position in space (i.e., give the localization compared to other objects);
- Their position in time (i.e., give the temporal position of the object regarding to others);
- Their form (i.e., give the object structure).

At the beginning of the design phase, part-to-part relationships are only known. Then they are translated into geometric skeletons in order to control assembly geometrical models in a centralized manner. Thus, the object form depends on two different geometrical entities (Fig. 3), namely:

- "Assembly skeleton", which ensures assembly positioning, is represented by the *k* letter (e.g., straight line representing the rotation and/or rotation axis);
- "Interface skeleton", which describes geometric boundaries (e.g., circle, square…) used to build a functional surface and supported by an assembly skeleton.

When one attribute (time, space and form) is modified, a change during the assembly design occurs. Humans used to describe everything in the spatial dimension (e.g., 1 h is measured with the little hand doing a complete round and coming back at its initial spatial position). Time can be considered with spatial coordinates:

this section introduces the concept of spatialized time or spatial time. As such, Bergson [21] writes “I simultaneously get that I think in duration and that I am in duration”. In the context of AOD, this statement means that I am designing the product assembly and I am aware of its assembly sequence. The definition of spatial time from Heidegger [22] can be adapted to the moving process, which starts from the object initial position and finishes when the object reaches its final destination (and changes its spatial primitives). By adding temporal part, it enables the description of the object changes during product design stages. As the purpose of this theory is to be adapted to every kind of design, it must remain a qualitative work. The object description – when (defined by assembly planner), where (by product architect), which form (by designer) – is known with this kind of modelling technique (i.e., theory frame). Afterwards they all understand the transformation process (how, why) [25]. When an object moves, properties from the past are preserved in the present (which is represented by the filiation relationship such as described later on). Therefore links between temporal regions exist. If the cause of the past changes is known, then the form of the object can be explained (but not predicted) in the present. The object adaptation is understandable (e.g., why a planar pair has been used?). Product life is a series of adaptations and decisions that the designer has to make. As an object is related to numerous other objects, decisions impact the entire assembly (and not only the object).

Description of the Spatial Dimension of the Theory

The spatial dimension of the theory describes the spatial mereotopological relationships between spatial objects (e.g., mechanical parts) at a specific instant of the assembly process. Each spatial object is considered as a spatial region at a specific instant. Here a discrete space is considered as the description of the spatial position of an object according to others depends on their spatial mereotopological primitives. An empty region (Ø), where no object is located, is also considered as a spatial region. The overall assembly (*A*) is the region considering all regions forming the final product.

Spatial relationships capture the way how objects are related at a certain time [26]. So the relative position of objects compared to others can be fully described [25]. Each region is in relation with at least another region. If there is no contact between two objects, then the *Discrete* ***D*** primitive is used. When the relationship is valid for every spatial primitive (except *discrete* primitive), the letter ***R***, which means *Relationship* is used. Consider that at time *t* there are two different parts: *x* and *y*.

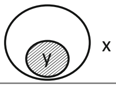
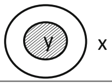
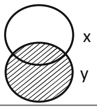
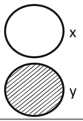
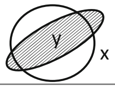
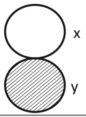
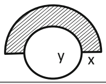
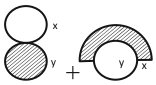
$$\forall \mathrm{t} \forall x \exists y / x_{\mathrm{t}} \boldsymbol{R} y_{\mathrm{t}}$$

Here the spatial primitives are introduced to formally describe product-process knowledge. They are based on Smith’s mereotopological primitives [27], and have already been addressed in Demoly et al. [16] and Kim et al. [11] who have both focused their research efforts in assembly design field. Salustri [19] also used similar primitives, which have the same properties but not the same names (e.g., *Interior Part* ***IP*** is equivalent to *Non Tangential Proper Part* ***NTPP*** or *Tangential Proper Part* ***TPP***). Moreover he did not use the *Crosse* ***X*** primitive, which seems to be relevant in the above mentioned context and which is intensively used during assembly sequence planning. By using mereotopological operators (see Table 2), eight spatial primitives have been considered and described in Table 3.

Table 2 Fundamental mereotopological operators

Symbol	$\wedge$	:=	$\rightarrow$	$\exists$	$\neg$	$\forall$
Name	Logical conjunction	Definition	Logical implication	Existential quantifier	Logical negation	Universal quantifier

Table 3 Spatial mereotopological primitives description and representation

Primitive name	Description	Representation	Mechanical example
Part of ***P***	$y\mathbf{P}x := \forall_z(z\mathbf{O}y \rightarrow z\mathbf{P}x)$		A part *is part of* an assembly
Interior Part of ***IP***	$y\mathbf{IP}x := y\mathbf{P}x \wedge \neg y\mathbf{T}x$		Balls are *interior part of* the ball bearing
Overlap ***O***	$y\mathbf{O}x := \exists z(z\mathbf{P}y \wedge z\mathbf{P}x)$		Two objects occupied a region in common
Discrete ***D***	$y\mathbf{D}x := \neg y\mathbf{O}x$		No contact between two parts
Crosses ***X***	$y\mathbf{X}x := \neg y\mathbf{P}x \wedge \neg y\mathbf{D}x$		Contact between a shaft and a ball-bearing
Tangent ***T***	$y\mathbf{T}x := \exists z(z\mathbf{P}x \wedge z\mathbf{B}y)$		Two parts are in contact along a planar surface
Boundary ***B***	$x\mathbf{B}y := \forall z(x\mathbf{P}x \rightarrow z\mathbf{St}y)$		Contact between convex and concave surfaces
Straddle ***St***	$x\mathbf{St}y := \forall z(x\mathbf{IP}z \rightarrow z\mathbf{X}y)$		Tangential contact between convex and concave surfaces

Description of the Temporal Dimension of the Theory

After describing spatial part of JANUS, the temporal aspect with temporal relationships and temporal objects is then defined. The lifetime of an object is broken down into several smaller temporal objects [28], which are created each time a part change occurs [25]. The object change is generally associated to an event or a process. The purpose is to describe changes that affect objects during design and assembly process. As soon as an event occurs, changes occur at a precise instant. An event indicates a discontinuity in a stable referential. There is a break between the "before" and the "after" event. In that sense, Sider [7] provides the following definition: x is an *instantaneous temporal part* of y if, at instant t, x exists at, but only at, t. Therefore the region is only spatial and there is no swept volume (i.e., no spatiotemporal region). On the other hand swept volumes, which are generated during process, are used to show all product evolution over time. So process expresses a change during the intern evolution of the object and is composed of an initial event, a duration between process and a final event. Here each temporal object is considered as a Temporal Region (TR). To understand the notion of temporal regions, mechanical assembly design history needs to be considered as a story. It describes when the base part (first part to be assembled) is placed, how the second is assembled to the previous one and so on. Like all stories, this story has temporal regions created at each new assembled spatial region. Temporal regions can be either intervals or instants [7]. Assembly operations are composed of temporal regions. There are two types of temporality: the temporality of the object itself (how it evolves over time) and the temporality of the object regarding to the others (which is shown with spatial primitives). The relative chronology between assembly operations is known using temporal relationships [25]. Table 4 shows the temporal relationships between two intervals (fourth column), one interval and one point (fifth column), and two points (with ***Bt*** the temporal boundary primitive) (sixth column). The proposed description is similar to Allen's work [30] who formalizes topological relationships between temporal intervals.

Temporal primitives are inspired from spatial mereotopological primitives and adapted to represent every temporal phenomenon that may be identified during an assembly process. Phenomena have been classified according to the type of mechanical assemblies. Just two of them can describe all types of assembly (see Table 5): *temporally precede*, denoted $<$, and *temporally tangent*, denoted ***Tt***. The different types of assembly are considered as follows [31]: interconnected serial, serial, constrained serial and parallel. In such a temporal dimension, time is considered as linear [32] and discrete.

Description of the Spatiotemporal Dimension of the Theory

Basic Problem with Spatiotemporal Visualization

As the main interest is concerned about the description of assembly design evolution, a section dedicated to spatiotemporal dimension has been added. Each

Table 4 Temporal mereotopological primitives description and representation

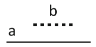
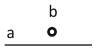
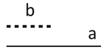
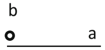
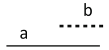
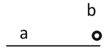
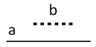
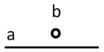
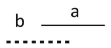
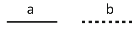
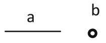
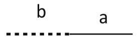
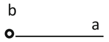
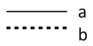
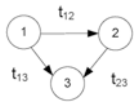
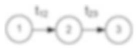
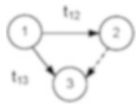
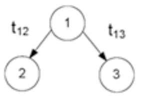
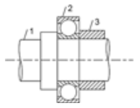
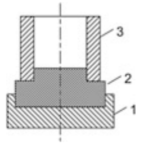
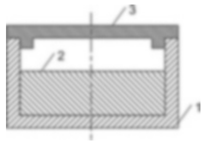
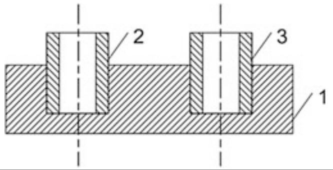
Primitive name	Symbol	Description	Representation between two intervals	Representation between an interval and an instant	Representation between two instants	Available in assembly type
Temporally Part	$\boldsymbol{Pt}$	$b\boldsymbol{Pt}a := \forall c(c\boldsymbol{Ot}b \rightarrow c\boldsymbol{Pt}a)$				aParallel
Start Temporally Part	a$\boldsymbol{Pts}$	$b\boldsymbol{Pts}a := \exists c\ (b\boldsymbol{Pt}a \wedge c\boldsymbol{Tt}b \wedge (c < a)$				aParallel
Finish Temporally Part	a$\boldsymbol{Ptf}$	$b\boldsymbol{Ptf}a := \exists c(b\boldsymbol{Pt}a \wedge b\boldsymbol{Tt}c \wedge a < c)$				aParallel
Temporally Interior Part	$\boldsymbol{IPt}$	$b\boldsymbol{IPt}a := b\boldsymbol{Pt}a \wedge \neg b\boldsymbol{Pts}a \vee \neg b\boldsymbol{Ptf}a$				Parallel
Temporally Overlap	$\boldsymbol{Ot}$	$b\boldsymbol{Ot}a := \exists c\ (c\boldsymbol{Pt}b \wedge c\boldsymbol{Pt}a)$				Parallel
Temporally Precede	$<$	$a < b := \exists c(a\boldsymbol{Tt}c \wedge b\boldsymbol{Ptf}c)$			a b **o o**	All
Temporally Tangent	$\boldsymbol{Tt}$	$b\boldsymbol{Tt}a := \exists c(c\boldsymbol{Pts}a \wedge c\boldsymbol{Bt}b)$			ab **oo**	All
Temporally equal	$=$	$b = a := b\boldsymbol{IPt}a \wedge a\boldsymbol{IPt}b$			**o** a **o** b	Parallel

Table 5 Assembly types of three components

Assembly type	Interconnected serial	Serial	Constrained	Parallel
Directed graph				
Example				

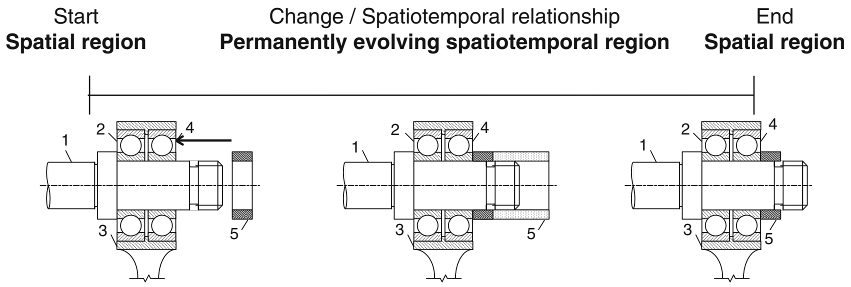

Start
Spatial region
Change / Spatiotemporal relationship
Permanently evolving spatiotemporal region
End
Spatial region
1
2
3
4
5

Fig. 4 Decomposition of the *y* object move

spatiotemporal object is considered as a spatiotemporal region such as swept volume during deformation, modification or transformation. The latter notion is the spatial region occupied by an object during a specific duration. By considering swept volume, the past coexists with the present. Skeletons do not have spatiotemporal parts, since spatiotemporal regions consist of spatial and temporal components [34]. For instance it is possible to see, during a change (e.g., Move of *part* 5), the path of an object and the related needed space to get to its final position (Fig. 4).

Several problems concerning spatiotemporal visualization can be encountered by human. Actually human can only see the initial spatial region and the final spatial region during a move. He cannot percept the swept volume (spatiotemporal region). For instance during an assembly sequence, when a part is moving in order to cross the shaft, the designer cannot see the part being moved. He has to break down the move into several instants. This is due to human intelligence which is only able to see motionless object and which fixes the time. Humans cannot think in terms of continuous evolution [21]. The definition of the theory with the spatiotemporal dimension sounds like cartoons. If an object must undergo two consecutive changes, then the final spatial region of the first change is the first spatial region of the second change. This type of description does not provide all information about what happens during intervals. A quantitative study has to be carried out with laws to know what happens in the interior of an interval.

Filiation Relationships

Descartes [35] made an experiment with wax. When he saw it, the wax was solid. But when he put it near fire, it melts. His conclusion is that with human understanding, the wax is wax and will always be wax whatever happens to it (e.g., transformation). Moreover Sider [7] and Hawley [29] describes the object as persisting over time. When an object changes, its properties change (but not itself). The object will have attributes showing for instance if the object is deformed or not. For few changes (e.g., union or split), a new object is created from the "initial" object. The same identity basis is shown with filiation relationships. It is a dependence relationship where the kinship is transmitted from the parent to the child. In assembly design, the parent can be an assembly and the child, which is based on its parent, can be a part. It enables the identification of every object and its uniqueness. Based on the identity concept, it is possible to distinguish an object from others. Two types of filiation relationships can be found during AOD phase [26]:

- γ: Continuation relationship (i.e., exactly the same object but at two different times);
- δ: Derivation relationship (i.e., a part of the identity from the original object is present in the changed object).

In the proposed JANUS theory, the continuation relationship is considered as implicit. The latter always occur between two objects with the same name at different instants. Derivation relationships occur when two objects are in relation with spatiotemporal primitives and do not have the same name.

Classification of Spatiotemporal Primitives

Changes have spatiotemporal aspects and need spatiotemporal relationships to link spatial objects. The latters, which are used to link spatial and temporal dimensions [26], can be described as follow:

- Between two spatial regions at two different instants (e.g., ***Move***);
- Between a spatial and an empty spatial region at two different instants (e.g., ***Deletion***);
- Between an empty spatial and a spatial region at two different instants (e.g., ***Addition***).

As a consequence, spatiotemporal primitives express the change over space and time [36], which provides a powerful mechanism to visualize and communicate design and assembly intents [37]. The evolution of an object may be seen as a succession of events and processes. Consider two different times t_i and t_j and two different parts (x and y):

$$\forall \mathrm{ti} \neq \mathrm{t\,j} \forall x \exists y / x_{\mathrm{ti}} \boldsymbol{R} y_{\mathrm{t\,j}}$$

In the domain of mechanical engineering, two parts during the assembly cannot physically *overlap*, on the contrary two spatiotemporal regions can *overlap* themselves at two different instants. That is the reason why spatiotemporal primitives are needed to model the actual world. By describing new spatiotemporal primitives, product design evolution is managed and understood. In Table 6, all spatiotemporal primitives are described and classified (according to Haddad's [36] classification) in terms of their changes over time, space and form. For growing and movement, the change occurs during the process (SpatioTemporal Region STR form changes),

Table 6 Classification of changes and associated spatiotemporal primitives

Change types	Spatiotemporal primitives	Temporal change	Spatial change	Form change	Parts number change	Example of event or process
Movement	***Move***	●	●	STR +SRf	○	P: position an object
	Rotation	●	●	STR +SRf	○	P: orient an object
Replacement process	***Permutation***	●	●	STR +SRf	○	P: designer oversight
Basic change	***Addition***	●	○	SRf	●	E: missing object
	Deletion	●	○	SRf	●	E: useless object
Restructuring processes	***Split***	●	○	SRf	●	E: complex assembly
	Union	●	○	SRf	●	E: subassem-bly creation
Trans-formation	***Growing***	●	○	STR +SRf	○	P: object too small
	Decrease	●	○	STR +SRf	○	P: object too large
	Change of form	●	○	STR +SRf	○	P: change of idea
	Deformation	●	○	STR +SRf	○	P: rivet positioning

Table 7 Legend of the spatiotemporal graph

$\underline{\text{T}}$	Spatial relationship (e.g., Tangent)
$\underline{\text{Move}}$	Spatiotemporal relationship (e.g., Move)
TR_i	Temporal region number i
(y)	Spatial region (e.g., named y)
$\boxed{y}$	Spatiotemporal region (e.g., named y)
(k)	Assembly skeleton (e.g., named k)
(f)	Interface skeleton (e.g., named f)

but also during the instant just after (change of Spatial Region final SRf). On the contrary for union, changes in form just occur at the final spatial region.

Each spatiotemporal primitive is described in details using spatial and temporal mereotopological primitives (see Tables 8 and 9). The spatiotemporal primitive called ***Cylindrical OP*** (Operation) is described in details in a graph (Fig. 5) with its legend (Table 7). Processes are considered continuous, non-instantaneous (e.g., object cannot move from one point to another in one instant) and linear. *B* ***Planar OP*** *A* means that *A* is the base part and *B* is moving to be positioned in a planar

Table 8 Mereotopological description of the "Movement" primitives

Spatiotemporal primitives	Mereotopological description
Move (k$_1$ and k$_2$ point, line or surface)	$(x\,\boldsymbol{T}k_1)_{\mathrm{TR1}} \wedge ((x\,\boldsymbol{T}k_2) \wedge (x\,\boldsymbol{R}\,y))_{\mathrm{TR2}}$
Cylindrical OP and ***Prismatic OP*** (k line, f surface)	$((x\boldsymbol{D}y) \wedge (x\boldsymbol{O}k) \wedge (y\boldsymbol{D}k))_{\mathrm{TR1}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR2}} \wedge ((x\boldsymbol{D}y)\ (x\boldsymbol{O}k) \wedge (y\boldsymbol{O}k))_{\mathrm{TR3}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR4}} \wedge$ $((x\boldsymbol{X}y) \wedge (x\boldsymbol{O}k) \wedge (y\boldsymbol{O}k))_{\mathrm{TR5}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR6}} \wedge ((x\boldsymbol{X}y) \wedge (x\boldsymbol{O}k) \wedge (y\boldsymbol{O}k) \wedge (y\boldsymbol{T}f))_{\mathrm{TR7}}$
Screw OP (k$_1$ line, f$_1$ area, f$_2$ and f$_3$ elliptical curve)	$((x\boldsymbol{D}y) \wedge (x\boldsymbol{O}k_1) \wedge (y\boldsymbol{D}k_1))_{\mathrm{TR1}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR2}} \wedge ((x\boldsymbol{D}y) \wedge (x\boldsymbol{O}k_1) \wedge$ $(y\boldsymbol{O}k_1))_{\mathrm{TR3}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR4}} \wedge ((x\boldsymbol{X}y) \wedge (x\boldsymbol{O}k_1) \wedge (y\boldsymbol{O}k_1))_{\mathrm{TR5}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR6}} \wedge ((x\boldsymbol{X}y) \wedge$ $(x\boldsymbol{O}k_1) \wedge (y\boldsymbol{O}k_1) \wedge (y\boldsymbol{T}f_1) \wedge (f_2\boldsymbol{O}f_3) \wedge (y\boldsymbol{T}f_2)(x\boldsymbol{T}f_3))_{\mathrm{TR7}}$
Revolute OP (k$_1$ line and k$_2$ plane, f surface)	$((x\boldsymbol{D}y) \wedge (x\boldsymbol{O}k_1) \wedge (y\boldsymbol{D}k_1))_{\mathrm{TR1}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR2}} \wedge$ $((x\boldsymbol{D}y) \wedge (x\boldsymbol{O}k_1) \wedge (y\boldsymbol{O}k_1))_{\mathrm{TR3}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR4}} \wedge$ $((x\boldsymbol{X}y) \wedge (x\boldsymbol{O}k_1) \wedge (y\boldsymbol{O}k_1))_{\mathrm{TR5}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR6}} \wedge$ $((x\boldsymbol{T}y) \wedge (x\boldsymbol{X}y) \wedge (x\boldsymbol{O}k_1) \wedge (x\boldsymbol{T}k_2) \wedge (x\boldsymbol{T}f) \wedge (y\boldsymbol{O}k_1) \wedge (y\boldsymbol{T}k_2) \wedge (y\boldsymbol{T}f))_{\mathrm{TR7}}$
Planar OP (k plane, f surface) and ***Point-contact OP*** (k point, f point) and ***Line-contact OP*** (k line, f segment)	$((x\boldsymbol{D}y) \wedge (x\boldsymbol{T}k))_{\mathrm{TR1}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR2}} \wedge ((x\boldsymbol{T}y) \wedge (x\boldsymbol{T}f) \wedge (y\boldsymbol{T}k) \wedge (f\boldsymbol{T}k))_{\mathrm{TR3}}$
Spherical OP (k point, f surface)	$(x\boldsymbol{D}y)_{\mathrm{TR1}} \wedge (y\boldsymbol{Move}x)_{\mathrm{TR2}} \wedge ((x\boldsymbol{St}y) \wedge (x\boldsymbol{IP}y) \wedge (x\boldsymbol{T}f) \wedge (y\boldsymbol{T}f) \wedge (x\boldsymbol{O}k) \wedge (y\boldsymbol{O}k))_{\mathrm{TR3}}$
Rotation (k$_1$ point, k$_2$ and k$_3$ line)	$((x\boldsymbol{O}k_2) \wedge (k_1\boldsymbol{O}k_2) \wedge (k_1\boldsymbol{O}k_3) \wedge (k_2\boldsymbol{D}k_3))_{\mathrm{TR1}} \wedge$ $(x\boldsymbol{Move}k_1)_{\mathrm{TR2}} \wedge ((x\boldsymbol{O}k_2) \wedge (k_1\boldsymbol{O}k_2) \wedge (k_1\boldsymbol{O}k_3) \wedge (k_2\boldsymbol{O}k_3) \wedge (x\boldsymbol{O}k_3))_{\mathrm{TR3}}$

Table 9 Mereotopological description of the other spatiotemporal primitives

Spatiotemporal primitives	Mereotopological description and final representation
Serial permutation (k_1 line, f_1 and f_2 surface)	$((x\boldsymbol{R}z) \wedge (y\boldsymbol{R}z) \wedge (x\boldsymbol{R}y) \wedge (x\,\boldsymbol{T}f) \wedge (x\boldsymbol{R}k) \wedge (z\,\boldsymbol{T}f_1) \wedge (z\,\boldsymbol{T}f_2) \wedge (y\,\boldsymbol{T}f_2) \wedge (y\boldsymbol{R}$ $k_1))_{\mathrm{TR1}} \wedge (x\boldsymbol{MOVE}z)_{\mathrm{TR2}} \wedge ((x\boldsymbol{D}z) \wedge (y\boldsymbol{R}z) \wedge$ $(x\boldsymbol{D}y) \wedge (x\boldsymbol{R}k_1) \wedge (zTf_2) \wedge (y\,\boldsymbol{T}f_2) \wedge (y\boldsymbol{R}k_1))_{\mathrm{TR3}} \wedge (y\boldsymbol{Move}z)_{\mathrm{TR4}} \wedge ((x\boldsymbol{D}z) \wedge$ $(y\boldsymbol{D}z) \wedge (x\boldsymbol{D}y))_{\mathrm{TR5}} \wedge (y\boldsymbol{Move}z)_{\mathrm{TR6}} \wedge ((x\boldsymbol{D}z) \wedge (y\boldsymbol{R}z) \wedge (x\boldsymbol{D}y) \wedge (y\,\boldsymbol{T}f_1) \wedge (y\boldsymbol{R}k_1) \wedge$ $(z\,\boldsymbol{T}f_1))_{\mathrm{TR7}} \wedge (x\boldsymbol{Move}z)_{\mathrm{TR8}} \wedge ((x\boldsymbol{R}z) \wedge$ $(y\boldsymbol{R}z) \wedge (x\boldsymbol{R}y) \wedge (x\,\boldsymbol{T}f_2) \wedge (x\boldsymbol{R}k_1) \wedge (z\,\boldsymbol{T}f_1) \wedge (z\,\boldsymbol{T}f_2) \wedge (yTf_1) \wedge (y\boldsymbol{R}k_1))_{\mathrm{TR9}}$
Parallel permutation (k_1 and k_2 line, f_1 and f_2 surface)	$((x\boldsymbol{R}z) \wedge (y\boldsymbol{R}z) \wedge (x\boldsymbol{D}y) \wedge (x\,\boldsymbol{T}f_1) \wedge (x\boldsymbol{R}k_1) \wedge$ $(z\,\boldsymbol{T}f_1) \wedge (z\,\boldsymbol{T}f_2) \wedge (y\,\boldsymbol{T}f_2) \wedge (y\boldsymbol{R}k_2) \wedge (f_1\boldsymbol{D}f_2) \wedge (k_1\boldsymbol{D}k_2))_{\mathrm{TR1}} \wedge ((x\boldsymbol{Move}z) \wedge (y\boldsymbol{Move}z))_{\mathrm{TR2}} \wedge ((x\boldsymbol{D}z) \wedge (y\boldsymbol{D}z) \wedge$ $(x\boldsymbol{D}y))_{\mathrm{TR3}} \wedge (x\boldsymbol{Move}z) \wedge (y\boldsymbol{Move}z)_{\mathrm{TR4}} \wedge ((x\boldsymbol{R}z) \wedge (y\boldsymbol{R}z) \wedge (x\boldsymbol{D}y) \wedge (x\,\boldsymbol{T}f_2) \wedge (x\boldsymbol{R}k_2) \wedge$ $(z\,\boldsymbol{T}f_1) \wedge (z\,\boldsymbol{T}f_2) \wedge (y\,\boldsymbol{T}f_1) \wedge (y\boldsymbol{R}k_1) \wedge (f_1\boldsymbol{D}f_2) \wedge (k_1\boldsymbol{D}k_2))_{\mathrm{TR5}}$
Addition (f surface)	$((x\boldsymbol{P}\varnothing))_{\mathrm{TR1}} \wedge ((x\boldsymbol{P}A) \wedge (x\,\boldsymbol{T}f))_{\mathrm{TR2}}$
Deletion (f surface)	$((x\boldsymbol{P}A) \wedge (x\,\boldsymbol{T}f))_{\mathrm{TR1}} \wedge (x\boldsymbol{P}\varnothing)_{\mathrm{TR2}}$
Split (f_1 and f_2 surface)	$((\varnothing\,\boldsymbol{Addition}\,y) \wedge (\varnothing\,\boldsymbol{Addition}\,z) \wedge (x\boldsymbol{\delta}y) \wedge (x\boldsymbol{\delta}z) \wedge (x\,\boldsymbol{T}f_1))_{\mathrm{TR1}} \wedge$ $((y\boldsymbol{P}x) \wedge (z\boldsymbol{P}x) \wedge (z\,\boldsymbol{T}f_1) \wedge (y\,\boldsymbol{T}f_2))_{\mathrm{TR2}} \wedge ((x\ \ \boldsymbol{Deletion}\ \ \varnothing) \wedge (z\ \ \boldsymbol{T}\ \ f_1) \wedge (y\,\boldsymbol{T}f_2))_{\mathrm{TR3}}$
Union (f_1 and f_2 surface)	$((x\boldsymbol{Addition}\varnothing) \wedge (z\,\boldsymbol{T}f_1) \wedge (y\,\boldsymbol{T}f_2))_{\mathrm{TR1}} \wedge$ $((y\boldsymbol{P}x) \wedge (z\boldsymbol{P}x) \wedge (z\,\boldsymbol{T}f_1) \wedge (y\,\boldsymbol{T}f_2))_{\mathrm{TR2}} \wedge$ $((\varnothing\,\boldsymbol{Deletion}\,y) \wedge (\varnothing\,\boldsymbol{Deletion}\,z) \wedge (x\boldsymbol{\delta}y) \wedge (x\boldsymbol{\delta}z)(x\,\boldsymbol{T}f_1))_{\mathrm{TR3}}$
Change of form, deformation, growing and ***decrease*** (f_1 and f_2 surface)	$(x\ \ \boldsymbol{T}\ \ f_1)_{\mathrm{TR1}} \wedge (x\ \ \boldsymbol{T}\ \ f_2)_{\mathrm{TR2}}$

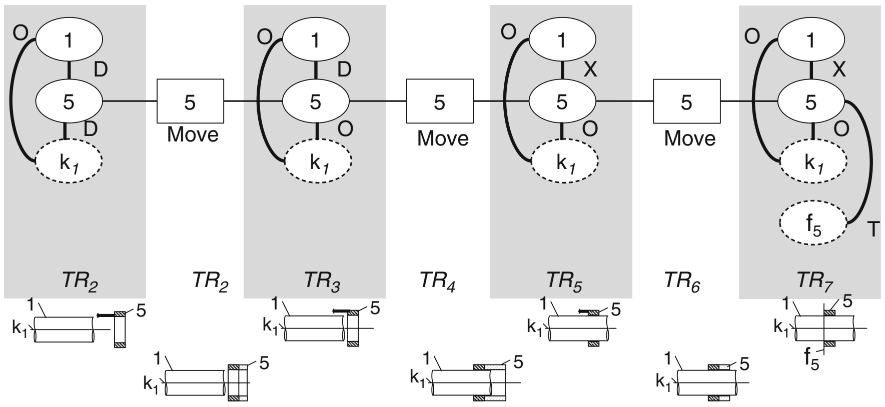
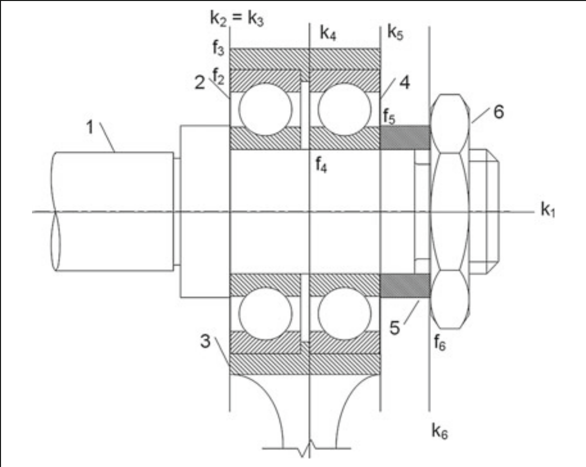


Fig. 5 Description of ***cylindrical OP*** in a spatiotemporal graph

Table 10 Cross-section view of the considered mechanical assembly and Parts list

	No.	Part name
 k2 = k3 k4 k5 f3 f2 2 4 1 f5 6 f4 k1 5 3 f6 k6	1	Shaft
	2	Bearing 1
	3	Hub
	4	Bearing 2
	5	Spacer
	6	Nut
	Assembly skeletons	Line *k_1*, Plans *k_2*, *k_3*, *k_4*, *k_5* and *k_6*
	Interface skeletons	Surfaces *f_2*, *f_3*, *f_4*, *f_5* and *f_6*

manner. *B* ***Change of form*** *A* means that *B* is deformed because of *A*. For next descriptions, *TRi* always precedes *TRi+1*.

Illustrative Case Study of the Theory

Table 10 represents the example used to illustrate the JANUS theory, which is composed of six spatial regions: Parts *1, 2, 3, 4, 5* and *6*. The assembly sequence has been obtained using the ASDA algorithm from Demoly et al. [16]. The best

Table 11 Example of assembly operations and their associated primitives

Description of operations [33]	Temporal relationships between operations
OP10: Positioning of *1* (base part) *OP20*: Positioning of *3* *(base part of SA)* $\left.\right\}$ *SA* *OP30*: Insertion of *2* on *3* *OP40*: Insertion of *4* on *3* *OP50*: Insertion of *SA* on *1* *OP60*: Insertion of *5* on *1* and *4* *OP70*: Insertion of *6* on *1* and *5*	*OP10* = *OP20* *OP20* ***Pts*** *OP30* *OP30* ***Ot*** *OP40* *OP20* < *OP40* (*OP40* < *OP50*) ∧ (*OP30* < *OP50*) (*OP50* ***Ot*** *OP60*) ∧ (*OP50* < *OP60*) (*OP70* ***Ptf*** *OP50*) ∧ (*OP70* ***Ptf*** *OP60*)

Table 12 Mereotopological description of the mechanical case study

OP n°	Mereotopological description
10 and 20	$(1\ \mathbf{D}\ 3) \wedge (1\ \mathbf{O}\ k_1) \wedge (3\ \mathbf{T}\ k_3) \wedge (3\ \mathbf{T}\ f_3)$
30	*2* ***Revolute OP*** *3*
40	*4* ***Cylindrical OP*** *3*
50	*SA* ***Cylindrical OP*** *1*
60	*5* ***Cylindrical OP*** *1*
	5 ***Planar OP*** *4*
70	*6* ***Screw OP*** *1*
	6 ***Planar OP*** *5*

assembly sequence is: {*1*, (*3*, *2*, *4*), *5*, *6*}. The spatial regions *2*, *3* and *4* constitute a Sub-Assembly (*SA*). Assembly operations are described in Table 11. Moreover product-process mereotopological description is described in Table 12. With the theory, designers will know how and when parts are related, therefore that will improve their understanding about the assembly process and the product-process evolution.

Discussions

The proposed theory has some limitations, as it does not take into account the exact shape of the object (e.g., sphere), its exact position (e.g., within a coordinates system) and its orientation during its insertion. But the purpose is to have a qualitative theory, which could be applied to every assembly of every shape or size, as at the early design phases the geometry is not known. In the future the theory will be extended to describe not only assembly knowledge, but also the knowledge from the manufacturing process (e.g., additive manufacturing, tapping…). In that case, new spatiotemporal regions will be described such as material flux, and also new primitives. Moreover this kind of theory could be applied to any domains where description of relationships between objects is needed. For instance it is already used in GIS (Geographic Information Systems). If someone wants to implement the theory in this domain, he will just have to use it as a basis and develop its own regions (e.g., city) and its own primitives. Primitives are very specific to the domain. For instance in GIS, "**Revolute OP**" is completely obsolete.

Conclusions and Future Work

The paper has introduced a novel mereotopological theory, called JANUS, in AOD based on three dimensions: spatial, temporal and spatiotemporal. Spatial and temporal objects and primitives have been defined, as well as the association between both dimensions. Indeed new spatiotemporal objects and primitives have been presented in order to be able to describe product-process knowledge and information. With the spatiotemporal dimension and the mereotopology-based theory, information has been added to objects so as to express their history in a consistent and understandable manner. This description aids to understand phenomenon during AOD. The actual stake is to get a long-term dynamic vision of the space in order to facilitate the understanding of assembly and design changes. In future work, this JANUS theory will be implemented into an ontology by using OWL-DL (Web Ontology Language) and SWRL (Semantic Web Rule Language) languages and also the Protégé software. The ontology will allow formalizing the theory with axioms and will be machine-interpretable. In addition, a specific reasoning layer will be developed so as so reason on spatiotemporal associations within PLM systems. Such efforts will enable the introduction of novel procedures for consistency checking of product-process information and knowledge in PLM.

References

1. Demoly F, Deniaud S, Gomes S (2012) Towards an harmonious and integrated management approach for lifecycle planning. In: International conference on advanced production management systems, Greece
2. Kusiak A, Salustri FA (2007) Computational intelligence in product design engineering: review and trends. IEEE Trans Syst Man Cybern Part C Appl Rev 37(5):766
3. Zeng Y, Gu P (1999) A science-based approach to product design theory Part II: formulation of design requirements and products. Robot Comput Integr Manuf 15(4):341–352
4. Huang GQ, Lee SW, Mak KL (1999) Web-based product and process data modelling in concurrent design for X. Robot Comput Integr Manuf 15:53–63
5. Helms RW (2002) Product data management as enabler for concurrent engineering. PhD thesis, Technische Universiteit Eindhoven
6. Sapuan SM, Osman MR, Nukman Y (2006) State of the art of the concurrent engineering technique in the automotive industry. J Eng Des 17(2):143–157
7. Sider T (2001) Four dimensionalism: an ontology of persistence and time. Clarendon, Oxford

8. Zha XF, Du H (2002) A PDES/STEP-based model and system for concurrent integrated design and assembly planning. Comput Aided Des 34(14):1087–1110
9. Mantripragada R (1998) Assembly oriented design: concepts algorithms and computational tools. PhD thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology
10. Wang L, Keshavarzmanesh S, Feng H-Y, Buchal RO (2008) Assembly process planning and its future in collaborative manufacturing: a review. Int J Adv Manuf Technol 41(1–2):132–144
11. Kim KY, Yang H, Kim DW (2008) Mereotopological assembly joint information representation for collaborative product design. Robot Comput Integr Manuf 24(6):744–754
12. Fenves SJ, Foufou S, Bock C, Sriram RD (2008) CPM: A core model for product data. J Comput Inf Sci Eng 5:238–246
13. Fenves SJ, Foufou S, Bock C, Sriram RD (2008) CPM2: a core model for product data. J Comput Inf Sci Eng 8(1):1–14
14. Sudarsan R, Fenves SJ, Sriram RD, Wang F (2005) A product information modeling framework for product lifecycle management. Comput Aided Des 37(13):1399–1411
15. Lesniewki S (1929) Fundamentals of a new system of the foundations of mathematics. Fundam Math 14:1–81
16. Demoly F, Matsokis A, Kiritsis D (2012) A mereotopological product relationship description approach for assembly oriented design. Robot Comput Integr Manuf 28(6):681–693
17. Duntsch I, Wang H, McCloskey S (2001) A relation-algebraic approach to the region connection calculus. Theor Comput Sci 255:63–83
18. Varzi AC (1998) Basic problems of mereotopology, Formal ontology in information systems. Ios Press, Italy, pp 29–38
19. Salustri FA (2002) Mereotopology for product modeling. A new framework for product modeling based on logic. J Des Res 2:2
20. Salustri FA, Lockledge JC (1999) Towards a formal theory of products including mereology. In: Proceedings of the 12th international conference on engineering design, Munich, pp 1125–1130
21. Bergson H (1923) Creative evolution. H. Holt and Company, New York
22. Heidegger M (1962) Being and time. Harper & Row, New York
23. Sartre J-P (1975) Existentialism is a humanism. In: Kauffman W (ed) Existentialism from Dostoevsky to Startre, rev. edn. Meridian/Penguin, New York, pp 345–369
24. Le Moigne J-L (1994) La théorie du système général: théorie de la modélisation. Presses universitaires de France, Paris
25. Rodier X, Saligny L, Lefebvre B, Pouliot J (2010) ToToPI a GIS for understanding urban dynamics based on the OH FET model. In: Fricher B, Crawford J, Koler D (eds) Computer application and quantitative methods in archaeology. Granada, pp 337–349
26. Del Mondo G, Stell JG, Claramunt C, Thibaud R (2010) A graph model for spatiotemporal evolution. J Univers Comput Sci 16(11):1452–1477
27. Smith B (1996) Mereotopology: a theory of parts and boundaries. Data Knowl Eng 20 (3):287–303
28. Hadjieleftheriou M, Kollios G, Tsotras VJ, Gunopulos D (2002) Efficient indexing of spatiotemporal objects. Advances in database technology. Springer, Prague, Czech Republic, p 251–268
29. Hawley K (2004) Temporal parts. In: Zalta EN (ed) The Stanford encyclopedia of philosophy (Winter 2010 Edn). http://plato.stanford.edu/archives/win2010/entries/temporal-parts/
30. Allen JF (1983) Maintaining knowledge about temporal intervals. Commun ACM 26 (11):832–843
31. Demoly F, Yan XT, Eynard B, Rivest L, Gomes S (2011) An assembly oriented design framework for product structure engineering and assembly sequence planning. Robot Comput Integr Manuf 27(1):33–46
32. Renolen A (1999) Concepts and methods for modelling temporal and spatiotemporal information. Partial fulfilment for the degree "Thesis", NTNU

33. Boothroyd G, Dewhurst P, Knight W (2002) Product design for manufacture and assembly, 2nd edn. Taylor & Francis, Boca Raton, FL
34. Bittner T (2001) Rough sets in spatio-temporal data mining. Temporal, spatial, and spatio-temporal data mining. Springer, Berlin Heidelberg, pp 89–104
35. Cottingham J (1996) Meditations on first philosophy with selections from the objections and replies. Cambridge University Press, Cambridge
36. Haddad H (2009) Une approche pour supporter l'analyse qualitative des suites d'actions dans un environnement géographique virtuel et dynamique. Thèse de Doctorat, Département d'informatique, Université Laval
37. McKinney K, Kim J, Fischer M, Howard C (1996) Interactive 4D-CAD. In: Proceedings of the third congress on computing in civil engineering, Anaheim, pp 383–389

Part II
Design Cognition

Combination of Eye Tracking and Think-Aloud Methods in Engineering Design Research

Anne Ruckpaul, Thomas Fürstenhöfer, and Sven Matthiesen

Abstract In order to understand the engineers' behavior while designing it needs to be revealed how the designer perceives function-shape-relations of a technical system. Eye tracking is a adequate method to observe the proceedings of the human analyzing technical systems. However, further information for concluding on the designer's perception is needed. Well-established methods in order to elicit further implicit and tacit knowledge are think aloud approaches. The combination think-aloud and eye tracking is not yet observed in detail; especially how think-aloud influences the eye movements and which additional data is gained in the context of engineering design research. This paper presents an eye tracking study, which compares two think-aloud methods, concurrent and retrospective think-aloud, in combination with eye tracking. The results show no significant influence on the eye movements. However, the two think-aloud approaches generate differing contents of verbalizations and complement the recorded gaze data with different scopes.

Introduction

Understanding the engineers' behavior while designing is one focus of the past and current design research community. Still the human behavior in design is a brought field of research with a lot of unknown. The recognition of function-shape-relations is one of those fields. This paper addresses the combination of eye tracking and think-aloud methods in the context of function-shape-relations for eliciting further insights on the human behavior in design. The engineering designer is the driver for innovation in product development. Thus, the methods for product development should fit to the human designer. The aim is to measure and quantify the human behavior in order to adapt the methods to the human designer.

For designing, the human needs the ability to pre-think form and shape as well as the functional structure of a technical system. The transformation of functions into

A. Ruckpaul (✉) • T. Fürstenhöfer • S. Matthiesen
Karlsruhe Institute of Technology, Karlsruhe, Germany
e-mail: anne.ruckpaul@kit.edu

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_5

an adequate shape is an iterative process of analysis of the existing (or pre-thought) system and the synthesis into a new structure [1, 2]. During synthesis a new structure is developed that should fulfill the intended function. The analysis represents the examination if the designed structure actually fulfills the intended function. Synthesis and analysis are two accompanying steps.

Eye tracking supports observations of the designer's cognition abilities. The gaze point reveals which shape is considered to suggest a certain function. However, the gaze point alone does not elicit if the human perceived information or if the designer stares at a certain part of a technical system while thinking about a different problem. Hence, for these observations on the human behavior the combination of eye tracking with other methods is preferable. As this paper focuses on the analysis process of product development, it is advantageous to also record the subjects' thoughts. As measurement for the behavior during analysis, the eye movements in combination with verbalizations are chosen. Think-aloud methods are well established and elicit implicit and tacit knowledge. The verbalization of the designer thoughts is helpful and often crucial to prevent misinterpretations. However, the interactions and influences between eye tracking and think-aloud methods need to be discovered before combining both approaches during studies. The main objective of this paper is the identification of those interactions and influences.

The paper presents the results of an experiment examining different think-aloud methods in order to support the choice of an advantageous method depending on the case of application. The research questions this paper addresses are: (1) Do think-aloud methods influence the eye movements of the subjects? (2) Do different think-aloud methods elicit the same contents of verbalizations?

After giving an overview on think-aloud in eye tracking research, the motivation and the resulting research questions are presented. The experimental setup is followed by an exemplary presentation of the experiment results for a deeper understanding of the conducted eye tracking study. The analysis concentrates mainly on the produced verbal data and the comparison of two think-aloud methods.

Think-Aloud Methods in Research

In empirical research interviews, observations and protocol analysis are often used and well established. As shown by Ahmed [3] tacit knowledge cannot be elicited by all techniques. The protocol analysis by Ericsson and Simon [4] is one method to evoke information of the thinking process of the designer. This is done by letting the subjects think-aloud during the given design task.

Think-aloud methods have been introduced as thought analysis in the 1920s by Watson [5]. Ericsson and Simon [4] made it applicable for observing cognitive processes and integrated the method in the protocol analysis. It is often used in empirical studies in order to elicit information on the behaviour of the subject and his or her implicit and tacit knowledge.

There are two different think-aloud approaches, the concurrent and the retrospective think-aloud method. It is named concurrent think-aloud when the subject talks while fulfilling a task and verbalises his/her thoughts or comments the own course of action. Different opinions on the influence of the concurrent think-aloud method on the performance can be found in literature. Van Someren et al. [6] show that thinking aloud stresses all parts of the cognitive system and thus, slows down not only the eye movements but also the exploration and learning process. Davies [7] states that concurrent think-aloud even influences the course of action and the subject changes the order of performing the task. However, in complex tasks concurrent think-aloud can be essential because it provides the momentous perspective. An experiment, which is conducted with the retrospective think-aloud approach, could deviate or ignore verbal comments on the momentous perspective. Concurrently thinking aloud can then be used in order to collect task-related thoughts as stated by Ryan and Haslegrave [8].

When recording these thoughts or comments after the actual task fulfilling, retrospective think-aloud is conducted. A very effective way is the gaze video cued retrospective think-aloud. As stimulation during retrospective think-aloud the subject regards a video of his or her own eye movement data. Also for commenting the recorded data retrospectively, different results of observations can be found in literature. Van Gog et al. [9] revealed that the verbalisations with cued think-aloud elicited more information on the actions done and the approach of the subjects to solve the given problem. In contrast, Kuusela and Paul [10] state that concurrently thinking-aloud produces more action and outcome statements than retrospectively commenting. Further, they explain that retrospective think-aloud gives more information on strategies and reasons for actions. In design research, Gero and Tang [11] compare concurrent and retrospective protocols a study on observing the design process in detail. They state that the concurrent protocol is advantageous for revealing functional aspects during the design process. More information on solutions and evaluations are evolved by retrospective protocols. However, Gero argues that concurrent and retrospective think-aloud have the same qualification on revealing additional information in the field of process-orientated aspects of designing.

Due to the varying conclusions in literature depending on the use case and the context of the think-aloud application, it is obvious that the influence of think-aloud needs to be observed context-dependent. For observations on the design engineer's behaviour while analysing a technical system the advantages and disadvantages of think-aloud methods need to be examined. Hence, the following research questions derive:

Does the concurrent think-aloud method affect the eye movements of the subject when analysing a technical system in comparison to a quiet experiment? Do concurrent think-aloud and retrospective think-aloud generate the same set of additional information and perspectives for eye tracking experiments?

Method and Experimental Setup

For answering the research questions, an eye tracking study for the comparison of concurrent and retrospective think-aloud was designed. For embedding the context of design engineering, the task of the study is the approval of a technical drawing. The analysis of a technical system is one key competence in engineering design as the design process is an iterative process of synthesis and analysis [1]. Additionally, the approval of technical systems is a compulsory step during product development as the technical drawing needs to be check for correctness before it is forwarded to the manufacturing.

A drawing of a technical system was presented to each participant of the study. They should decide if the system is fulfilling its function or if there is an error. For this purpose the participants were divided in two groups, each group for one think-aloud method. The same stimulus was shown to all participants in order to exclude the influence of differing tasks. For concurrent think-aloud, which is abbreviated by CTA in the following, the subjects were instructed to verbalise their thoughts. For retrospective think-aloud (RTA) it was chosen to use the previous recorded eye movements and thus to conduct gaze video cued RTA. The verbalisations of the subjects were recorded during the whole experiments.

The eye movements of both groups are recorded for answering the first research question if CTA affects the gaze behaviour. For this observation, the RTA group is used as comparison group. The generated verbal data of both groups is analysed by using coding schemes according to the content of the verbalisation. The distribution of verbal data to each type is used for answering the second research question if the different think-aloud methods elicit the same contents of protocol. The results of the study are presented in section "Analysis and Results".

After presenting the hardware setup, a short pilot study is described which was arranged in order to test the applicability of two stimuli in combination with the think-aloud methods. Out of the pilot study's findings, it was determined to use the stimulus, which is less complex for the actual experiment. By use of further insights gained during carrying out the pilot study, the actual experimental setup was sharpened.

Hardware Setup

The hardware setup shown in Fig. 1 is the same as in [12], using the binocular remote eye tracker SMI RED 250 with a 22″ monitor with a resolution of 1,680 × 1,050 pixels for displaying the stimuli. The audio track during the experiments with concurrent think-aloud is recorded by a webcam, which is directed to the subject and records a video. For recording the retrospective comments on the scanpath, a screen-recording software is used to record the subject's audio together

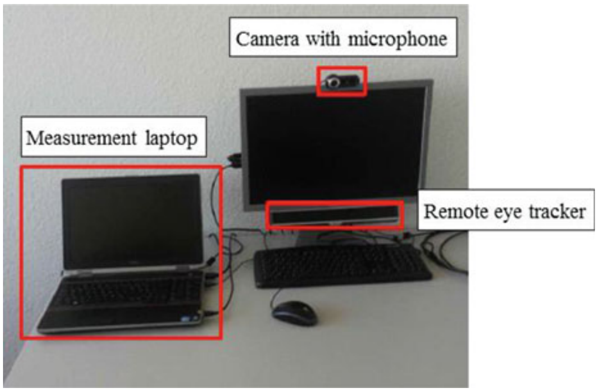


Fig. 1 Hardware setup with binocular eye tracker SMI RED 250

with the replay of the recorded scanpath. The experimenter executes and controls the experiment by use of the measurement laptop.

Pilot Study and Research Question

The pilot study was conducted in order to identify possible interfering variables and to assess and improve the experimental setup. During the experiment, two different technical drawings were used as stimuli, which were shown to four subjects. Each subject performed CTA on one of the stimuli and RTA on the other, while the order of the think-aloud methods was changed for two subjects. For CTA the subjects were given an introduction how to conduct the think-aloud method and tested the method by performing a short think-aloud exercise as proposed in [13].

The participants who first performed the RTA experiment were given the think-aloud instructions after the first experiment. The participants who started with the concurrent thinking-aloud were told that they did not need to think aloud in the second part. In addition, they were interviewed afterwards.

The main results of the pilot study concern the design of the experiment. The exercise and introduction into think-aloud were confirmed as being very helpful. During the interviews all participants said, that they felt influenced by thinking aloud concurrently, either in a positive or a negative way. Using two different stimuli for each participant leads to problems examining the influence of the think-aloud methods, as the influence on the gaze behaviour is too high. A difference between the recorded eye movements of the two methods, if existing, is not observable. The task dependent influence of the differing two stimuli was too high. Another disadvantage of using two stimuli with CTA and RTA is the time factor. Especially when starting with RTA, it is time consuming to load the gaze data with the analysis software in order to display the gaze path of the subject. As one of the stimuli was more complex than the other, loading the gaze data of this stimulus took even longer. Thus, it was decided to use the less complex technical drawing for the experimental setup of the study.

Test Persons and Experimental Design

The study was conducted with 15 participants, all of them studying mechanical engineering at the KIT and finished with the second or higher academic year. Comparable background knowledge on analyzing technical systems represented by technical drawings is ensured. Four students finished the second year, nine students finished the third year and two students the fourth year. The experimenter carried out the assignment of the participants into the two groups of CTA and RTA randomly. The two students who finished the forth academic year were assigned each to the CTA and the RTA group. Thus, eight subjects conducted CTA while seven subjects conducted RTA.

As explained before, the task was to approve a technical drawing. The check for functionality and correctness of a drawing is essential in product development. As stimulus, the drawing of a gearbox in Fig. 2 was chosen. In order to expose the subjects' behavior while they were building of an understanding of the function-shape-relations, the drawing includes a functional error. There are no drawing mistakes as wrong presentations of bearings or gearwheels, but the system is not fulfilling its actual function: the transmission of torque and rotational speed.

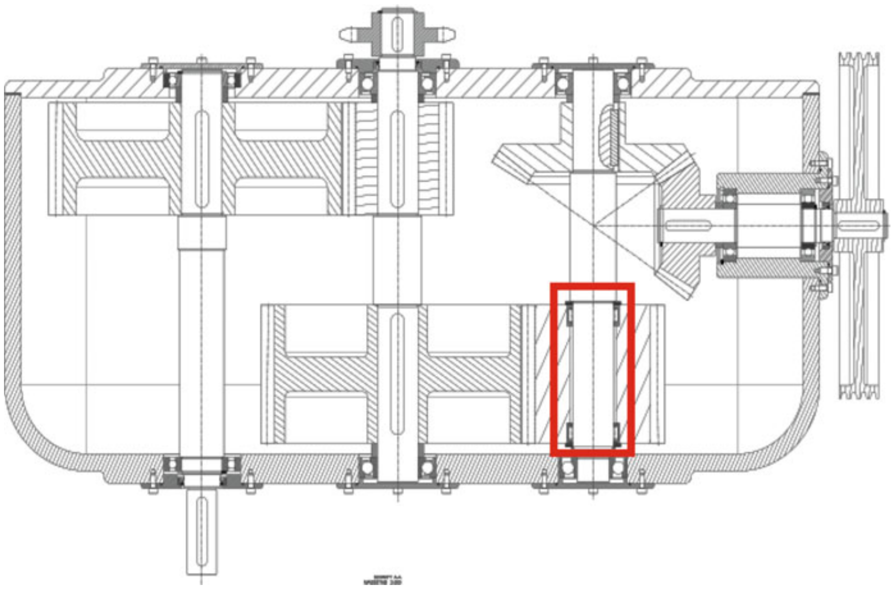


Fig. 2 Technical drawing of the gearbox with a functional error (*red rectangle*)

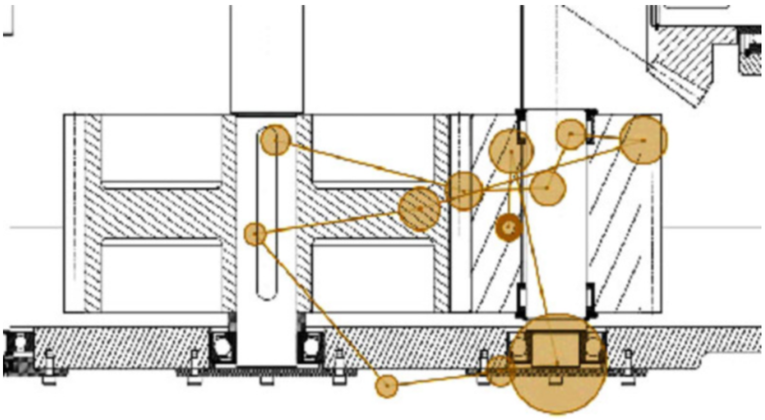


Fig. 3 Scanpath of P07 after 30 s, looking on the error without recognizing

Marked with a red rectangle in Fig. 2, the gear wheel on the bottom right is mounted on two needle roller bearings instead of being connected with a shaft-hub-joint. Thus, it cannot transmit any torque from or to the shaft. It is a substantial functional error, which the subjects should detect for the approval of the drawing.

One group of participants (CTA) was instructed to think aloud concurrently while performing the task. This instruction was left out for the RTA group. After analyzing the technical drawing the RTA group was shown a dynamic replay of their own scanpath, which the subjects were asked to comment on. The scanpath is a representation of the gaze behavior consisting of circles with varying radius, representing the fixations and their durations. The lines connecting them represent the saccades [13]. To maintain a clear view only the events of the previous 3.5 s were displayed (see Fig. 3). After the recordings, all subjects completed a questionnaire on their performed think-aloud method in order to get a direct feedback and their perception on the method.

Analysis and Results

As different kinds of data are recorded during the experiment, different methods are used to analyze the data. Not only the recorded eye movements are evaluated and interpreted. They are linked to the verbal data in order to refer to differences between both think-aloud methods. Additionally, the completed questionnaires are charted and evaluated. Because of a poor tracking rate or missing audio data, the results of subjects P02 und P10 (RTA) and P12 and P14 (CTA) cannot be taken into account for the analysis. A high tracking rate of the eye movements and an audio recording of good quality characterizes all other subjects.

Table 1 Details of subject P07

Subject	Age	Sex	Semester	Task	Words per minute in average
P07	23	male	6	CTA	44

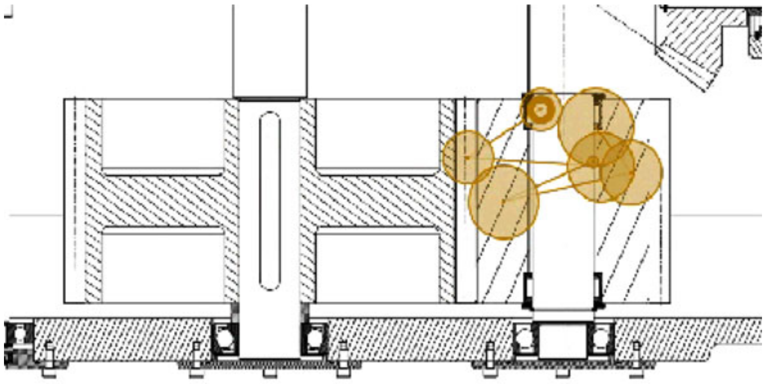


Fig. 4 Scanpath of P07 after 167 s, recognizing the error

Exemplary Detailed Analysis of the Gaze Data

For better understanding, the presented gaze data and the results of the analysis one experiment for each think-aloud method is presented in detail below. As shown in Table 1, the participant P07 solved the task while thinking aloud concurrently.

During the first 57 s he gets a general idea on the overall system to understand how it works. While looking around he names single parts of the system, e.g. "*There is a pair of gear wheels*" or "*… and an input shaft*". After 1 min, he announces to look at the bearings as shaft support: "*The bearings*, *this is where I am going to look at next*." and so he did in the following. Again he names single parts of the system as well as what kind of bearings are used: "*In the middle… there is an adjusted bearing assembly in X-arrangement*." As he cannot find any mistakes, he moves on to check the shaft-hub joints and recognizes the error after 165 s: "*Ah*! … *Here is a bearing*, *too*." Although the participant starts his search with skimming – very randomly and without any structure – he develops a plan for his further course of actions while getting a long overview, follows his plan and thereby locates the error.

The following two figures, Figs. 3 and 4, show that the additional verbal data of the concurrent think-aloud is necessary in order to interpret the scanpath correctly. Figure 3 shows the scanpath during the phase of overview. The subject looks at the gear wheels and the bearings which cause the error. Based on the fixations of the scanpath the experimenter could assume that the subject identified the error during the first 30 s. However, the subject actually did not realize that there is an error. The corresponding think-aloud data is: “*Here is a gear pair* (*short pause*) *and a transmission*.”

When recognizing the error after 165 s, he stated: “*Ah*! … *Here is a bearing*, *too*.” The scanpath in Fig. 4 shows that he scrutinized the bearing and the error. However, it is difficult to distinguish the different situations, not seeing the error and analyzing the error, by only recording the gaze data without the additional information of the thoughts.

Task Performance

Six of eight subjects, who performed concurrent think-aloud, found the error of the technical system and needed an average of 109 s for solving the task. From the retrospective think-aloud group, four of seven subjects found the error after an average of 115 s. The remaining participants declared the system to be functional after 121 s (CTA) and 137 s (RTA) on average. Accordingly, the CTA-group performed slightly better, however the variation of results and thus the standard deviations are high (57.9 s for CTA and 42.0 s for RTA).

Quantitative Analysis of Eye Movements

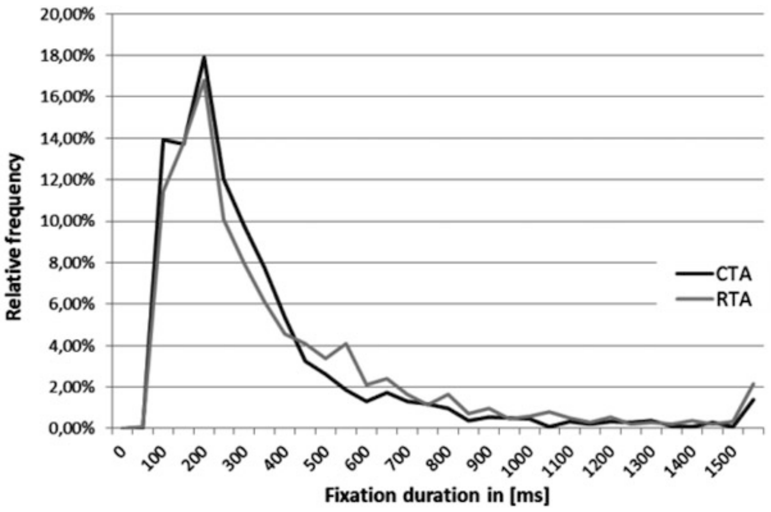
The important factors, the fixations and saccades are examined for the quantitative analysis and comparison of concurrent and retrospective think-aloud.

Fixation Duration

One hypothesis investigated with this study derives from the findings of Van Someren et al. [6] and claims that the group performing CTA shows longer fixation durations, as the subjects have to extract visual information and talk at the same time. Therefore, the distribution of fixation duration was compared between both groups.

For this purpose the durations in milliseconds are divided into bins with a size of 50 ms. Out of this, the relative frequency distribution is calculated for each subject and then averaged for both groups. The resulting chart in Fig. 5 shows the distribution of each method. The rise of both curves at the right arises from the fact, that the last point concludes all fixations lasting longer than 1,500 ms. Both curves are of the same shape and vary significantly. The calculated correlation coefficient is 0.988.

The average fixation duration of the CTA-group is 314.3 ms with the standard deviation of 88.4 ms. For the RTA-group the average fixation duration is 369.6 ms with the standard deviation of 74.2 ms. With a p-Value of 0.123, the fixation duration of CTA is not significantly lower as for the RTA-group. The lower average duration of fixations of the CTA-group does not fit to the conclusion of Van



Think-aloud method	Mean value [ms]	Standard deviation [ms]	Variance	p-Value
CTA	314.3	88.4	7,815.7	0.123
RTA	369.6	74.2	5,508.1	

Fig. 5 Relative frequency distribution of the averaged fixation duration

Someren et al. [6] that concurrently verbalizing thought slows down the eye movements. However, due to the high p-Value the conclusion of Van Someren et al. cannot be disproved for this particular experiment.

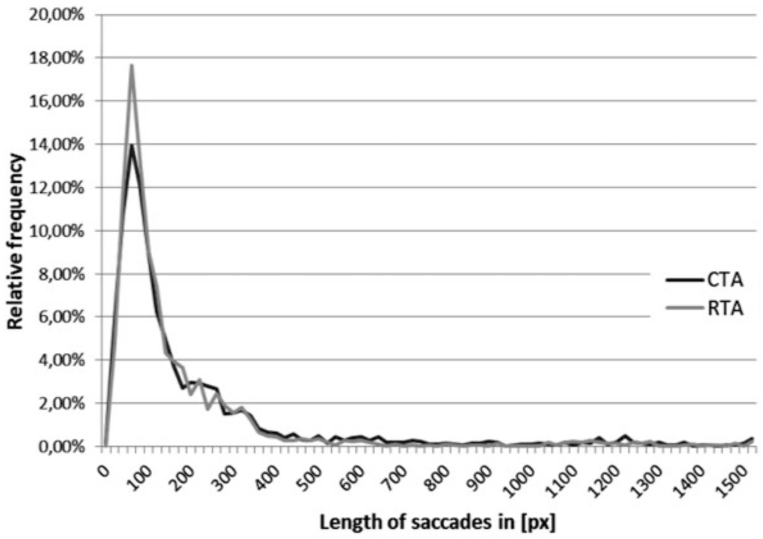
Thus, it can be concluded that the think-aloud method has no high significant influence on the fixation duration. No conclusion concerning an overload of the cognitive capacity of the subjects can be drawn for this study.

Saccade Length

The spatial length of saccades is compared in a similar way. A higher cognitive load could cause shorter length of saccades. Furthermore, shorter saccades could suggest a more thoroughly analysis and scrutinizing, as explained by Lohmeyer et al. [14].

First, the saccade length in pixels is calculated from the start and end coordinates of the saccades, given in an x-y-coordinate system. These values are then again divided into bins with a size of 20 px. The relative frequency distribution is calculated for each subject taken in account, averaged within both groups and displayed in Fig. 6. Again, both curves display the same shape and vary very little. This is also proven by the correlation coefficient of 0.986.

The average saccade length for CTA is 195.2 px with the standard deviation of 70.5 px. For the RTA-group, the average saccade length is 166.3 px with the



Think-aloud method	Mean value [px]	Standard deviation [px]	Variance	p-Value
CTA	195.2	70.5	4,975.7	0.175
RTA	166.3	29.6	877.5	

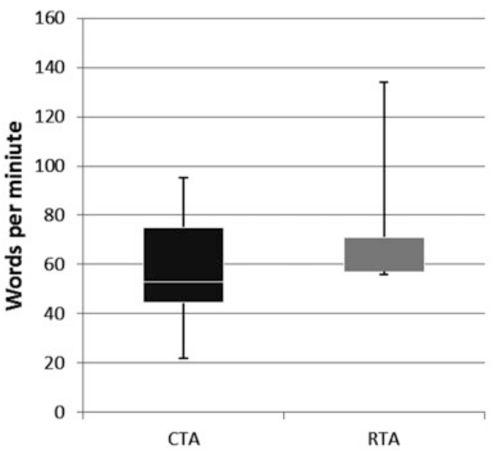
Fig. 6 Relative frequency distribution of averaged saccade lengths

standard deviation of 29.6 px. The p-Value of 0.175 shows that the saccade lengths for conducting concurrent think-aloud are not significantly higher than conduction RTA. Because the experiment shows that the length of saccades is higher for CTA than for RTA it can be argued that the cognitive load during CTA was not higher for the subjects.

In comparison the two important eye movement events for static stimuli show tendencies to longer fixation durations during a quite experiment and longer saccade length for CTA. However, no significant difference between the two groups and hence no influence of thinking aloud concurrently on the basic gaze behavior can be elicited out of the recorded data.

Verbal Data

The amount of spoken words during the recordings varies strongly between the participants. Following diagram in Fig. 7 shows that the subjects performing RTA spoke few more words per minute than subjects who concurrently spoke out their thoughts. However, the subjects who commented on their scanpath retrospectively in total spoke more words because the audio recording was not limited to the length



Think-aloud method	Mean value [min^{-1}]	Standard deviation [min^{-1}]	Variance	p-Value
CTA	57.8	26.43	698.6	0.191
RTA	75.0	33.56	1,126.5	

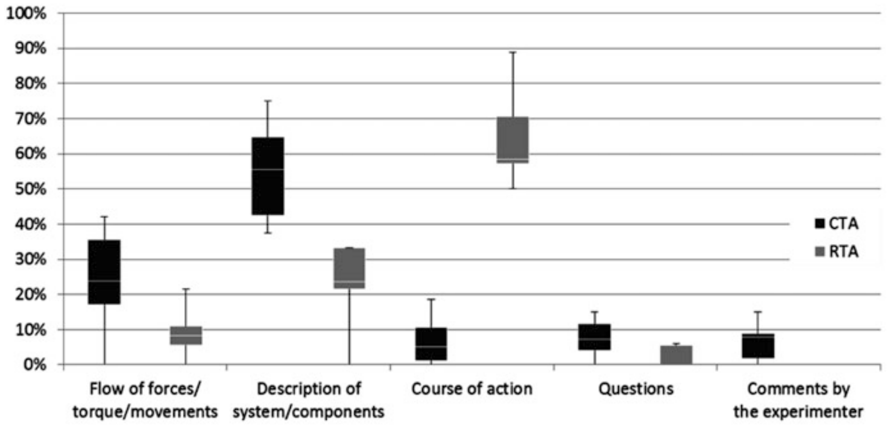
Fig. 7 Spoken words per minute during the experiments

of the scanpath video. The average words per minute for CTA are 57.8 with a standard deviation of 26.43. For RTA the average words per minute are 75 with a standard deviation of 33.56. With the p-Value of 0.191 the difference of spoken words per minute by the CTA- and the RTA-group is of low significance.

In order to investigate the content of the verbal data the participants' statements are assigned to coding schemes. Holmqvist et al. [13] suggests adapting the coding of the verbal data to the given task. Due to the analysis activity of the design process and after scanning all recorded videos, the following five schemes are determined:

- Comments on the functional level – flow of force, torque or movements, e.g.: "*Here the torque is transmitted to the other shaft*."
- Comments on the shape/form level – naming of components or descriptions of the technical system, e.g.: "*Here is another pair of gear wheels*."
- Comments on the process level – course of action, e.g.: "*Then I checked if there could be an error somewhere outside the system*, *as a mounting error*."
- Question asked during the recording, e.g.: "*Can force be transmitted here*?"
- Comments by the experimenter, e.g. "*Please continue speaking*."

If a statement fits into more than one scheme it is assigned to both schemes. In order to be able to compare both think-aloud groups the ratio of comments to the total amount of comments from each group is calculated. For the RTA group only the comments during the retrospective recording are taken into account, but not the ones made during the task performing. As can be seen in Fig. 8 the distribution of the comments varies strongly between CTA and RTA.



Think-aloud method		Flow of forces/ torque/ movements [%]	Description of sys-tem/ components [%]	Course of action [%]
CTA	Mean value	24.12	55.0	7.0
	Standard deviation	15.6	15.03	7.56
	Variance	243.4	226.0	57.2
RTA	Mean value	9.2	22.2	65.0
	Standard deviation	7.73	13.52	15.41
	Variance	59.7	182.7	237.5
p-Value		0.0375	0.002	0.000

Fig. 8 Distribution of the think-aloud comments over the coding schemes

The figure illustrates the main difference very well. For all three main schemes – the flow of forces/torque/movement, descriptions of the systems/components and the course of actions – the p-Value is low. Thus, the content of verbalization differ significantly between both think-aloud approaches. While thinking aloud concurrently participants mainly describe and comment on what they see resulting in more naming of single components or parts of the technical system and descriptions of the flow of force, flow of torque or movements. Thinking aloud retrospectively on the other hand leads to significantly more comments about the own course of actions. The difference in the last two groups refers to the fact mentioned above, that for the RTA group only the retrospective recordings are taken into account. Questions on the technical drawing mainly came up while performing the task. Thus, the experimenter did not need to comment during the experiment.

For the combination of eye tracking and think-aloud these results help to choose the right combination for the intended purpose. For the aim of study to observe the perception of engineer designers for function-shape-relations, the concurrent think-aloud method is preferable. It has the potential to reveal the fact if the subject actually perceived information or if the subject only looked at the object without perceiving any information. Experiments which aim on examining engineers' different procedures, e.g. for analyzing technical systems, the retrospective think aloud method is the appropriate choice. It provokes the subjects to comment their own gaze path and thus reflect their preceding course of action.

Results of the Questionnaire

Subsequently to the experiments the experimenter interviewed the participants on their personal impressions and conclusion on the think aloud methods. The questionnaires are set up differently for the two think-aloud groups. For the CTA-subjects, it comprises the following questions:

- How comfortable did you feel when thinking aloud?
- How much did the think-aloud influence your course of action?

For the subjects retrospectively commenting their scanpath the questions are adapted in the following way:

- How comfortable did you feel commenting your own gaze path?
- How well could you remember your own course of action?

Figure 9 shows the answers to the first questions on how well the participants felt while thinking aloud concurrently or retrospectively. The varying evaluation highly depends on the subject's preferences.

Only one subjects out of each group felt strongly uncomfortable during the experiment. However, half of the subjects felt well with thinking aloud while the other half did not. A general difference between CTA and RTA cannot be detected.

The CTA group was also asked how strong the thinking aloud approach influenced their course of actions. Figure 10 shows that everyone felt influenced, but no one felt a huge influence during CTA. The second question to the RTA group was how good their memories of their own course of action were during the retrospective recording. As those recordings took place only a few minutes after

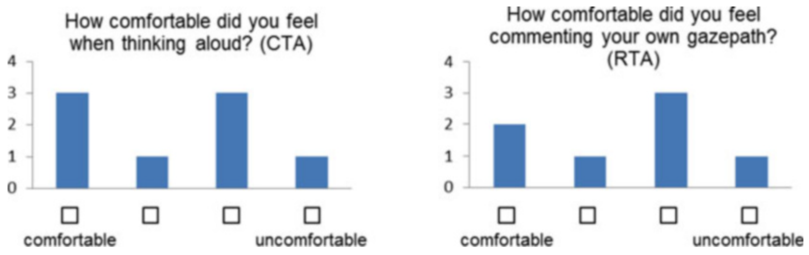
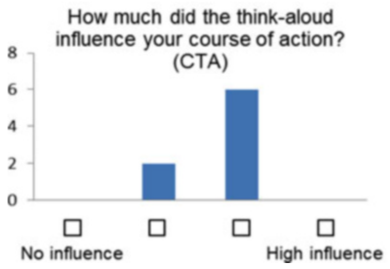


Fig. 9 Level of comfortableness during CTA (*left*) and RTA (*right*)



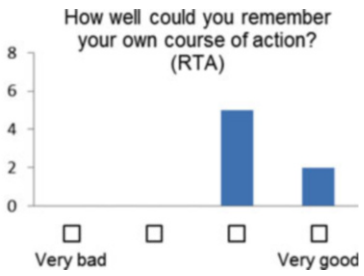


Fig. 10 Influence of CTA (*left*)/remembering course of action with RTA (*right*)

performing the task, no subject had problems remembering and reflecting on the own course of action.

The results of the questionnaire also suggest using RTA for studies on the procedure of engineers. Concurrently thinking aloud could influence the course of action and thus, the gaze path as well. However, longer experiments can cause gaps in retrospective protocols. The gaze video cued approach presents beneficial support to this problem. If lapse of memory still occur, a thorough consideration of both think-aloud methods is needed.

Discussion and Conclusion

In order to understand the engineers behavior during designing it is essential to understand how the human builds up function-shape-relations. With eye tracking it is possible to observe with shapes and forms the designer looks at in order to comprehend the complex correlations in technical systems. In order to analyze the perception of the seen objects, additional information is needed. Thus, this paper investigated the combination of eye tracking and think-aloud methods.

As discussed both compared think-aloud methods, concurrent think-aloud and gaze video cued retrospective think-aloud, have strengths in combination with eye tracking and can be applied for different use cases and research objectives.

The first research question of this paper, if concurrent think-aloud does affect the eye movements of the subjects while they analyze a technical system in comparison to not speaking, can be negated. The two important indicators are the duration of fixations and the length of saccades. For this study, both indicators do not significantly vary between CTA and a quite experiment. Thus, the results cannot verify the conclusion of Van Someren [6] that concurrently thinking aloud slows down eye movements.

The second research question “Do concurrent think-aloud and retrospective think-aloud generate the same set of additional information and perspectives for eye tracking experiments?” is analyzed with coding schemes on the functional, the shape/form and the process level. The produced verbal data of the think-aloud methods significantly differ in their content. The concurrent think-aloud method is recommendable for studies on the observation of the functional context, which is also identified by Gero [11]. CTA elicits more information on the functional level (e.g. flow of forces, torque and movements) as well as on the shape/form level with descriptions of the technical systems. Thus, combining CTA and the recording of eye movements reveals the possibility to examine the perception of function-shape-relations by engineers.

RTA is the adequate choice for observing the engineers' procedure during the design process. It reveals most information on the process level. The analysis of the verbal data shows a deeper reflection on the subjects' courses of action, which strengthens the Van Gog's observations [9]. Using the recorded gaze path for cued retrospective think-aloud provokes even more comments on the proceedings of engineers.

It needs to be pointed out that the analysis of the eye movements and verbal data considers only 11 subjects. The recordings of the other four participants are not taken into account due to bad tracking rate or missing audio track. The effect of CTA on the eye movements needs to be verified by conducting a study with a greater amount of subjects in order to be able to state valid conclusions. In order to compare both think-aloud methods relating to the subjects' performance, each subject would need to conduct both methods as applied in the pilot study. However, with the change of the stimulus new deviations are caused.

Combining eye tracking and think-aloud approaches is very useful for research on the engineer's behavior while analyzing technical systems. Both think-aloud methods have the qualification on revealing relevant additional information to the recorded eye movements. Further experimental setups of design studies, inclusive eye tracking experiments, can build on the findings of this paper. The findings of this research work helps to choose the appropriate think-aloud method in combination with eye tracking for setting up new empirical experiments for analyzing the engineer designer's behavior in his/her natural environment.

References

1. Matthiesen S (2011) Seven years of product development in Industry – experiences and requirements for supporting engineering design with 'Thinking Tools'. Proceedings of the International Conference on Engineering Design, Copenhagen, 236–245
2. Meboldt M, Matthiesen S, Lohmeyer Q (2012) The dilemma of managing iterations in time-to-market development processes. International Workshop on Modelling and Management of Engineering Processes, Cambridge, UK
3. Ahmed S (2007) Empirical research in engineering practice. J Des Res 6:359–380
4. Ericsson KA, Simon HA (1993) Protocol analysis. MIT Press, Cambridge, MA
5. Watson JB (1920) Is thinking merely the action of language mechanisms? Br J Psychol 11:87–104
6. Van Someren MW, Barnard YF, Sandberg JAC (1994) The think aloud method: a practical guide to modelling cognitive processes. Academic, Amsterdam
7. Davies SP (1995) Effects of concurrent verbalization on design problem solving. Des Stud 16:102–116

8. Ryan B, Haslegrave CM (2007) Use of concurrent and retrospective verbal protocols to investigate workers' thoughts during a manual-handling task. Appl Ergon 38:177–190
9. Van Gog T, Paas F, van Merriënboer JJG, Witte P (2005) Uncovering the problem-solving process: cued retrospective reporting versus concurrent and retrospective reporting. J Exp Psychol Appl 11:237–244
10. Kuusela H, Paul P (2000) A comparison of concurrent and retrospective verbal protocol analysis. Am J Psychol 113:387–404
11. Gero JS, Tang H (2001) The differences between retrospective and concurrent protocols in revealing the process-oriented aspects of the design process. Des Stud 22:283–295
12. Matthiesen S, Meboldt M, Ruckpaul A, Mussgnug M (2013) Eye tracking, a method for engineering design research on engineers' behavior while analyzing technical systems. Proceedings of the International Conference on Engineering Design, Seoul, 277–286
13. Holmqvist K, Nyström M, Andersson R, Dewhurst R, Jarodzka H, van de Weijer J (2001) Eye tracking. Oxford University Press, Oxford
14. Lohmeyer Q, Matthiesen S, Mussgnug M, Meboldt M (2014) Analysing visual behaviour in engineering design by eye tracking experiments. Proceedings of TMCE 2014 (accepted)

An OTSM-TRIZ Based Framework Towards the Computer-Aided Identification of Cognitive Processes in Design Protocols

Niccolò Becattini, Gaetano Cascini, and Federico Rotini

Abstract This paper presents an original approach for the analysis of design protocols. It presents a coding scheme based on the OTSM-TRIZ Network of Problems, so as to map the protocols by both considering design moves and their mutual relationships with links. Stemming from this coding scheme, the authors propose a novel set of rules for the identification of cognitive processes to be integrated into a Computer-aided tool, so as to reduce the time employed for the analysis of design protocols. The five preliminary tests carried out to verify the potentiality and the feasibility of the approach have demonstrated that it provides good results both in terms of quality and in terms of time-savings for the whole protocol analysis.

Introduction

Whatever the specific reason is that drives a study about the cognitive processes of designing people, it is overall aiming at better understanding how people creatively think and what influences their creativity. On these bases, the design research defines and refines theories, methods and tools with the purpose of supporting the designers to have more effective ideas with more efficient methods. Design cognition of individuals and groups has been studied under a wide range of different conditions and with very different profiles of designer [1, 2]. In this context, the search for regularities or cognitive patterns, which better drive to suitable and novel solutions, plays a paramount role for the definition of guidelines, preferable logical paths, stimuli as well as techniques to prevent undesired effects as fixation. Computers, in this perspective, may play a double role. On the one hand, they can support the analysis of design protocols by reducing the efforts of the analysts. On

N. Becattini (✉) • G. Cascini
Politecnico di Milano, Milano (Milan), Italy
e-mail: niccolo.becattini@kaemart.it; Gaetano.cascini@polimi.it

F. Rotini
Università di Firenze, Firenze (Florence), Italy

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_6

the other hand, it is also possible to exploit their high performances in processing information, so as to better identify the presence of emergent behaviours that are not clearly visible with a simple human-driven analysis. Stemming from these premises, the paper aims at introducing a set of computable rules to interpret the design moves in a protocol, so as to codify them according to specific cognitive processes.

Besides this introduction, the paper is organized into five sections. The first introduces protocol analyses, together with some references to the recent introduction of computerized means. The second section describes an adaption of OTSM-TRIZ Network of Problems, as a graphical model to map design protocols. The third and fourth sections respectively present the original set of computable rules to further codify the cognitive processes along the protocol and their application, so as to verify the feasibility, the benefits and the pitfalls of the proposed approach. The concluding section sums up the results of this approach and envisions further directions of development and their potential impact on the analysis of cognitive processes in design.

Design Protocol Analysis: Criteria and Supporting Tools

The analysis of design protocols aims at capturing and understanding the behaviour of people dealing with specific design tasks or, more in general, design problems. It consists in the exploratory search for regularities or differences in the design path followed by designers having different experiences, different backgrounds, as well as different genders or age. By shedding light on the cognitive processes characterizing the design activity, it is thus possible to recognize the best practices and improve prescriptions and guidelines, towards an increased effectiveness and efficiency of the whole design process.

This approach has already shown a good potential in several applications, such as: comparing the effects on the reasoning of users of different design methods [3]; showing different logical patterns between experienced and non-experienced designers [4]; the effects of sketching on the idea generation process [5] and so forth. One of the most interesting results emerged by these studies concerns the characteristic process followed by more experienced designers. If compared to the structured sequence of steps of some prescriptive design methods, experienced designers tend to skip some steps without following a systematic logic. In other words, they move back and forth between the problem and the solution space in a co-evolutionary process that aims at finding a match between the two [6]. Some literature reviews have shown the growing interest in the field of the analysis of design protocols. In 2009 [7], Jiang and Yen showed the distribution of studies carried out with these approaches, being think-aloud protocols (an individual describing his/her thoughts) or conversational ones (a team discussing on the design choices). Conversely, in 2011, Chai and Xiao [8] have pointed out the increased efforts in carrying out these studies. Pouhramadi and Gero [9] identified the following seven phases as necessary to carry out a complete protocol analysis:

1. Definition of a coding scheme
2. Recording the activity of designers
3. Transcription of the recordings
4. Segmentation of the design discourse, according to the coding scheme
5. Analysis of coded protocols
6. Definition of links between design steps
7. Analysis of the graph of links among design moves (linkography).

The above numbered list clearly shows that it is necessary to define a-priori a coding scheme. Such schemes allow organizing the results according to variables and parameters in order to characterize the elementary design moves described during think-aloud protocols or conversational ones. For instance, Dekoninck et al. [10] have developed a coding scheme, based on creative 'modes of change', to show the strategies that are used by a creative designer to skip from one 'train of solutions' to new avenues. Among the many available alternatives, the FBS framework [11] has been frequently used to map design moves.

Moreover, the need of recording and transcribing the design, as for steps 2 and 3, highlights that the protocols cannot be reliably built with an a-posteriori approach (designers trying to reconstruct their reasoning after the conclusion of the design session), because of the potential loss of relevant information. In addition, steps 6 and 7 point out that the analysis should also take into account the presence of links (e.g. by causal reasoning, analogy, etc.) between the design moves characterizing the protocol.

Since its first formulation, about two decades ago [12], the approach aimed at analysing design moves and their mutual relationships (linkography). This approach has shown a good potential in capturing important key elements that haven't been previously highlighted (e.g. the most critical design moves are those which have a higher number of links in the linkography graph [13]).

Moreover, from the preliminary studies on linkography, the definition of links among the design moves has become more and more important to highlight potential cognitive patterns. Van der Lugt, for example, in [14] proposed a more fine-grained characterization of links that has further enriched the range of potential coding schemes. According to his vision, the links have to be characterized by the distance from the previous design move. A relatively small change in the definition of the idea corresponds to a supplementary link. On the contrary, a radical modification of the ideas between two adjacent design moves shows that the stream of thought has drifted into a tangential direction (tangential links). Intermediate situations, such as the ones where the changes are quite significant, but the cognitive process has been following the previous line of reflection, should be characterized by the so-called modification links. On a similar wavelength, the recent work by Cai et al. [15] reduced the number of alternative links to two: the so-called "transformation" from a design move to the next one can occur on lateral or vertical direction. The former shows a broadening of the design space by proposing an idea that is slightly different or alternative to the previous one. The latter, on the contrary, is related to the deepening of the design space by means of an idea that is more detailed than the previous one. A recent work by Perry and Krippendorf [16] argues that the segmentation from the transcriptions of the whole design discourse into design moves is not a repeatable process, which largely depends on the analysts and their experience. Therefore, they claimed that there exists a clear need for better and more robust definition of the coding schemes, so as to remove ambiguities and improve the reliability of design protocol studies.

Getting back to the sequence of steps to be carried out during a protocol analysis study, it is worth noticing that the phases from 3 to 7 are the most time consuming. With reference to the efforts required by this activity, Jiang and Yen also showed that the analyses require from 10 to 100 times the duration of the protocols, independently from the chosen coding scheme. From this perspective, the exploitation of computerized tools supporting the most time consuming phases of a design protocol analysis represents a good opportunity to reduce the efforts of the analysts for textually/graphically transcribing, segmenting and organizing the recordings of the protocol according to the coding scheme (modelling) and carrying out the analysis.

In a previous paper [17] the authors investigated the opportunities for speeding up the analyses of design protocols by means of computer tools. That contribution focused on presenting the opportunities due to the adoption of a computerized algorithm for supporting the cognitive processes of designers facing design task, mainly oriented towards the transformation of an ill-Structured Problem into a well-Structured one. Despite the supporting nature of the algorithm [18], its flexible structure leaves the designers a certain freedom for choosing the path for the problem analysis and the preliminary generation of design concepts. Such a computer support, therefore, goes into the direction of collecting the stream of reflections of the designers every time they introduce novel elements within the analysis or make a choice.

Moreover, the algorithmic nature of the design process allows also an a-priori structuration of the design discourse that is consistent with the coding scheme predefined together with the structure of the algorithm. However, despite this computer-based approach is helpful to reduce the analysts' efforts in phases 2–4 of a protocol analysis, the lack of full freedom for the designers shows that these systems should be further investigated, so as to start allowing a proper segmentation and coding without any kind of constraints for the cognitive processes of the designers.

On the other hand, Pourmohamadi and Gero described a computer application, called LINKOgrapher, addressed at standardizing the coding schemes of protocols. Still with reference to the above numbered list, this tool supports the analysts in carrying out the steps 5, 6 and 7 that are mainly oriented towards the analysis of the protocol, rather than on its modelling according to the chosen coding scheme.

According to these premises, the authors consider that the better exploitation of computerized means and their speed in analysing data should pass through the combined definition of an appropriate coding scheme for both design moves and their links, as well as from a set of rules, aiming at the automatic identification of cognitive processes.

The next section briefly presents the authors' proposal for modelling the design discourse according to a novel coding scheme that has already produced encouraging results. The definition of rules for the interpretation of cognitive process out of the design discourse represents, on the contrary, the novel contribution of this paper and they are deeply analysed in a dedicated section.

The Constructs of the OTSM-TRIZ Network of Problems as a Coding Scheme for Design Protocols

As mentioned at the end of previous section, the authors have already carried out some experiments to check the suitability of the OTSM-TRIZ Network of Problems at describing the design protocol of a group of people dealing with a design task [19]. These experiments have shown that the approach is effective to map the design moves and the cognitive processes of groups of designers. The Network of Problems is a design tool originally proposed by Nikolai Khomenko in the boundaries of the OTSM- TRIZ theory. Its main aim is to support designers in dealing with the initial steps of complex problems solving activity [20]. The earlier version of the Network of Problems where just roughly outlined through the coarse definition of its main constructs: "Problems" (Pb) and "Partial Solutions" (PS) to be hierarchically organized, but without any definition for the links connecting two nodes. In recent years, several applications on real case studies have also provided further evidences about the capabilities of this instrument as a problem solving tool by itself (e.g. [21]) or integrated into more articulated procedures based on TRIZ and OTSM-TRIZ to cope with inventive problems of complex nature [22].

Cavallucci et al. [23] presented a more formal and repeatable definition for the main constructs of the Network of Problems, considering both nodes and links. Moreover, the same contribution also suggested switching from the hierarchical structure towards a less organized organization of nodes, so as to better describe the essence of problems, for which a strict top-down hierarchy represents a further complication producing a small added value. However, this approach just aims at better supporting the earliest stage of the design process (such as the initial situation analysis) and it is not suitable to describe, for instance, the presence of hypothesis and conjectures, which are typical of the design processes aimed at searching new solutions. As well, the decomposition of a problem in simpler problems or the detailing of Partial Solutions described at different detail levels are not representable within this framework without a proper adaptation. To this purpose, the authors have integrated some new constructs in the Network of Problems, as presented in [19]. More in detail, the adapted Network of Problem demonstrated with practical applications to be capable of mapping all the different moves by three design teams, each of them working on two different design problems, using Problems and Partial solutions as a coding-scheme.

A summary of the constructs (both nodes and links characterizing the network) is shown in Figs. 1 and 2. There exist several typical combinations between problems and partial solutions and their connections can be determined by causal relationship, the synthesis of working principles and structure capable to address problems (completely, partially, hypothetically), as well as by the logic of detailing or decomposing an item into simpler parts.

The description of the design protocol, and thus the application of the coding scheme based on such constructs, requires also mapping the sequence of design moves by adding a label inside each node, which univocally specifies what design move each node pertains to (see Fig. 3).

The following rules are applied to codify the protocols:

- Each product attribute, functional or geometrical requirements, as well as goal should be characterized as a Problem;
- Each concept aimed at satisfying on the abovementioned attributes and requirements should be characterized as a Partial Solution;
- Each of the design move should be characterized by a numerical ID to be inserted within the node;

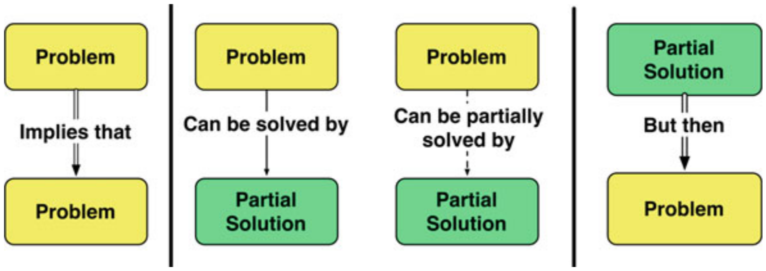


Fig. 1 Graphical representation of the constructs of the network of problems according to [23] (from *left*): causal relationship between a pair of problems; sufficient and insufficient solving links; causal relationship between a problem and a solution

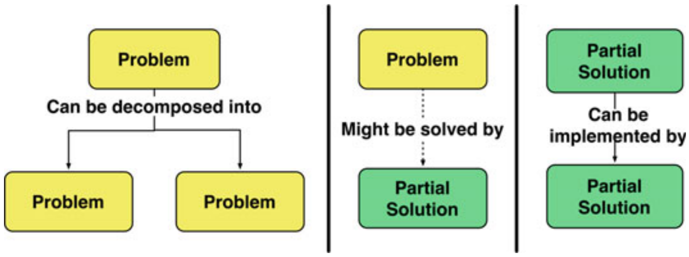


Fig. 2 Graphical representation of the novel contribution proposed in [19] to enrich the network of problems. From *left* to *right*: (**a**) Problem decomposition, (**b**) hypothesis of solution, (**c**) synthesis of a more general solution

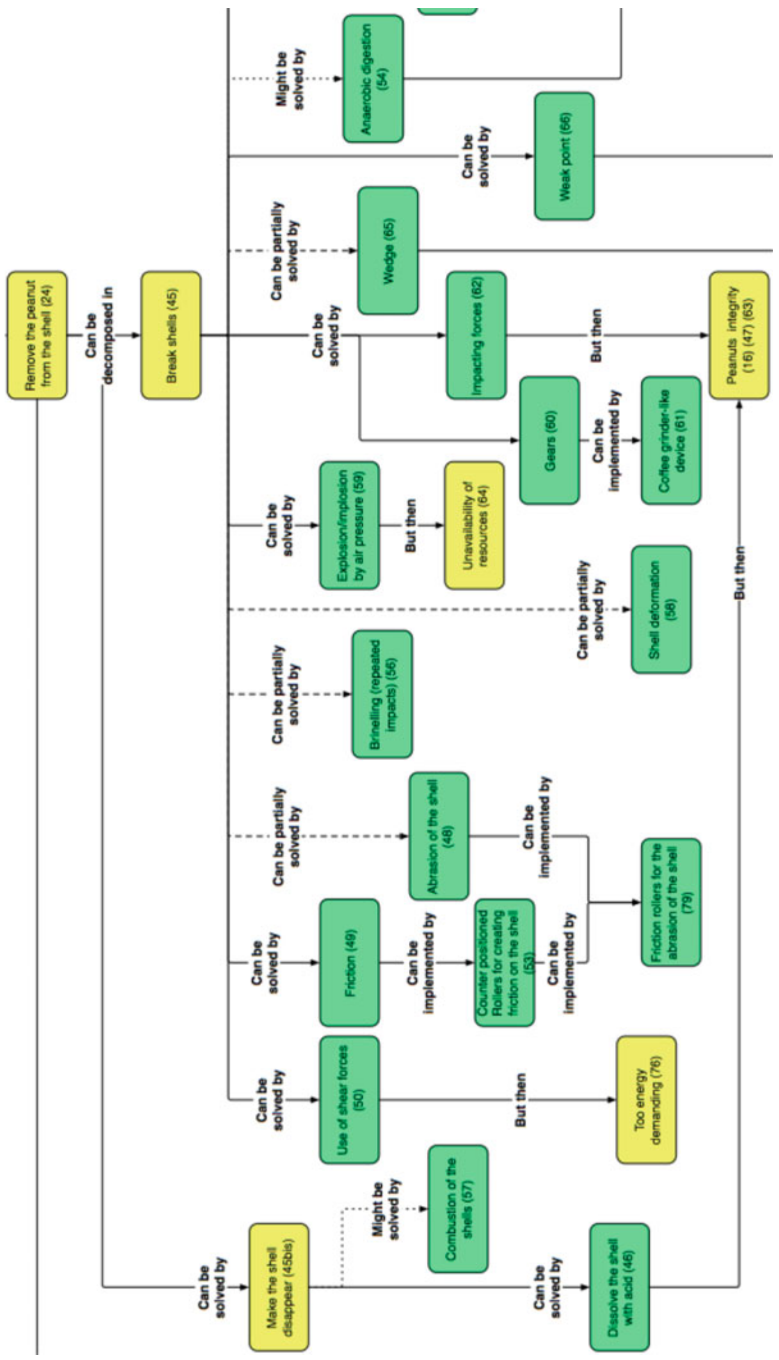

Remove the peanut from the shell (24)
Can be decomposed in
Break shells (45)
Can be solved by
Make the shell disappear (45bis)
Might be solved by
Combustion of the shells (57)
Can be solved by
Dissolve the shell with acid (46)
But then
Can be solved by
Use of shear forces (50)
But then
Too energy demanding (76)
Can be solved by
Friction (49)
Can be implemented by
Counter positioned Rollers for creating friction on the shell (53)
Can be implemented by
Friction rollers for the abrasion of the shell (79)
Can be partially solved by
Abrasion of the shell (48)
Can be implemented by
Can be partially solved by
Brinelling (repeated impacts) (56)
Can be partially solved by
Shell deformation (58)
Can be solved by
Explosion/implosion by air pressure (59)
But then
Unavailability of resources (64)
Gears (60)
Can be implemented by
Coffee grinder-like device (61)
Can be solved by
Impacting forces (62)
But then
Peanuts integrity (16) (47) (63)
Can be partially solved by
Wedge (65)
Can be solved by
Weak point (66)
Might be solved by
Anaerobic digestion (54)

Fig. 3 An excerpt of a network of problems for the design problem as for the protocol analysis carried out in [19]

- The different nodes should be linked consistently with the meaning of the constructs presented in Figs. 1 and 2 and, especially, according to the meaning described by the team of designers, as interpreted by the analyst.

The next section will present a set of rules to interpret the design protocol. Their purpose is to shift towards the computer-driven identification of cognitive process. As a result, this approach is expected to allow the protocol analysts to speed up their work. The introduction of computers should also provide a support in recognizing cognitive patterns or regularities that are not immediately inferable with standard non-aided analyses.

A Set of Rules Towards the Automatic Interpretation of Design Protocols

Since the above presented coding scheme for the design moves has proved to be practically usable to map the protocols of designing teams, the authors aim at defining a set of rules that allow a computer to process and interpret the cognitive processes characterizing a design protocol after a preliminary arbitrary coding, using Problems and Partial Solutions. To this purpose it is necessary to clarify that despite the benefits coming from the automatic execution of the process, it is preferable to allow the analysts to keep control over the application of the set of rules, so that it is possible to check the correctness of the automatically generated analyses and remove the potential ambiguities that can emerge.

Figure 4 shows the generic scheme proposed for the identification of cognitive processes by a computer. Indeed, the scheme points out that both the nodes and the links of the network are taken into account, also considering the number of connections that comes in or goes out from the "current node". The current node

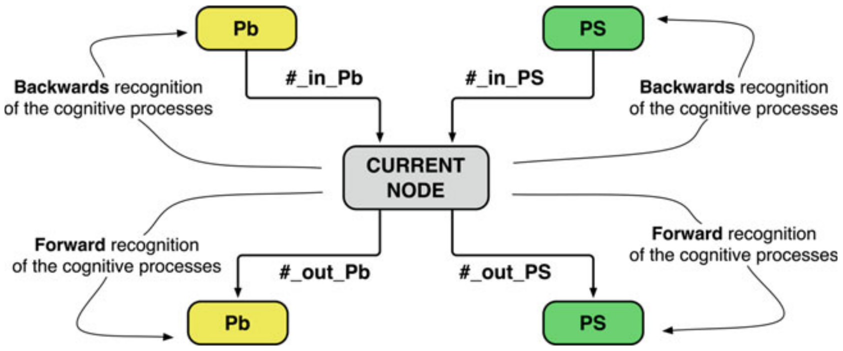

Pb
PS
#_in_Pb
#_in_PS
Backwards recognition of the cognitive processes
Backwards recognition of the cognitive processes
CURRENT NODE
Forward recognition of the cognitive processes
Forward recognition of the cognitive processes
#_out_Pb
#_out_PS
Pb
PS

Fig. 4 Scheme describing the logic for the recognition of cognitive processes in design protocols that have been segmented and whose design moves have been organized consistently with the basic constructs of a network of problems

refers to a specific design move, which can be obviously characterized as a problem or as a partial solution.

It is also worth mentioning that this scheme is proposed in a more abstract version if compared to the coding scheme presented in the previous section, since the meaning attributed to links is overlooked. This approach represents a trade-off to carry out preliminary tests about the feasibility and the reliability of the approach with reduced computer coding efforts, without missing the objective of considering the connections among design moves. With reference to the scheme of Fig. 4, Table 1 collects the set of rules to be applied within a computer system in order to identify cognitive processes out of a Network of Problems that maps the protocol of an individual or a team during a design task.

The authors have defined the rules in order to connect the potential patterns appearing on the Network of Problems to some of the most relevant cognitive processes, as they are described in literature.

Gero's original FBS framework [11] collects several cognitive processes that are relevant to this kind of investigation. Formulation, Reformulation [1] and Evaluation [7] are three of the cognitive processes explicitly mentioned in that framework. The first two cognitive processes concern with shaping the problem space, while the third one aims at verifying the suitability of an actual specific solution concept against the expectations the designer had at the beginning.

The Synthesis of solution concepts, which is a relevant cognitive process as well, still with reference to Gero's framework, is here detailed by means of more than one cognitive process. For instance, Exploration ([4], with reference to Table 1) and Combination [5] are two of the main cognitive processes that Boden [24] considered as the ones distinguishing creativity. The former searches for potential

Table 1 The novel set of rules towards the automatic recognition of cognitive processes out of a design protocol, as it can be implemented in a computer

ID	Cognitive process to be recognized	Current node	Direction of the analysis	Condition to be verified
1	(Re) formulation	Problem	Backwards	#_in_Pb = 1
			Forward	#_out_Pb = 1
2	Decomposition	Problem	Forward	#_out_Pb > 1
3	Problem aggregation	Problem	Backwards	#_in_Pb > 1
4	Exploration	Problem	Forward	#_out_PS > 0
		Solution	Backwards	#_in_Pb > 0
			Forward	#_out_PS > 0
5	Combination	Solution	Backwards	#_in_PS > 1
6	Detailing	Solution	Backwards	#_in_PS = 1
			Forward	#_out_PS = 1
7	Evaluation	Problem	Backwards	#_in_PS > 1
		Solution	Forward	#_out_Pb > = 1
8	Co-evolutionary reasoning	Solution	Backwards (*) and Forward (**)	#_in_Pb = 1 (*) and #_out_Pb = 1 (**)

solutions that are novel and capable of solving an initial issue through generating ideas in a divergent way. The latter, in turn, deals with the synthesis of a new solution concept stemming from two already existing ones. Moreover, even Detailing [6] a solution represents a step towards the Synthesis of more refined solution concepts.

Still with reference to the exploration of the whole design space, the authors also consider relevant the investigation of the design space that may occur when the designers just considers problems. This is the motivation for the authors to introduce the Decomposition [2] of problems, as a process that may be relevant for better understanding the width of a certain investigation for the generation of solution concepts. The assumption is that wider is the set of alternative problems from which a solution concept has been generated and wider could be the set of generated solutions. On the contrary, it is also possible that several problems can lead to the same consequences. This may happen whenever the designer or the designing team consider more effective to face a unique problem that deals with the different implications triggered by several problems, rather than addressing all the different causes one by one. These situations are captured as a cognitive process of Problem Aggregation [3]. At last, the potential emergence of the Co-evolutionary [8] way of reasoning may arise every time the stream of thought connects a generated solution to an existing problem and to a new problem triggered by the solution itself.

As for the scheme of Fig. 4, the structure of the network, together with the characterization of a node with the number of incoming and outgoing arrows may help to understand what are the cognitive processes that lead towards a suitable solution in the network (Partial solution nodes for which #_out_Pb = 0 or #_out_PS = 0). With a similar logic, it is also possible to distinguish which are the problems that, even if recognized, haven't been addressed by the designer or the design team (Problem nodes for which #_out_Pb = 0 or #_out_PS = 0). As well, it is possible to calculate an index for describing the overall efficiency of the design process through a ratio between the number of suitable solution concepts, as for the above definition, and the overall number of solution nodes generated along the protocol.

At last the organization of the protocol into numbered design moves, which can be referred to both Problem and Partial solution, also allows mapping the transitions between the problem and the solution space and vice versa, so as to shed further lights on the co-evolutionary processes.

Still for what concerns the scheme of Fig. 4, Table 1 represents just a simplified set of rules, so as to comply with the need of testing the feasibility and the reliability of the approach with reduced computing efforts. An aprioristic analysis of the rules before their application shows clearly that some potential conflicts may arise during the automatic recognition of the cognitive processes.

For instance, one of the conditions to be verified for the recognition of Detailing processes (considering the links oriented forward) conflicts with one of the conditions for recognizing Exploration (still considering the links oriented forward).

Table 2 Computational steps for the identification of cognitive processes

Cognitive process to be recognized (ID)	Computational steps
1–7	1. Current node: [problem, solution]
	2. Condition to be verified: [#_out(in)_Pb(PS) $\geq$ X]
8	1. Current node: [solution]
	2. Condition #1 to be verified: [#_in_Pb = 1]
	3. Condition #2 to be verified: [#_out_Pb = 1]

According to the set of rules of Table 1 and the scheme of Fig. 4, the recognition of cognitive patterns can be carried out by considering one single node of the network a time:

- Together with its incoming or outgoing connections from other nodes (cognitive processes 1–7); or
- Both the incoming and the outgoing connections (cognitive process 8).

Moreover, the overall logic to be followed with computational steps for the recognition of cognitive processes is presented in Table 2.

Considering the simplified computational logic of Table 2, it is clear that the same node on the network may satisfy more than a condition at time, thus resulting in further conflicts of attribution.

To this purpose, by means of the results obtained through a relatively simple computer implementation of the abovementioned rules, the next section aims at estimating the potential and the limitations of this approach as well as the benefits it can generate.

Results Obtained Through a Computer-Aided System for Design Protocol Analysis

In order to apply the set of rules presented in the previous section, the authors took into consideration the existing protocols that have been already codified according to the constructs of the Network of Problems. They consist of the design activities of three teams working on one or two problems (totally five networks). The problems considered along these analyses are quite known in the field of design protocol analysis:

- Design a machine to shell the peanuts to be used in agricultural contexts [25];
- Design a device to carry a backpack whenever the user is riding a mountain bike [1].

The tests have been carried out with small groups of students (three to four people) of an engineering design course within a MS in Mechanical Engineering. More information about the way the tests have been carried out are available in [19].

Such results have been carefully reorganized from graphs to spreadsheet tables, introducing one design move (node of the network) per row and by collecting the exact amount of incoming and outgoing arrows from both problems and partial solutions for each node. Furthermore, given the exploratory nature of this study, the authors simplified the analysis by capturing the cognitive processes considering one node at a time, thus overlooking the sequence that links one node to the next one and so forth, as for the scheme of Fig. 4. Nevertheless, the abovementioned organization of steps into rows allows the identification of some other relevant elements like the attitude to focus firstly on elements pertaining the problem space and then on the solution space or, conversely, to quickly switch from one to the other and vice versa.

Results Obtained Through the Analysis of the Sequence of Nodes

As mentioned in the section about the main constructs for building a Network of Problems, the nodes are also labelled through a number corresponding to the position of the design move in the whole protocol. Through the organization of this sequence of design moves in rows of a spreadsheet and their related characterization according to Problems and Partial Solutions it is possible to infer some relevant information about the behaviour of designers. Among the main elements that it is possible to highlight through the sequence of links, there is the already mentioned tendency to fixate the attention on specific domains of the design space (problem or solution) or, on the contrary, to move quickly back and forth between them. To this purpose, Table 3 collects all the relevant information emerged from the analysis of the sequences of nodes characterizing the five protocols that have been taken into consideration. The first column specifies the content of each row, while the other columns presents the numerical values through which it is possible to infer the behaviour of the different groups.

Table 3 shows that the different groups have worked with different overall behaviours. For instance, group A and group C have carried out a comparable number of design moves (e.g. Problem 1) with an almost balanced number of design steps between problem and solution space. Moreover, there are quite long chains of design moves during which the focus of the analysis was concentrated on one between Design and Solution space. In these terms, almost all the groups have shown a marked behaviour in prolonging the focus on the same domain of the design space.

Besides, the overall number of design moves in the same time interval (the testers were asked to face the design problems in a constrained timeframe) shows that some groups have evidenced a better fluency in analysing problems and generating ideas. This facet, if compared to the frequencies of shifts between the Problem and the Solution, shows a preliminary indication of a potential correlation

Table 3 Results emerged from the analysis of the sequence of nodes, without considering the links among the nodes of the network

	Group A problem 1	Group B problem 1	Group B problem 2	Group C problem 1	Group C problem 2
# of design moves	79	41	23	78	63
Longest Pb sequence	11	8	13	13	12
Longest PS sequence	12	12	5	9	5
# of Pb/PS and PS/Pb transitions	24	11	3	30	30
Frequency of Pb/PS-PS/Pb transitions per design move	≈30/100	≈27/100	≈13/100	≈38/100	≈48/100
% Pb nodes	45.6 %	56.1 %	65.2 %	53.5 %	55.6 %
% PS nodes	54.4 %	43.9 %	34.8 %	46.5 %	44.4 %

between these two factors. In other words, it seems that the frequency with which a designer or a team shifts from the problem to the solution space influences the number of investigated concepts, being them Problems or Partial solutions.

Results Obtained Through the Application of the Rules for the Identification of Cognitive Processes

As already mentioned in the previous section, the automatic recognition of cognitive processes out of a properly codified design protocol has been just partially implemented into a computer system, so as to reduce the efforts for writing computer code while preserving the overall objectives of obtaining preliminary indications about the reliability and the feasibility of such an approach. In other words, this means that the following approach suffers from its beginning of potential ambiguities because of both:

- The partial overlapping between the conditions to be verified for the identification of cognitive processes;
- The potential characterization of a design move according with more than one cognitive process, because the check is carried out on a single set of links at a time. In order to better clarify the already mentioned limitations, Table 4 shows two examples of good and ambiguous interpretation of the cognitive processes.

With reference to the last columns, the labels "bkw" and "fwd" specify through which rule the computer has identified the cognitive process. Still with reference to the rightest column on Table 4, it is also worth mentioning that, being both the cognitive processes identified by looking backwards, the only reliable indication coming from the example refers to the passage from step 17 to 18. In fact, the

Table 4 Examples of appropriate and ambiguous interpretation of cognitive processes with the presented approach. The underlined cognitive processes in the last column clarify which alternative should be chosen to properly remove ambiguities (*bkw* = backwards, *fwd* = forward)

	Node Id	Design move	#_in_X		#_out_X		Cognitive process ID
			Pb	Ps	Pb	Ps	
Example of appropriate interpretation	17 (PS)	Place the backpack on the front side of the bike	1	0	2	0	*4 (bkw)*
	18 (Pb)	Comfort during riding	1	0	0	0	*7 (bkw)*
Example of ambiguous interpretation	70 (PS)	Shell explosion by heating	1	0	1	0	4 (bkw) or *7 (fwd)*
	71 (Pb)	Peanut integrity	3	3	3	0	2 (fwd) or 3 (bkw) or *7 (bkw)*
	72 (PS)	Application of a pressure field	1	0	1	0	*4 (bkw)* or 7 (fwd)

cognitive process linking those nodes concerns the Evaluation of the proposed solution (node 17) with reference to one of the criteria for considering it suitable or not for its implementation (the comfort of riding of node 18).

On the contrary, the last three rows of Table 4 show an example of ambiguous interpretation of the cognitive processes. What was just mentioned about the impossibility to judge the correctness of the assignation of a cognitive process to node 17 is not valid anymore for node 70. Indeed, in such a situation, two cognitive processes have been recognized as potentially relevant to the node. One looks backwards (Exploration) and the other looks forward (Evaluation). Limiting the consideration to this small excerpt of three nodes, it is possible to define the passage between node 70 and 71 deals with the Evaluation of the solution, which is also suggested as one of the alternative cognitive process identified for node 71 (looking backwards to node 70).

Besides, the transition between node 71 and node 72 follows a different logic: since the explosion of shells by heating may compromise the integrity of peanuts, the designer chose to explore alternative ways to remove the shells from peanuts, e.g. through the application of a pressure field (Exploration backwards from the problem at node 71 and the Partial solution at node 72). Please note that the process of Decomposition (forward) or the Problem agglomeration do not refer to this transition and therefore should be considered as false positives.

Going back to the assignation of the cognitive processes to node 70, it is necessary to say that both the Exploration (backwards) and the Evaluation (forward) could be considered correct in principle. The correctness of the Evaluation has been already discussed before; nevertheless, it is also possible that the solution of node 70 has been presented as an explorative attempt to solve a problem at node 69 (not represented in Table 4). In case both the identification are correct, it is not possible to consider it as the emergence of a false positive. On the contrary, the information proposed by the computer system represents a further check to verify the consistency of the transition between design moves, as it has presented for node 70–71.

As it clearly appears from this last example, there exist several occurrences in which the analysts are asked to remove potential ambiguities in the interpretation of cognitive processes within the design moves of a protocol. The introduction of appropriate warnings for each node requiring the disambiguation allows keeping full control on the correctness of the analysis and, on the other hand, allows the analysts to skip some work in case of no ambiguity. To this purpose, Table 5

Table 5 Summary of the overall results concerning the automatic identification of cognitive processes out of a design protocol

# of identified cognitive processes	Tot	%	Max	Min	Avg	SE
1	138	47.26 %	45	19	27.6	10.6
2	102	34.93 %	39	4	20.4	15.3
3	48	16.44 %	17	0	9.6	6.5
4	4	1.37 %	2	0	0.8	0.8

summarizes values of descriptive statistics as they come out from the analyses of protocols in terms of the automatic recognition of cognitive processes.

The first column of Table 5 collects the number of identified cognitive processes per node of the protocols. Besides, the first row clarifies the content of the other columns by specifying the meaning of the values the cells report. Except for the first column, from left to right, Table 5 collects:

- The total number of nodes for which the computer has identified a finite number of alternative cognitive processes;
- The same value presented above, but as a percentage calculated with the overall number of design moves;
- The maximum and the minimum number of nodes within the analysed protocols that have been characterized by a finite number of alternative cognitive processes;
- The average value and the standard error of the abovementioned value considering the whole set of analysed protocols.

It shows that on a total of 292 design moves, the applied rules have produced no ambiguities in almost one half of the cases. Moreover, the investigation has also clarified that, with this kind of approach, it is necessary to remove the ambiguities among potentially appropriate cognitive process in a set of not more than four items.

In order to consider the feasibility and the suitability of the approach, it follows that the simpler manual coding scheme, considering just Problems and Partial Solutions, has demonstrated to be capable of shifting the recognition of cognitive processes towards a computer automated procedure, which produces unambiguous results for more or less half of the design moves. In the other cases, a small set to choose the appropriate cognitive process from reduces the time required to the analyst for assigning a proper characterization to the design move under investigation. From this perspective, a more complex and comprehensive implementation of the rules, thus including further checks so as to both consider, at the same time, the links entering to or coming out from a node and the sequence of steps that the designers have followed while designing. In more explicit terms, the addition of more articulated set of rules and a relatively greater number of computational steps per each design move of the protocol will allow switching from the current situation presented in this paper towards a more ideal and less time demanding analysis of design protocol. Besides, as briefly mentioned in the previous section, the implementation of some other relatively simple rules, allows the analysts to identify further relevant elements concerning the behaviour of designers during a design activity. With the check of the number of outgoing arrows from a solution node the computer can identify potentially satisfactory solutions that do not trigger further problems as a consequence of its implementation (#_out_Pb = 0) or that haven't been further explored or detailed (#_out_PS = 0).

Table 6 collects this kind of data, with indications about the overall number of nodes (also according to their characterization as Pb and PS).

Table 6 Other characteristics that the proposed approach can map after the automatic analysis of the design tasks carried out by testers

	Group A problem 1	Group B problem 1	Group B problem 2	Group C problem 1	Group C problem 2
# of design moves	79	41	23	78	63
# of problem nodes	36	23	15	46	35
# of partial solution nodes	43	18	8	40	28
# of unexplored problems	21	14	13	17	18
# of satisfactory solution concepts [sat_PS]	23	9	7	15	6
Efficiency of the design process [sat_PS/tot_PS]	53.5 %	50 %	87.5 %	37.5 %	21.4 %

In addition to the number of potentially satisfactory solutions (row 6 from top, including the first one in bold), Table 6 also presents information about the number of problems that haven't been further investigated (row 5) and a preliminary index for the estimation of the efficiency of the problem solving process (ratio between potentially satisfactory PS and the overall number of generated PS, row 7).

From the analysis of Table 6, it emerges that the number of problems the designers left back without further investigation is relatively high if compared to the overall number of Problem nodes (more than 50 % of problems). Furthermore, also with reference to Table 4, it seems that there may exist a negative correlation between the number of unexplored issues and the co-evolutionary behaviour of designers. Higher the tendency of switching between Problem and Design space is, lower is the amount of knowledge elements that gets neglected during the design task. Besides, it is also important to notice that this behaviour does not result in a better efficiency of the design process. Nevertheless, the authors consider that this last aspect require a deeper investigation, because the high number of potentially satisfactory solutions may depend on a very efficient design process or, and it seems that it was the case of the experiment, on the lacks of skills or knowledge to properly perform the evaluation of solution concepts.

Conclusions

Stemming from the increased interest on the analysis of design protocols emerged in literature, and considering the current limitations of such an approach, this paper has presented an original proposal towards the automatic recognition of cognitive processes out of design protocol. The authors have suggested to implement the core constructs of the OTSM-TRIZ Network of Problem as a means to map the design protocols and subsequently apply rules for the identification of cognitive process coherently with the number of links each design move shares with other elements in the network.

The proposed approach has been tested with a simplified computer coding, so as to carry out a preliminary assessment about its feasibility and potentiality. As documented in the previous section, the proposed OTSM-TRIZ framework has shown the capability of reducing the efforts in charge of the analysts, together with the almost automatic generation of relevant information supporting the protocol investigation (almost 50 % of unambiguously identified cognitive processes). To this purpose, the paper has also purposefully focused on the current shortcomings of the approach, which should be proactively considered as promising opportunities for further development in order to slash down the time required for the analysis. The authors have estimated that this approach can reduce the ratio between the time spent for designing and for the analysis of the related protocol from 1:10–100 to 1:3–4, given the simple classification of design moves in Problems and Partial Solutions.

The refinement of the current rules to automatically recognize the cognitive process, in terms of number and complexity can be considered as an opportunity for the development of this approach in the short-medium term. On the contrary, the proper implementation of computing codes for the automatic association of cognitive processes, which goes step-by-step with the building of the network of problems, represents one of the most promising long-term developments of this approach.

References

1. Cross N, Christiaans H, Dorst K (1996) Analysing design activity. Wiley, Chichester
2. Cross N (2001) Design cognition: results from protocol and other empirical studies of design activity. In: Eastman C, Newstetter W, McCracken M (eds) Design knowing and learning: cognition in design education. Elsevier, Oxford, pp 79–103
3. Gero JS, Jiang H, Williams CB (2012) Design cognition differences when using structured and unstructured concept generation creativity techniques. Proceedings of the 2nd International Conference on Design Creativity, Glasgow
4. Cross N (2004) Expertise in design: an overview. Des Stud 25(5):427–441
5. Bilda Z, Gero JS, Purcell T (2006) To sketch or not to sketch? That is the question. Des Stud 27 (5):587–613
6. Dorst K, Cross N (2001) Creativity in the design process: co-evolution of problem–solution. Des Stud 22(5):425–437
7. Jiang H, Yen CC (2009) Protocol analysis in design research: a review. Proceedings of the International Association of Societies of Design Research Conference, pp 147–156
8. Chai KH, Xiao X (2012) Understanding design research: a bibliometric analysis of design studies (1996–2010). Des Stud 33(1):24–43
9. Pourmohamadi M, Gero JS (2011) LINKOgrapher: an analysis tool to study design protocols based on FBS coding scheme. Proceedings of the 18th International Conference on Engineering Design (ICED11)
10. Dekoninck EA, Yue H, Howard TJ, McMahon CA (2010) Developing a coding scheme to analyse creativity in highly-constrained design activities. Proceedings of the International Conference on Design Creativity, Springer
11. Gero JS (1990) Design prototypes: a knowledge representation schema for design. AI Mag 11 (4):26–36

12. Goldschmidt G (1990) Linkography: assessing design productivity. In: Trappl R (ed) Cyberbetics and system. World Scientific, Singapore, pp 291–298
13. Goldschmidt G, Tatsa D (2005) How good are good ideas? Correlates of design creativity. Des Stud 26(6):593–611
14. van der Lugt R (2000) Developing a graphic tool for creative problem solving in design groups. Des Stud 21(5):505–522
15. Cai H, Do EYL, Zimring CM (2010) Extended linkography and distance graph in design evaluation: an empirical study of the dual effects of inspiration sources in creative design. Des Stud 31(2):146–168
16. Perry GT, Krippendorff K (2013) On the reliability of identifying design moves in protocol analysis. Des Stud 34(5):612–635
17. Becattini N, Borgianni Y, Cascini G, Rotini F (2012) Experiencing protocol analysis of computer-guided design tasks. Proceedings of the DESIGN Conference, Dubrovnik, pp 1821–1830
18. Becattini N, Borgianni Y, Cascini G, Rotini F (2012) Method and algorithm for computer-aided inventive problem analysis. Comput Aided Des 44(10):961–986
19. Becattini N, Cascini G, Rotini F (2013) OTSM-TRIZ network of problems for evaluating the design skills of engineering students. Proceeding of the TRIZ Future conference 2013, Paris, 55–66
20. Cavallucci D, Khomenko N (2007) From TRIZ to OTSM-TRIZ: addressing complexity challenges in inventive design. Int J Prod Dev 4(1):4–21
21. Fiorineschi L, Frillici FS, Rissone P (2011) A comparison of classical TRIZ and OTSM-TRIZ in dealing with complex problems. Proceedings of the Triz Future Conference, pp 103–112
22. Baldussu A, Becattini N, Cascini G (2011) Network of contradictions analysis and structured identification of critical control parameters. Procedia Eng 9:3–17
23. Cavallucci D, Rousselot F, Zanni C (2010) Initial situation analysis through problem graph. CIRP J Manuf Sci Technol 2(4):310–317
24. Boden MA (2009) Computer models of creativity. AI Mag AAAI 30(3):23–34
25. Linsey JS, Tseng I, Fu K, Cagan J, Wood KL, Schunn C (2010) A study of design fixation, its mitigation and perception in engineering design faculty. J Mech Des 132–041003:1–12

How Do Interruptions During Designing Affect Design Cognition?

John S. Gero, Hao Jiang, Kinga Dobolyi, Brooke Bellows, and Mick Smythwood

Abstract This paper reports an experimental study exploring how interruptions during designing affect designers' cognition. The results are from studying 14 teams of two undergraduate computer science students. In an experiment with three conditions, each team completed three software design tasks of comparable complexity and scope. The first condition captured designers' activities without interruptions, which served as a baseline for comparison with the other two conditions that explicitly incorporated two interruptive tasks. Design activities of all three conditions were videoed and analyzed utilizing an ontologically-based protocol analysis coding scheme. Inter-experiment comparisons showed that the design cognition of interrupted sessions were significantly different from the uninterrupted sessions, with increased cognitive efforts expended on generative aspect of designing, and decreased efforts on analytic and evaluative aspects. These differences could be accounted for by a strategic compensation, i.e., designers shifted their problem-solving strategies to make up for the interferences produced by interruptions.

Introduction

Interruptions, i.e., postponements or cessations of an ongoing task by another interpolated task, are pervasive phenomena. Most people have the experience of being interrupted by a phone call, text message or an unexpected visitor while

J.S. Gero (✉)
Department of Computer Science and School of Architecture, University of North Carolina, Charlotte, NC, USA
e-mail: john@johngero.com

H. Jiang
Zhejiang University, China

K. Dobolyi • B. Bellows
George Mason University, Fairfax, USA

M. Smythwood
University of North Carolina, Charlotte, USA

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_7

engaging in a problem-solving task. Interruptions are usually considered as disruptive interferences, resulting in lowered efficiency, and/or increased error rates [1], and negative effects on affective states of people, e.g., increased anxiety [2]. Several theoretical accounts for this disruptiveness have been provided [3, 4]. The majority of the research into interruptions is concerned with simple problem-solving tasks or tasks not considered as creative in nature, and the purpose of those studies focused on general human cognitive and perceptual mechanisms.

Creative workers, such as artists and designers, often hold another interpretation for interruptions from a more macroscopic and pragmatic view. Temporarily stepping away from a wicked problem is seen as a heuristic for creative problem solving. Archimedes' "eureka" story is one of many well-known anecdotes for incubation and insights. Empirical studies also reported some improved performance after interruptions [5–7]. These findings indicate that interruptions could possibly facilitate positive incubation effects [8]. This possibility is of particular interest to designers and other creative workers.

Designers often interleave multiple design projects and non-design tasks. Interruptions are natural ingredients of authentic design activities in the real world. The effect of interruptions on design cognition however has not been adequately studied in current research on creative design. Previous work on design cognition has primarily focused on observing continuous designing processes, in which experimental settings explicitly prevented the occurrence of interruptions, e.g., [9, 10]. Recent literature showed a trend of shifting from laboratory-based design experiments to design meetings in real settings, e.g., [11, 12]. Though interruptions during the designing process were observed and audio-visually captured in these studies, they were often treated as noise and not analyzed and discussed in detail. Whether interruptions affect designers' cognition during a designing process remains unclear.

Interruptions, the act of putting the problem aside and temporarily engaging in other interpolated activities, is the hallmark of the incubation stage. Zijlstra et al. [7] argued that the implication of interruptions goes beyond the execution of additional tasks; people may adapt their problem-solving strategies to compensate for the potential performance deterioration. The beneficial effects may be due to the overcompensation. The essence of famous "eureka" story is that Archimedes reshaped his strategy to measure the volume of a crown. Similar strategy adaptions have also been observed in empirical studies involving complex tasks e.g., aviation [15].

Hypotheses

This protocol study examined the potential influence of interruptions on design cognition. Two hypotheses are tested empirically. The first hypothesis is that, regardless of the direction of any effects, interruptions affect design cognition. In other words, designers' cognitive behaviors would be significantly different between interrupted and uninterrupted conditions. This hypothesis is tested through a statistical significance comparison of the measured cognitive behaviors of design session with and without interruptions.

The second hypothesis is that interruptions during designing change designers' strategies. This hypothesis is tested by examining the change in proportions of cognitive effort on reasoning about problem analysis and solution generation. The outcome of designing is the creation of artifacts, tangible or intangible. If designers try to compensate for interruptions, the cognitive effort spent on the generative aspect of designing would be expected to be larger than in uninterrupted conditions. Consequently, the cognitive effort spent on reasoning about problem analysis and solution evaluation would decrease.

Methods

Research Participants

In the research reported in this paper, the effect of interruptions on design cognition were be explored in the domain of software design. Twenty-eight undergraduate students, currently enrolled in introductory level programming and/or software engineering classes at George Mason University, voluntarily participated in this study. They were paired into 14 design teams of two persons. All participants had at least two semesters of programming experience.

Experiment Design and Tasks

This study adopted a repeated-measures design. Each team was asked to carry out the same three types software design tasks, designing a simple algorithm, to potentially be turned into Python code. Descriptions of these tasks are summarized in Table 1. They were set in the same level of complexity, assessed by the educators and researchers. To avoid a situation where designing a solution to an initial

Table 1 Three conditions and tasks in this interruption study

Condition	Task description	Interruption task
1 (control)	To differentiate among colors, favorite numbers and hourly salaries from a list of general website inputs, to form 3 sub-lists and sort each sub-list in a natural order	Not interrupted
2 (experiment)	To find the minimum, maximum, mean, median and most frequently occurring element of a list of integers (without using built-in functions)	Two interruptions at 5 min and 20 min respectively
3 (experiment)	To find all duplicate elements and unique (non-duplicate) elements be-tween two input lists	Two interruptions in 7 min and 19 min respectively

problem would yield inspiration for the solution of a later one, the problems were written to solve unrelated tasks.

To ensure that tasks were of approximately at the same level of difficulty, each task involved the following components: (1) The sorting of a list where each element must be examined at least once; (2) an additional examination of the list where elements must be further sorted or analyzed; (3) both the sorting and analysis components involve a comparison between two elements at a time; and (4) each task involved a higher-level comparison where a single element is being compared to something more complex than just another individual element.

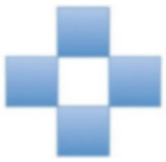
The design tasks are harder than a typical coding question on a final exam of an introductory programming course, but not expected to take a software professional in industry more than 15 min to design and implement. Each of the experiments allocated 45 min to the design task, in which the time spent on tasks during the interruptions was not included. Since many of the components of the solutions could be trivially solved through the use of functions in the Python standard library, subjects were asked to not use any of these built-in functions in their algorithms.

The experiment had three conditions. Each condition implemented one of three software design tasks. The first condition was conducted as the control condition, explicitly excluding any interruptions during the designing process. It served as a baseline for comparison with the other two experiment conditions. The uninterrupted condition also makes possible comparisons with other design cognition studies reported in the literature.

The other tasks comprised the experiment condition. Two interruptions were introduced in the course of designing. This study did not randomize the sequence of control and experimental conditions. Rather, the possible ordering effects or learning effects were assessed by the replication of the experiment conditions.

All interpolated tasks in the experiment conditions were structured in the same format. This format included ten sub-tasks requiring low to medium cognitive demands, including memory tasks, mental arithmetic and visual reasoning. Table 2 presents some examples for each type of question. The interrupted time points were

Table 2 Sample items for the interrupted task

Type of questions	Memory task	Mental arithmetic	Visual reasoning task
Cognitive demand	Low	Low to medium	Medium
Examples	Today's date: ________	9 times 9 = _______	How many squares do you see in the figure below? 
	Your name: ________	12 times 11 = _______	

slightly different between experimental conditions, as exactly the same setting may make participants expect interruptions during the third task.

Measurements

All the design activities, including designers' utterances, drawings and gestures, were videoed and then examined using the FBS ontologically-based protocol analysis methodology [16]. In this methodology, design cognition is operationalized by a set of design issues and syntactic design processes.

A principled coding scheme, based on the FBS ontology [13, 14], that classifies designers' cognitive issues in terms of the ontological variables of function, behavior and structure, plus an external requirement and a design description, Fig. 1. The function (F) of a designed object is defined as its teleology; the behavior (B) of that object is either derived (Bs) or expected (Be) from the structure, where structure (S) represents the components of an object and their compositional relationships. These ontological classes are augmented by requirements (R) that come from outside the designer and description (D) that is the document of any aspect of designing, without introducing any new ontological classes.

Transcripts of audio-visually recorded design activities were segmented and coded using these six FBS codes. Each segment of design activity was strictly assigned with only one of the six codes, corresponding to a single underlying design issue. The design cognition of each design session was thus transformed into a sequence of design issues. A syntactic model was then applied to derive syntactic design processes as transitional processes between pairs of design issues [17]. The relationship between design issues and syntactic processes is illustrated in Fig. 1.

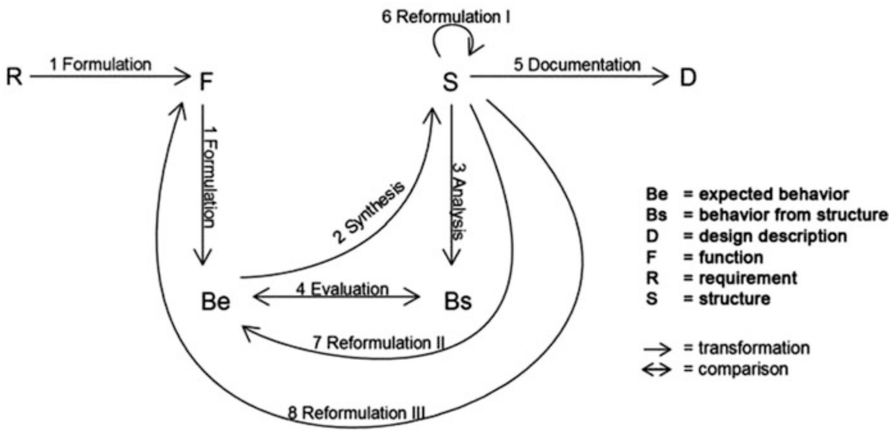


Fig. 1 The FBS ontology (After [14])

Table 3 Mapping design issues and syntactic processes onto three-pronged design model

General aspects of design cognition	Design issues	Syntactic design process
Analytic aspect (problem framing)	Requirement (R)	Formulation
	Function (F)	Reformulation II
	Expected behavior (Be)	Reformulation III
Generative aspect (solution synthesis)	Structure (S)	(Solution) synthesis
		Reformulation I
Evaluative aspect (solution evaluation)	Behavior from structure	(Solution) analysis
		(Solution) evaluation

In the design literature, the three-pronged "analysis-synthesis-evaluation" model [18–20] is a well-accepted, basic theoretical framework of designing. A mapping scheme, Table 3, was utilized to translate our research questions into operational hypotheses directly testable with the measurements of design issues and syntactic processes. As description issue and documentation process do not have equivalent components in this three-pronged design model, results concerned with these two measurements will not be elaborated or discussed further in this study.

Operational Hypotheses

Utilizing the measurements of design issues, the theoretical hypotheses presented earlier can be translated in operational terms as follows:

Hypothesis 1a (generative aspect of design cognition):
Interrupted sessions have a higher percentage of structure issues than uninterrupted sessions.

Hypothesis 1b (generative aspect of design cognition):
Interrupted sessions have a higher percentage of synthesis and reformulation I processes than uninterrupted sessions.

Hypothesis 2a (analytic aspect of design cognition):
Interrupted sessions have lower percentages of requirement, function and expected behavior issues than uninterrupted sessions.

Hypothesis 2b (analytic aspect of design cognition):
Interrupted sessions have lower percentages of formulation, reformulation II and reformulation III processes than uninterrupted sessions.

Hypothesis 3a (evaluative aspect of design cognition):
Interrupted sessions have a lower percentage of behavior from structure issue than uninterrupted sessions.

Hypothesis 3b (evaluative aspect of design cognition):
Interrupted sessions have lower percentages of analysis and evaluation processes than uninterrupted sessions.

This paper focuses on the strategic adaption due to interruptions, i.e., compensating acts for interruptions. Hypotheses 1a and 1b are our main hypotheses. Hypotheses 2a, 2b, 3a and 3b are additional hypotheses, which are natural consequences if main hypotheses are supported.

Methods of Analysis

The data analysis consists of two steps. It first examines whether there is statistically significant difference of design issue/process distributions between two experimental conditions. If no difference is found, these two experimental conditions are then collapsed by averaging the corresponding measurements, and then compared with the uninterrupted condition. Paired-samples *t* test or Wilcoxon signed ranks test were used, depending on whether sampling distributions of measurements were approximately normal.

If there a statistically significant difference is found between the two interrupted sessions, three conditions will be compared using one-way analysis of variance (ANOVA). All significance tests were performed using IBM SPSS v21. The effect sizes were calculated by G*Power 3.1.7.

Results

Descriptive Statistics

Applying the FBS ontologically-based segmentation and coding scheme, the intercoder agreements for each session were between 86 % and 92 %. The final data for analysis were the arbitrated data that resolved the segmentation and/or coding disagreements. After the protocol segmentation, coding and arbitration, the observations of these three conditions were transformed into an average of 210 ~ 280 (SD: 53 ~ 76) design issues and 110 ~ 148 (SD: 33 ~ 56) syntactic design processes, Fig. 2.

Figure 3 presents the distributions of design issues measured across the three conditions of this experiment. It shows that the majority of cognitive effort was expended on reasoning about structure and behaviors of design. Less than 5 % of design issues articulated requirements and functions. Two interrupted sessions, Tasks 2 and 3, share a similar design issue distribution, which were different from the distribution of the uninterrupted session, Task 1. Figure 3 shows that Task 1 has a lower percentage of structure issues, while having higher percentages of the two behavior issues.

Figure 4 illustrates the syntactic process distributions of the three experiments. The processes of formulation and reformulation III occupied a very limited

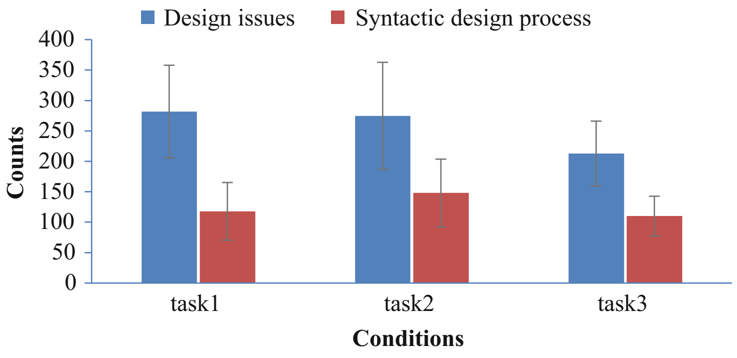


Fig. 2 Descriptive results of protocol segmentation

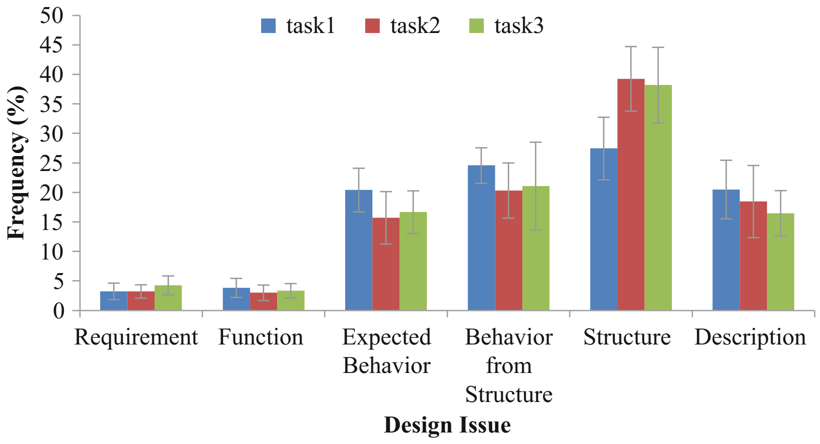


Fig. 3 Distribution of design issues (Error bars represent standard deviations)

percentage of total syntactic processes (less than 5 %). The most obvious inter-task differences can be seen in evaluation and reformulation I processes. The percentage of reformulation I processes in the uninterrupted task (task 1) was almost half of the interrupted tasks (tasks 2 and 3), but the percentage of evaluation processes was more than double of the interrupted tasks. The percentages of synthesis, analysis and reformulation II processes also tended to be different between uninterrupted and interrupted tasks.

Comparisons Between Two Experimental Conditions

Before the inferential statistical analysis, measurements of design issues and syntactic design processes were tested to determine if they fulfilled the normality

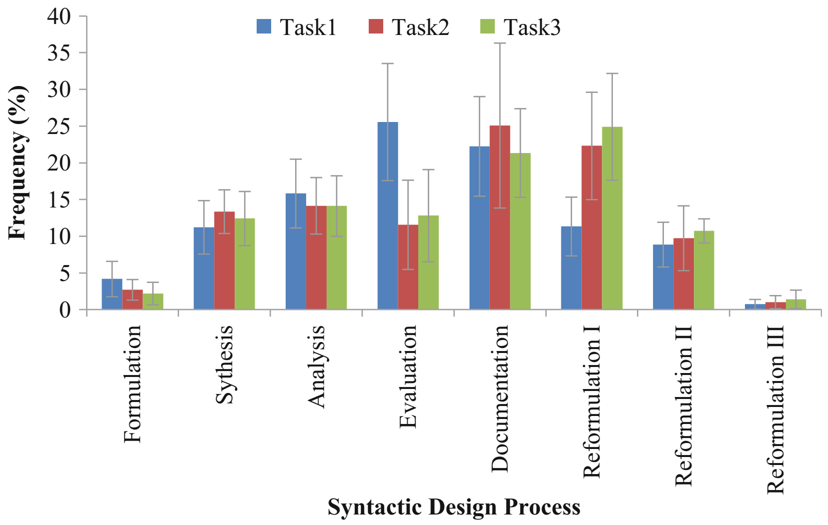


Fig. 4 Distribution of syntactic design processes (Error bars represent standard deviations)

Table 4 Pairwise comparisons of design issues between experimental conditions

Design issue	*t* statistic	Sig (2-tailed)
Requirement (R)	−1.844	0.088
Function (F)	−0.675	0.511
Expected behavior (Be)	−0.633	0.537
Behavior from structure (Bs)	−0.337	0.742
Structure (S)	0.611	0.552
Description (D)	1.217	0.245

assumption, using the Shapiro-Wilk *W* test. The paired-samples *t* test was used when the sampling distributions of two counterparts were both approximately normal. If the normality assumption was violated, the Wilcoxon signed ranks test was used instead. Statistically significant differences were assumed at a significance level (α) of 0.05.

Table 4 tabulates pairwise comparisons of design issues between the experimental conditions, Tasks 2 and 3. No statistically significant issue differences were observed between these two interrupted sessions. The same negative results were replicated using the measurement of syntactic design processes, Table 5. The homogeneity of the two interrupted conditions was supported. Measurements of these two experiment conditions were thus aggregated, and compared with the uninterrupted (control) condition as a whole.

Table 5 Comparisons of syntactic processes between experimental conditions

Syntactic design process	*t*/*W* statistic	Sig (2-tailed)
Formulation	0.282	0.778
Synthesis	0.639	0.534
Analysis	0.024	0.981
Evaluation	−0.973[a]	0.331
Documentation	1.193	0.254
Reformulation I	−1.076	0.302
Reformulation II	−0.747	0.469
Reformulation III	−0.628[a]	0.530

[a]W statistic of paired-sample Wilcoxon Signed Rank test; the remaining statistics are the t statistic of paired-sample *t* test

Table 6 Inter-conditional comparisons of design issues

Design issue	*t* statistic	Sig (2-tailed)	*d*
Requirement (R)	−1.490	0.162	−0.413
Function (F)	1.318	0.212	0.366
Expected behavior (Be)	3.512	0.004*	0.974
Behavior from structure (Bs)	2.137	0.054	0.593
Structure (S)	−5.178	0.000*	−1.436
Description (D)	2.075	0.060	0.575

*Statistically significant at the level of $p \leq 0.05$

Comparisons Between the Interrupted and Uninterrupted Conditions

Table 6 and Fig. 5 summarize the inter-conditional comparisons using the measurements of design issues. Differences between uninterrupted and interrupted tasks were mainly found in the issues of structures and expected behaviors. Compared with the uninterrupted condition, designers exhibited a significantly higher percentage of structure issues and significantly lower percentage of expected behavior issues, when they were interrupted during designing. The magnitudes of difference, indicated by Cohen's *d*, were substantially large in terms of these two issues. The interrupted condition also had lower percentages of behavior from structure and description issue than uninterrupted condition. The differences were of marginal significance, and of medium effect size in terms of difference magnitude.

The differences across interrupted and uninterrupted conditions were then compared using the measurements of syntactic design processes, Table 7 and Fig. 6. The inter-conditional differences were mainly observed in evaluation and reformulation I processes. When interrupted, designers exhibited a significantly higher percentage of evaluation process, and a significantly lower percentage of reformulation I process. The effect sizes of difference were large in terms of Cohen's *d*. There was another marginal difference observed in formulation process ($p = 0.06$). But the frequency of this process was very low (about 5 % of total syntactic processes),

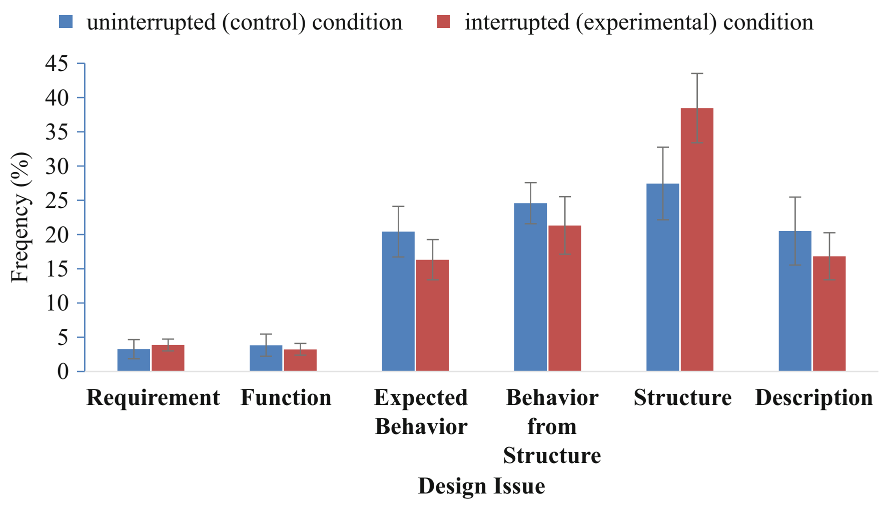


Fig. 5 Distribution of design issues (Error bars represent standard deviations)

Table 7 Inter-conditional comparisons of syntactic design process

Syntactic design process	*t*/*W* statistic	Sig (2-tailed)	*d*
Formulation	2.076	0.060	0.576
Synthesis	−1.321	0.211	−0.366
Analysis	0.859	0.407	0.238
Evaluation	4.971	0.000*	1.379
Documentation	−0.175[a]	0.861	−0.005
Reformulation I	−5.515	0.000*	−1.529
Reformulation II	−1.572[a]	0.116	−0.379
Reformulation III	−1.684	0.118	−0.467

[a]W statistic of paired-sample Wilcoxon signed rank test; the rest statistics are t statistic of paired-sample *t* test
*Statistically significant at the level of $p \leq 0.05$

we thus did not consider that the difference in this process was able to contribute a substantial difference in terms of overall design cognition.

Discussions

The experiments reported in this paper provide an opportunity to examine the effects of being interrupted during a designing process. Design cognition measured in the two interrupted conditions showed a statistical homogeneity: all pairs of design issue and syntactic design process were not significantly different between Tasks 2 and 3. Results from the comparisons between interrupted and uninterrupted

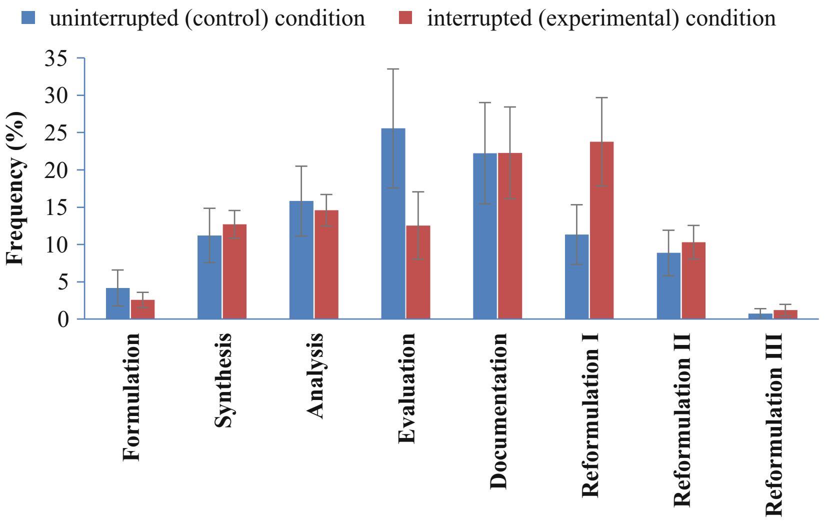


Fig. 6 Distribution of design issues (Error bars represent standard deviations)

conditions generally supported our hypotheses that the interruptions influence designers' cognition.

Main Hypothesis: Interruptions Make Designers More Focus on Solution Synthesis

This study used the structure issue and the syntactic processes of synthesis and reformulation I to operationalize the generative aspect of design cognition. Except from synthesis process (no statistically significant difference were found), results of the other two measurements strongly supported our main hypothesis. The percentages of structure issues and reformulation I processes in the interrupted condition were both significantly larger than the uninterrupted control condition, and the effect sizes of pairwise differences were all substantially large in terms of Cohen's d (-1.44 for comparisons of structure issue, and -1.53 for reformulation I process).

Interruptions could be detrimental to the performance of the primary task, as additional cognitive efforts are expended towards the interpolated task. The increasing percentages of generative aspect of cognition measurements during interrupted sessions suggest that designers may make some strategic shifts to increase problem-solving efficiency and compensate for the possible negative influences of interruption.

This solution-orienting effect of interruptions may have implications in creativity theory. Temporary shifts away from an ongoing task are often discussed as a stage for "incubation". The beneficial strategic adaption, more concentrated on solution generation, may partially explain the incubation effects [21–23].

Additional Hypotheses: Interruptions Make Designers Less Focused on Problem Analysis and Solution Evaluation

As a consequence of increased focus on solution generation, designers' cognitive effort spent on other aspects of designing, i.e., problem analysis/formulation and solution evaluation, are reduced during interrupted sessions.

The evidence of reduced focus on problem analysis/formulation was obtained from the expected behavior issues, Table 6. A large effect size (Cohen's $d = 0.97$) was observed.

The lowered focus on solution evaluation was mainly demonstrated in the syntactic processes of evaluation. Figure 6 shows that this syntactic process measured in the interrupted condition was only half of its uninterrupted counterpart. The dramatic drop of evaluation effort suggests that designers' compensating strategies come with a cost: designers may not critically scrutinize the consequences of their design solutions as much when they are interrupted amidst designing compared to not being interrupted.

The results about behavior from structure issues also complied with the pattern that designers in interrupted conditions expended less effort on the evaluative aspect of designing. The interrupted condition had a lower percentage of behavior from structure issue than the uninterrupted condition (21.3 % vs 24.6 %, Fig. 5) at the marginal significance level.

Issues Related to Validity

In this paper, we sought to expand upon traditional design tasks that have been previously studied by examining a more complex algorithm construction session in a group setting. Although we believe that the results of this study likely generalize to a larger population, our experiments are a first step in this domain, and may be subject to some limitations. We used students with approximately two semesters of programming experience as participants. It is possible our conclusions do not generalize beyond this group, and that subjects with more programming experience may be less affected by interruptions. Future work will be able to explore a broader subject pool.

Conclusion

This study explored 14 student teams' software design activities in uninterrupted and interrupted conditions, using the FBS ontologically-based protocol analysis methodology. Interruptions are often seen as a hallmark of an incubation period. Understanding the role of interruptions could help us to take advantage of their beneficial incubation effects and prevent their detrimental influences. Results from this preliminary study indicate that interruptions could significantly affect designers' cognition. In particular, designers were more focused on reasoning about solution generation during the interrupted conditions. This may be explained by designers shifting their problem-solving strategies to make up for interruptions [24, 25]. Details of strategic changes, as well as the pros and cons of this strategic compensation, will be further investigated in the future studies.

The significance of studying the effects of interruptions transcends its potential role in incubation and its consequential connection to creativity. Today we live in a world where interruptions are increasing: emails, text messages, Facebook messages and tweets are displayed or notified to us as they occur. It becomes increasingly difficult not to be interrupted while we are carrying out our tasks. The empirical results from these experiments show that interruptions have an effect on the designers while they are designing.

Acknowledgements This research is supported by the National Science Foundation under Grant Nos IIS-10020709 and CMMI-1161715. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. McFarlane DC (2002) Comparison of four primary methods for coordinating the interruption of people in human-computer interaction. Hum Comput Interact 17:63–139. doi:10.1207/S15327051hci1701_2
2. Bailey BP, Konstan JA (2006) On the need for attention-aware systems: measuring effects of interruption on task performance, error rate, and affective state. Comput Hum Behav 22:685–708. doi:10.1016/j.chb.2005.12.009
3. Altmann EM, Trafton JG (2002) Memory for goals: an activation-based model. Cognit Sci 26:39–83. doi:10.1016/S0364-0213(01)00058-1
4. Trafton JG, Monk CA (2008) Task interruptions. In: Boehm-Davis DA (ed) Reviews of human factors and ergonomics, vol 3. Human Factors and Ergonomics Society, Santa Monica, pp 111–126
5. Speier C, Vessey I, Valacich JS (2003) The effects of interruptions, task complexity, and information presentation on computer-supported decision-making performance. Decis Sci 34:771–797
6. Ratwani RM, Trafton JG, Myers C (2006, 16–20 Oct) Helpful or harmful? Examining the effects of interruptions on task performance. In: Proceedings of the Human Factors and Ergonomics Society, 50th Annual Meeting, San Francisco, CA. Sage Publications, pp 372–375

7. Zijlstra FRH, Roe RA, Leonora AB, Krediet I (1999) Temporal factors in mental work: effects of interrupted activities. J Occup Organ Psychol 72:163–185
8. Sio UN, Ormerod TC (2009) Does incubation enhance problem solving? A meta-analytic review. Psychol Bull 135:94–120. doi:10.1037/a0014212
9. Cross N, Christiaans H, Dorst K (eds) (1996) Analysing design activity. Wiley, Chichester
10. Gero JS, Jiang H, Williams CB (2013) Design cognition differences when using unstructured, partially structured, and structured concept generation creativity techniques. Int J Des Creat Innov 1:196–214. doi:10.1080/21650349.2013.801760
11. McDonnell J, Lloyd P (eds) (2009) About: designing: analysing design meetings. CRC Press, Boca Raton
12. Gero JS, Jiang H, da Silva VS (2013) Exploring a multi-meeting engineering design project. In: Chakrabarti A, Prakash RV (eds) ICoRD' 13: global product development lecture notes in mechanical engineering. Springer, New Delhi, pp 73–84. doi:10.1007/978-81-322-1050-4_6
13. Gero JS (1990) Design prototypes: a knowledge representation schema for design. AI Mag 11:26–36
14. Gero JS, Kannengiesser U (2004) The situated function-behaviour-structure framework. Des Stud 25:373–391. doi:10.1016/ j.destud. 2003.10.010
15. Latorella KA (1999) Investigating interruptions: implications for flightdeck performance. Virginia National Aeronautics and Space Administration, Hampton
16. Kan JWT, Gero JS (2009) Using the FBS ontology to capture semantic design information in design protocol studies. In: McDonnell J, Lloyd P (eds) About: designing: analysing design meetings, CRC Press, Boca Raton, pp 213–229
17. Gero JS (2010) Generalizing design cognition research. In: Dorst K, Stewart SC, Staudinger I, Paton B, Dong A (eds) DTRS 8: interpreting design thinking. University of Technology Sydney, New South Wales, pp 187–198
18. Archer LB (1984) Systematic method for designers. In: Cross N (ed) Developments in design methodology. Wiley, New York, pp 57–82
19. Cross N (2008) Engineering design methods: strategies for product design, 4th edn. Wiley, Chichester
20. Pahl G, Beitz W, Feldhusen J, Grote K-H (2007) Engineering design: a systematic approach (trans: Wallace K, Blessing L, 3rd English edn.). Springer, New York
21. Smith SM, Dodds RA (1999) Incubation. In: Runco MA, Pritzker SR (eds) Encyclopedia of creativity, vol 2. Associated Press, San Diego, pp 39–44
22. Brockett C (1985) Neuropsychological and cognitive components of creativity and incubation, Unpublished Doctoral Dissertation, Virginia Commonwealth University
23. Dodds RA, Ward TB, Smith SM (2004) A review of the experimental literature on incubation in problem solving and creativity. In: Runco MA (ed) Creativity research handbook, vol 3. Hampton Press, Cresskill
24. Elio R, Scharf PB (1990) Modeling novice-to-expert shifts in problem-solving strategy and knowledge organization. Cognit Sci 14:579–639. doi:10.1207/s15516709cog1404_4
25. Rijkes CPM, Kelderman H (2007) Latent-response rasch models for strategy shifts in problem-solving processes. In: Carstensen CH (ed) Multivariate and mixture distribution rasch models. Springer, New York, pp 311–328

Conversation and Critique Within the Architectural Design Process: A Linkograph Analysis

Pieter Pauwels, Tiemen Strobbe, Jeroen Derboven, and Ronald De Meyer

Abstract Conversation and critique are central to architectural design practice as they function as tools for probing and further improving design ideas. We study the kind of design activities that take place in such conversation and critique within the architectural design process. We use linkographs to characterise the design process taking place during conversation. More precisely, we study conversations between design teachers and design students. In this article, an example design process is considered that takes place via a traditional face-to-face meeting. Using the resulting linkograph, we are able to assess the kind of design activity taking place during such sessions of conversation and critique.

Introduction

In this article, we will investigate a specific kind of architectural design engagement. Namely, we focus on the interaction among architectural designers during a session of conversation and critique concerning presented design ideas. We do this via an experiment that consists of a design team, a design teacher, and a specific design task.

In the experiment, the design team presents their design using a slideshow presentation. This presentation takes place after the first month of a design process that spans about 3 months, and it functions as an intermediate presentation of results. The design teacher gives feedback on the presented design, in close interaction with the design students. We have analysed this process of conversation and critique as if it were a traditional design process. More precisely, we analyse it using think-aloud protocols and linkographs. In this case, the statements produced during the actual conversation are used as the instances of the think-aloud protocol. We try to identify 'design episodes' and analyse to what extent such design episodes can be subdivided into smaller design episodes in which smaller design experiments are performed and smaller design decisions are made. By doing this

P. Pauwels (✉) • T. Strobbe • J. Derboven • R. De Meyer
Ghent University, Ghent, Belgium
e-mail: pipauwel.pauwels@ugent.be

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_8

work, we hope to relate the larger picture of the design process to the specific design activities and design decisions that took place in the design process at hand.

We start in section "Introduction" with documenting the background of this study and the reasons why we would want to analyse conversation and critique in this manner. In sections "Conversion and Critique" and "Case Study: Refurbishing High-Rise Buildings in Antwerp", an outset and methodology are given for the experiment. In section "The Design Brief: Three Outdated High-Rise Buildings", a more detailed analysis is made using linkography.

Conversation and Critique

In this study, we consider the element of conversation and critique during an architectural design process. This focus can enhance our understanding of the effect of conversation and critique on the design process:

- How is ideation taking place during conversation,
- What is the role of design fixation in the role-play of design critique,
- To what extent evolves the design during the conversation,
- How is this evolution structured and characterised.

Conversation and critique are different from a traditional preliminary sketch phase. Yet, this kind of interaction is of considerable importance as well to the design process as a whole. Pauwels et al. [1] presents a schematic outline of the reasoning processes involved in designing. This schema entirely builds around the combination of an external world, on the one hand, and the human mind and its guiding principles, on the other hand. The interaction between both is crucial. In terms of this schema [1], conversation and critique among two people can be considered as a specific kind of interaction between two human minds and their respective external worlds. The external world of the first person then mostly consists of the feedback received by the second person in the dialogue, whereas the external world of the second person mostly consist of the feedback received by the first person in the dialogue. As the guiding principles or background knowledge of the two interacting people are personal and thus inherently different, a clash occurs between the two. The conversation then aims at finding some sort of general mutual agreement of how the design should be interpreted and how it should consequently evolve. In looking for such an agreement, not only the guiding principles of both actors in the conversation change, also the design itself evolves into something new. On this basis, we consider the design moves taking place during conversation and critique as part of a creative design process, similar to an equally dedicated preliminary sketch phase.

With every move of interaction in the conversation, a certain evaluation or reflection is performed by the actor, in this case the designer, about the external interaction. Based on this evaluation or reflection, the guiding principles of the actor change, as well as the current interpretation or interpretation of the design itself. Note that, in most cases, not only is there an evaluation or reflection performed *after* the

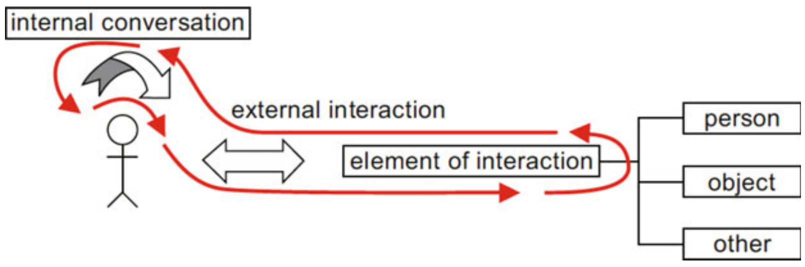


Fig. 1 One loop for each interaction that a designer makes with the environment, including both external interaction and internal conversation

interaction, there is also a form of reflection performed *before* the interaction. This means that the actor consciously or unconsciously considers what he or she expects as a reaction from the external element of interaction. Any act in the conversation thus starts from an internal expectation that is part of an internal conversation.

One might consequently argue that any external interaction inherently includes a form of internal conversation. Internal and external interaction might thus be considered as tightly joint elements in one recurrent and continuous interaction with a surrounding world. We can thus consider one process following the arrow lines in Fig. 1, including internal conversation and external interaction in one loop. "*Ideas are developed in the mind*; *they are thoughts*, *conceptions that serve us to reason with*" [2, p. 5].

Also other researchers have pointed towards the importance of conversation and/or interaction. For instance, Lymer et al. [3] considers conversation in architectural design as a "*rich site for the reproduction of architectural knowledge*, *in which multiple spatial and disciplinary contexts are embedded through representation*, *discourse*, *and embodied practice*" [3, p. 197].

Case Study: Refurbishing High-Rise Buildings in Antwerp

Our case study consists of the transcript of a conversation that was made between a design teacher and a team of design students. This conversation was part of an architectural design studio that took place during 2013 in the Department of Architecture and Urban Planning, Ghent University, Belgium.

The Design Brief: Three Outdated High-Rise Buildings

Design students had received the assignment to design an alternative concept for three outdated high-rise apartment buildings in the city of Antwerp, Belgium. The three towers are located along the A12 motorway in Antwerp. Furthermore, the students were asked to investigate to what extent the concept of co-housing can be accommodated in this high-rise type of building. The design brief is highly constrained by its location. The location between a residential area and the busy motorway presents a delicate urban context. Other constraints need to be addressed as well:

- The buildings on the site need to incorporate about 300 living units along with the facilities needed for *co-housing* and a *parking area* large enough to accommodate needs of the inhabitants and their visitors.
- Attention should be paid also to the *quality of the area surrounding* the high-rise buildings. The combination of the residential area, the area surrounding the high-rise buildings and the high-rise buildings themselves present considerable challenges in terms of *scale* and *feeling of safety and comfort*.
- *Sunlight* needs to penetrate not only into the building units within the high-rise buildings, it also needs to reach the residential area and the area surrounding the high-rise buildings.
- Considerable *fire safety* and *accessibility* constraints are present as well in the kind of high-rise buildings in the design context. For instance, fire safety and accessibility regulations implicate the need for compartmentalization measures, the need for large, separate evacuation staircases, the introduction of circulation shafts enclosed with fire doors, the prohibition of apartments spanning three floors, and so forth.
- The need for *privacy* within the living units.
- *Fluctuating wind turbulence* on the terraces and at the base of the high-rise buildings.
- *Structural constraints* inherent to any high-rise building.

The Considered Design Conversation

The particular conversation that is handled in this article dates 30 October 2013 and lasts for about 1 h. In this conversation, the student team presents their work of the last week by means of a slideshow presentation and a building model. The slideshow consists of eight slides displaying schemas and sketches that are used for reference during the conversation. In the conversation, the design team starts explaining the current design status while referring to their slides. During the presentation, the design teacher gradually starts to give feedback, making the presentation evolve into a discussion that influences the design process.

The design for the high-rise buildings that is presented in the current case study, starts from the *co-housing* concept. The design team hereby aims to implement cohousing at different scales (unit scale/community scale/tower scale/tower group scale/area scale – Fig. 2).

Apart from the scaled co-housing concept, the design team aims at incorporating *flexibility and variation* in the tower design. For example, the design team decides to diversify the type of units provided in the tower: large apartments, double-storey duplexes, small individual studios, and so forth. Also the *façade* follows this design intent, and finds its form through the patchwork of units, terraces and circulation shafts

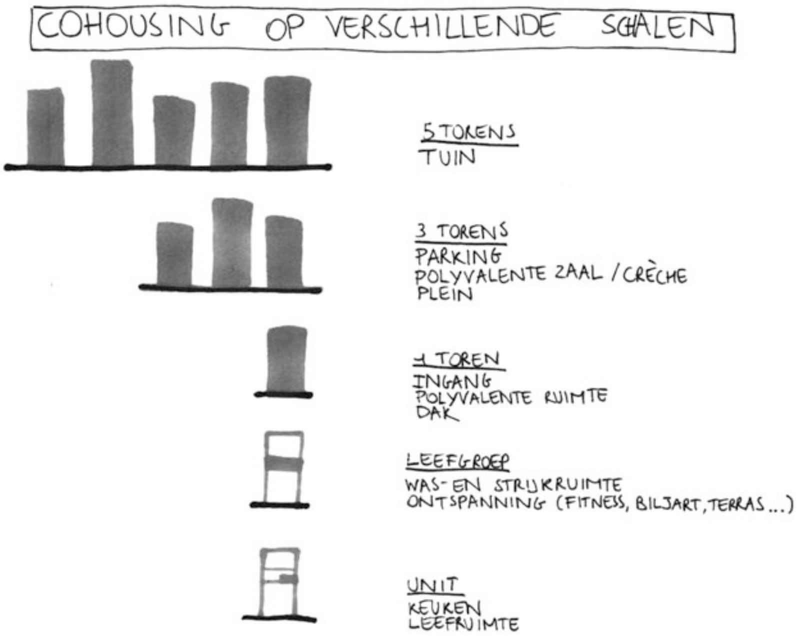


Fig. 2 The presentation of the cohousing concept, which is to be implemented on five scales: an area scale, a tower group scale, a tower scale, a community scale, and an individual unit scale

that are composed behind this façade. How this translates to a sound and logically formed *building structure* is also included in this part of the design conversation.

Finally, a large part of the design conversation also deals with the design of the relation between the high-rise buildings and the surrounding area, which includes *a parking and a park area*. The presentation of the design team ends with a sketch that represents the main idea behind this relation. Most importantly, an extra ground level or *deck* (+1) is introduced at the base of the towers. Beneath this local ground level, parking space is provided; and on top of the ground level, a surrounding park area is provided. This elevated ground level curves down to the actual ground level in areas without a tower.

Method: Linkograph Analysis

The session that is considered in this article was audio-recorded, transcribed and analysed using linkography, which is a well-documented and proven method to quantitatively study design processes. The method was first introduced by Gabriela Goldschmidt in 1990 [4]. The Function-Behaviour-Structure (FBS) ontology [5] was used within the linkograph analysis to assess which kinds of design moves are at play in the critical conversation between the design teacher and the design student team. For the actual analysis, the LINKOgrapher tool [22] was used.

Linkography

Processes of design thinking are most commonly analysed with protocol studies [6, 7]. In this method, a track record is obtained from designers involved in design activity through think-aloud protocols [8]. Example studies were documented by Ennis and Gyeszly [9] and Kavakli and Gero [10]. Although diverse methods exist to analyse protocol studies, linkography can be considered as one of the most successful. Linkography is a method for representation and analysis of design processes focusing on links among design ideas. The method was first introduced to protocol analysis for assessing the design productivity of designers [4]. It was then further developed by Goldschmidt [11–13] and used by others [14–18]. Linkography has been established as a quantitative evaluation technique in protocol analysis to study designers' cognitive activities.

In order to produce a linkograph, the recorded design protocol is transcribed and subdivided into small segments of approximately one sentence. This typically results in a spreadsheet file with a chronological list of all statements made in the design process. Each resulting segment is considered a *design move* and given a sequence number, typically using the same spreadsheet file. Goldschmidt defines a 'design move' as "*a step, an act, an operation which transforms the design situation relative to the state in which it was prior to that move*" [11]. Second, the protocol study is analysed for associations between the distinct design moves, resulting in a network of links between the design moves [19], which can also be recorded in the same spreadsheet file. Goldschmidt hereby distinguishes two types of links: backlinks (links from a particular design move to a *preceding* design move) and forelinks (links from a particular design move to a *subsequent* design move). The way in which these two types of links come about, and the way in which they ought to be interpreted is comprehensively outlined by Goldschmidt in 1995 [11]. "*For each move we pose but one question: is it linked to every one of the moves that precede it in a given sequence of moves such as a design unit? We use a binary reply system of 'yes' and 'no' only, and the sole criterion used to determine linkage or its absence is common sense, in the context of the design task. Thus we establish links among a given move and previous moves, and these links are called backlinks, because they go back in time. With hindsight, linkography allows us to specify the links that a move makes to subsequent moves. These links are the move's forelinks, because they go forward in time. In contrast to backlinks, which can be determined at the time a move is made, forelinks can be determined only after the fact, when the entire process is completed, and as a consequence of having registered all backlinks. The two kinds of links are very different conceptually: backlinks record the path that led to a move's generation, while forelinks bear evidence to its contribution to the production of further moves*".

Using a linkograph, typically recorded in the earlier mentioned spreadsheet file, the design process can be analysed in terms of the patterns in the linkograph, which display the structure of design reasoning. Using the Link Index (LI) and Critical Moves (CM) parameters, a quantitative analysis can be made of the protocol study [11, 20]. The LI parameter equals the ratio between the total number of links and the total number of design moves in the linkograph. A high link index then supposedly indicates a productive design process, as the produced design moves are highly related to each other, and many of the links thus were productive in creating a coherent design process. The CM parameter indicates design moves with a high number of forelinks or backlinks. A critical move can thus be understood as a design move that had a high impact on the design process, and, as such, also on the eventual design product.

Nevertheless, Kan and Gero [19] argue that the LI and CM parameters are not the best indicators of design productivity, by arguing that a fully saturated linkograph, which thus has a high LI and a high CM number, indicates no diversification in ideas, hence less design productivity. They point towards using entropy measures as indicators of design productivity. Shannon [21] defines entropy as a measure of information. The measure of information carried by a message or symbol depends on the probability of its outcome. If there is only one possible outcome, then there is no additional information because the outcome is already known, thus resulting in a low entropy value and a low design productivity [17]. We will use both measures (LI and CM; entropy) to analyse the studied design conversation.

The FBS Ontology

To further improve the analysis of a linkograph, a Function – Behaviour – Structure (FBS) ontology [5] can be used. The terms used in the ontology are schematically shown in Fig. 3, for reference.

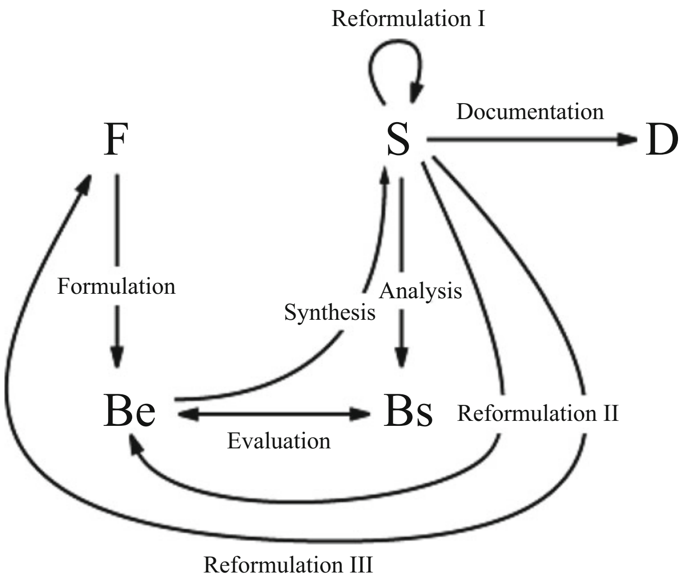


Fig. 3 Schematic overview of the FBS coding scheme

The FBS ontology allows coding the character of the design moves identified in the linkograph. The coding scheme consists of the six following codes.

- Requirements (R)
- Function (F)
- Behavior derived from expectations (Be)
- Behavior derived from structure (Bs)
- Structure (S)
- Documents or design descriptions (D)

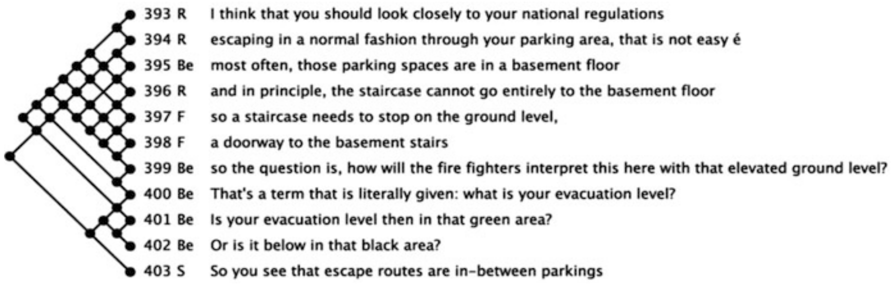
A brief description of the FBS ontology and its six codes is given by Kan et al. [18], so we do not elaborate on this any further in the remainder of this paper. When combining the FBS ontology and linkography, the kind of change initiated by every single design move in a linkograph can be formally characterized. The design process is hereby considered as a process that starts from a set of requirement (R) and function (F) statements, which are continuously analysed (Bs), evaluated (Be) and synthesised into structure (S) statements. Eventually, documentation (D) statements are produced, documenting the structure coming out of final design decisions. After encoding, eight design transformation types can be considered (Fig. 6) [22, 23]: formulation (F ->Be), synthesis (Be ->S), analysis (S ->Bs), evaluation (Bs<->Be), documentation (S ->D), reformulation I (S ->S), reformulation II (S ->Be), and reformulation III (S ->F). These transformation types will be referred to below as 'FBS processes'.

LINKOgrapher

For making the analysis of the considered case study, we used the LINKOgrapher tool [22]. This tool relies on an input spreadsheet file that encodes the distinct design moves, the links between the design moves, and the FBS codes affiliated to all design moves. Using this information, the LINKOgrapher tool not only generates a visual representation of the resulting linkograph, it also makes a set of graphs and calculations based on the linkograph and the FBS codes. These include link index tables, entropy value tables, Markov models and other more general statistics.

Results

A linkograph has been generated for the considered design session. We generated this linkograph using the method discussed above. Namely, we made an audio recording of the session in which the design team and the design teacher had a critical conversation. This audio recording was transcribed in an Excel spreadsheet, segmenting the whole session in design moves. Then, we added the FBS annotations and made the links between the design moves as we saw fit. The full linkograph is available online [24], including the original spreadsheet file and



393	R	I think that you should look closely to your national regulations
394	R	escaping in a normal fashion through your parking area, that is not easy é
395	Be	most often, those parking spaces are in a basement floor
396	R	and in principle, the staircase cannot go entirely to the basement floor
397	F	so a staircase needs to stop on the ground level,
398	F	a doorway to the basement stairs
399	Be	so the question is, how will the fire fighters interpret this here with that elevated ground level?
400	Be	That's a term that is literally given: what is your evacuation level?
401	Be	Is your evacuation level then in that green area?
402	Be	Or is it below in that black area?
403	S	So you see that escape routes are in-between parkings

Fig. 4 A randomly chosen part of the linkograph that is generated by the LINKOgrapher software. On the *right*, the transcript is shown of the diverse design moves. *Left* of these design moves, the FBS qualifications are shown (Be, S, R, etc.). On the extreme *left*, the links between the design moves are indicated with *lines* and *dots*

some of the documents that can be generated for the linkograph using the LINKOgrapher tool. For reference, a part of the linkograph is shown in Fig. 4.

Figure 4 also shows what is meant by the earlier mentioned FBS processes. For instance, the process of going from design move 402 to 403 involves a transition from a design move annotated as Behavior derived from expectations (Be) to a design move annotated as Structure (S), which is considered as a process of Synthesis (Be->S). When considering only the sequence of design moves, without the forelinks and backlinks, one refers to the *syntactic* occurrences of the FBS processes (e.g., 'Synthesis'). Alternatively, one can also consider the *semantic* occurrences of the FBS processes, meaning that not the chronological sequence of design moves is used, but the actual links between the design moves are considered. In the case of Fig. 4, the design moves 400 and 403 can be considered as a semantic occurrence of the FBS process 'Synthesis' (Be->S) and the sequence from move 402 to 403 is not taken into account as a semantic occurrence of an FBS process.

It must be clear that segmenting the transcript in design moves, annotating the design moves, and deciding which design moves are linked, is subject to personal judgement. As Goldschmidt [11] indicates as well, "*the sole criterion used to determine linkage or its absence is common sense*, *in the context of the design task*". In order to minimise the influence of personal judgement, it would be highly valuable if the design process was analysed by a third party as well, so that the conclusions can be further verified. Therefore, we have provided our initial data in [24].

General Statistics

The complete linkograph counts 811 segments or design moves and 4,383 links between those design moves. Each design move is thus linked to about 5.40 other design moves, resulting in a link index (LI) value of 5.40 [11]. This is a high value, considering that Goldschmidt marks a LI value of 0.83 as low and a LI value of 1.73

as high [20]. This might be one of the first differences between a common preliminary design or sketch process and a critical discussion as it is studied here. Namely, the high LI value might be explained by the fact that the studied conversation contains a significant amount of repetition. Initial ideas are coined, both by the design team and the supervising design teacher, and they are continuously referred to by them later on in the conversation when they aim to explain or defend the coined ideas. For more individually oriented design processes, it might be more often the case that designers continuously build on some initial idea and move forward towards a design concept and structure. In a critical design conversation, more effort is invested in finding mutual agreements on the functions, structures and goals that should be reached.

In the online schematic display of the full linkograph [24], an indication is included of the design episodes that were outlined for the considered design process, using the linkograph visualisation and the protocol study contents. Six main design episodes were identified: an introduction episode (moves 0–14); a duplex principles episode (moves 15–177); a shaping the façade episode (moves 178–357); a structural design episode (moves 358–495); a design of the urban context episode (moves 496–779); and a summarizing episode (moves 780–811). When looking at the LI values for these individual design episodes, equally high LI values are found. Namely, the LI values are, in sequential order: 3.21 (episode 0–14); 4.54 (episode 15–177); 3.96 (episode 178–357); 4.36 (episode 358–495); 4.68 (episode 496–779); 3.54 (episode 780–811). These LI values only take into account links that fall entirely within the considered episode and thus do not link to design moves in the other design episodes.

FBS Issue Distribution

Each design move has an FBS code assigned, resulting in the following frequencies for each of the FBS codes (Table 1) and their corresponding processes (Table 2). As can be seen from Table 1, most of the design effort goes to expected behaviour (Be), behaviour derived from structure (Bs) and structure (S), which is to be expected in such a design conversation.

The FBS process distribution (Table 2) can be considered in four ways, of which each is represented by a separate column in Table 2. It is especially important to note the difference between *syntactic* and *semantic* occurrences of FBS processes. Only the latter take into account the existence of links between the design moves: "*B>A is a valid transition process if B is linked back to A in the linkograph*". (Pourmohamadi and Gero - [22]). Therefore, we consider the two rightmost columns in Table 2 as the more significant indicators of the frequencies in which the different FBS processes occur.

As can be concluded from the statistics in Table 2, most attention goes to analysis (SBs) and evaluation (BB), followed by Reformulation I (SS). This image of the design process corresponds to our earlier conclusions based on Table 1:

Table 1 FBS issue distribution over the linkograph

FBS code	Number of occurrences	Percentage of occurrences
R	35	4.3 %
F	75	9.2 %
Be	211	26.0 %
Bs	253	31.2 %
S	217	26.8 %
D	20	2.5 %

Table 2 FBS process distribution over the linkograph

FBS process	Number of syntactic occurrences	Percentage of syntactic occurrences	Number of semantic occurrences	Percentage of semantic occurrences
Formulation (FBe)	23	6.5 %	105	5.5 %
Synthesis (BeS)	41	11.6 %	230	12.1 %
Analysis (SBs)	78	22.1 %	362	19.0 %
Evaluation (BB)	78	22.1 %	538	28.3 %
Documentation (SD)	3	0.8 %	9	0.5 %
Reformulation I (SS)	93	26.3 %	330	17.3 %
Reformulation II (SBe)	31	8.8 %	236	12.4 %
Reformulation III (SF)	6	1.7 %	93	4.9 %

the goal of the conversation is to assess an existing design proposal in order to improve it. Hardly any documentation (SD) is taking place, in which structural design decisions (S) result into explicit documentation.

A more detailed understanding can be found by looking at the distribution of FBS codes over the complete linkograph timeline (window set to 80). In this respect, the graphs shown in Figs. 5, 6, 7, and 8 were generated by the LINKOgrapher software (available also online [24]). In Fig. 5, all six FBS codes are shown on the linkograph timeline, indicating that design moves deal most often with Structure (S), Behaviour derived from structure (Bs) and Expected behaviour (Be) throughout the entire conversation, as indicated before. Additionally, one can notice how diverse 'peaks' appear to be generated by the design moves that focus on Requirements (R), further reinforced by local peaks of design moves that focus on Function (F).

When looking specifically at the design moves that focus on Function (F), these peaks can be distinguished even more clearly. The linkograph data additionally shows that the distribution of design moves focusing on Structure (S) complements the distribution of design moves focusing on Structure (F), Fig. 6. In other words, peaks in the F issue distribution coincide with valleys in the S issue distribution, and vice versa.

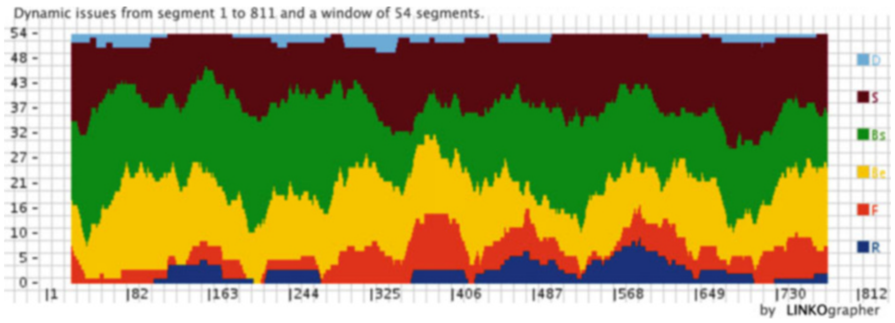


Fig. 5 Overview of the FBS code distribution over the complete linkograph timeline, as it is produced by the LINKOgrapher software (see original image online [24]). From *top* to *bottom*, the following FBS codes are represented: documentation (*D – light blue*); structure (*F – purple*); behaviour derived from structure (*Bs – green*); expected behaviour (*Be – yellow*); function (*F – orange*); requirements (*R – dark blue*)

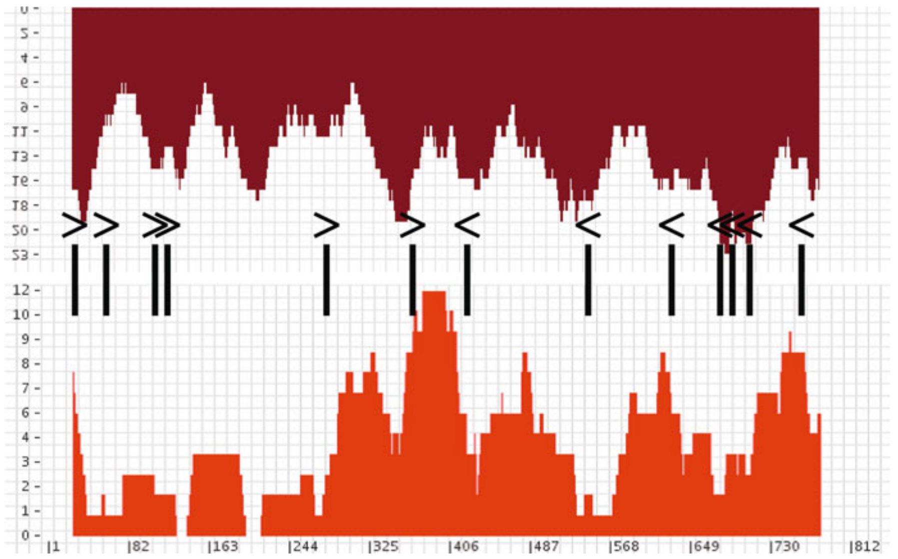


Fig. 6 Distribution of the design moves that focus on Function (*F – below*) and Structure (*S – above*) (see original image online [24]). Also, critical forward links (>) and critical backward links (<) are indicated

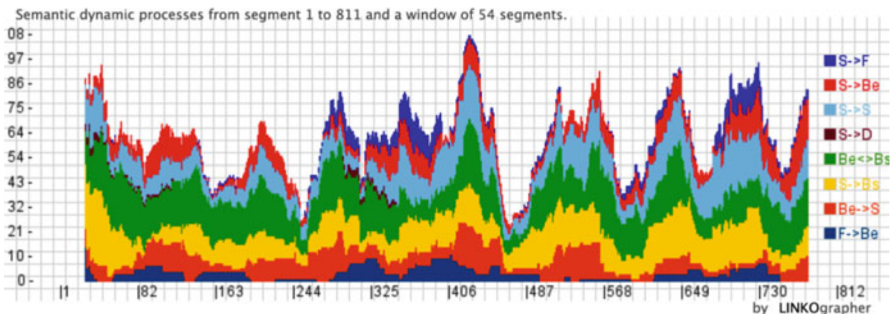


Fig. 7 Distribution of the FBS processes throughout the linkograph timeline (See original image online [24])

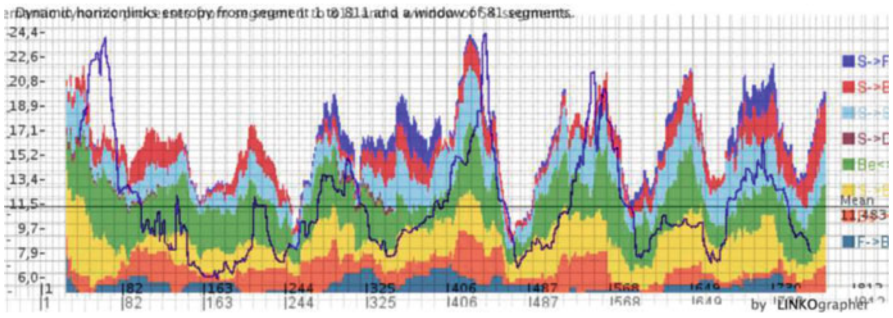


Fig. 8 Horizonlink entropy evolution over the linkograph timeline, overlaid with the FBS processes graph as it was given earlier in Fig. 7 (See original image online [24])

Table 3 Critical design moves in terms of forelinks, with an indication of the design move number, the content of the design move, and the assigned FBS code

Number	Content	FBS code
26	We have two areas for circulation	S
70	I had hoped that you would have reached something using that duplex principle	Be
131	Yes we thought to create contrasts	S
135	But the form is 'created', so to speak	Bs
288	So that the floor plan determines the form and the look of that tower	S
373	That we parked at the bottom at ground level, and that the entrance to the building	S

From this observation, one can conclude that the design decision process appears to start at the appearance of a certain requirement (R), or even more prominently, the appearance of a desirable functionality (F). Based on that, certain evaluations and analyses are made (Be – Bs), eventually leading to certain (ad hoc) decisions regarding Structure (S). These (ad hoc) decisions do not lead to documentation in the current design conversation, but supposedly, they will lead to documentation in the time period following this conversation, when the design students go back to their more individual design environments. In Fig. 7, an overview is given of the dynamic FBS processes occurring in the design conversation, showing peaks in the rightmost part of the graph that coincide with the rightmost peaks in Fig. 7 and the associated critical moves.

Critical Design Moves

Critical moves (CM) can be distinguished using the number of backlinks and/or forelinks starting at specific design moves. In terms of forelinks, design moves 26, 70, 131, 135, 288, 373 are the most critical (Table 3). These design moves indeed correspond to design ideas and intentions to which are often referred during the conversation and seem to steer the ideation process.

Table 4 Critical design moves in terms of backlinks, with an indication of the design move number, the content of the design move, and the assigned FBS code

Number	Content	FBS code
410	Come on, you are thinking so much about those duplexes and so forth	Bs
555	Shouldn't you attach entrance points to circulation and program?	Be
643	If you don't have the freedom to say: "on the corners where we think that such a connection is feasible, we will replace the apartment by some collective area"	Bs
686	I am curious though to the way in which those different constellations give form to that park	Bs
712	Don't you have anything else to do at the ground level of a tower besides placing pilotis between which cars are driving?	Be
721	It is obvious that a discourse is emerging about the ground level that is not yet fully designed	Bs
727	But, in that case, you expect something in terms of functionality	Be
759	The way in which you are handling the living units	S

In terms of backlinks, design moves 410, 555, 643, 686, 712, 721, 727, 759 are the most critical (Table 4). As can be seen in Table 4, these critical moves are most often assigned a Bs or Be code, which indicates that they are often of a evaluative or analytic nature, in contrast to the critical design moves in Table 3. Indeed, these design moves correspond to statements that can be considered as key in the *evaluation* of the current design.

The critical moves in terms of backward links (Table 4) can easily be recognised in the FBS processes graph in Fig. 7, as they coincide with the peaks at the right of this graph. In other words, these design moves are the end points of many of the links, which are interpreted as FBS processes in the graph of Fig. 7. They represent the key comments or conclusions for the design process. Critical design moves in terms of forward links (Table 3) can less easily be recognised in the FBS processes graph in Fig. 7. This is to be expected, as these design moves represent the starting points of such processes, which is not what is shown in this graph.

Critical design moves in general furthermore appear to coincide with the peaks in the distribution shown above in Fig. 6. This distribution represents the evolving number of design moves that are annotated as Structure (S). To conclude, the critical design moves thus represent the key structural elements in the presented architectural design, with the critical moves pointing backwards having an additional conclusive and evaluative character.

Entropy Evolution

As indicated above, entropy measures provide an alternative way to analyse the productivity of the design process (see also [17, 19]). By using entropy to characterise the links in the linkograph, an assessment can be made of the extent to which a design move is unexpected or surprising in the whole of the design process. As stated by Kan and Gero [19], "*information can then be defined in relation to the surprise it produces or the decrease in uncertainty*".

In this analysis, we will use the *horizonlink entropy* indicator that is produced by the LINKOgrapher software, following the calculation procedure documented by Kan and Gero [19]. A horizonlink is a different kind of 'link' than a forelink or a backlink in a linkograph. It is not an explicit link; rather, a horizonlink is a measure of the *distances* of links in a certain part of the linkograph. It is stated by Kan and Gero [19] that design moves are more likely part of a short term 'working memory' process, when they have a small horizonlink indicator, because they only have short-distance links. Design moves with a high horizonlink indicator include long-distance links, which are interpreted as 'incubated moves' [19]. Those links refer to reflection in action [19, 25]. We follow here the interpretation by Kan and Gero [19] that "*a good design process contains unsaturated short links plus a number of long links*". When using the entropy measure of horizonlinks, we have an indication of the unpredictability and 'chaos' that is present in certain portions of the linkograph. A low entropy measure indicates that the linkograph is either fully saturated (1) or completely without any links (0). In both cases, the entropy is 0. A high entropy measure indicates that the linkograph has an unpredictable and seemingly random structure. According to Kan and Gero [19], this feature indicates a process in which more 'opportunity for idea development' is present and which can thus be considered more productive.

The entropy evolution for the considered case study is given in Fig. 8, overlaid with the FBS processes graph that was given earlier in Fig. 7. This graph clearly shows a number of peaks, in which the entropy indicator maximizes temporally. These peaks coincide with the peaks that were encountered in the FBS processes graph (Fig. 7). The rightmost peaks indicate the points where the critical design moves were also found.

For the rightmost design moves in the current case study, both the horizonlink entropy indicator and the CM indicator thus point to the same (four) regions as highly productive. These peaks indicate the design episodes in which the conversation deals with the structural design and the design of the urban context of the tower, including the design of the parking spaces, public spaces and the park area. Indeed, there was quite some more discussion and less initial agreement about these topics. As a result, more opportunity for idea development is present in these design episodes. The leftmost design moves have less critical moves in terms of backward links. Also in terms of entropy, this region appears to be less 'productive'.

'Less productive' design episodes were found in the leftmost portion of the conversation, apart from the peak at the very beginning of the design conversation. The initial entropy peak can be explained as follows. In the beginning, the student design team presents the main ideas behind their design decision of the past week. References are made to these ideas from very diverse episodes in the design conversation. Hence, they are very valuable for the production of ideas. In the following period, which shows a lower overall entropy value, initial comments and questions are given regarding the design, providing the option to the design teacher to understand the reasoning behind the presented design decisions. Little new ideas are produced in this part of the process, also because it incorporates more agreement about the good points of the presented design.

Conclusion

Using linkographs, we have analysed a conversation in which a design team presents their design to a design teacher and receives feedback and remarks regarding their design decisions. By doing so, we give an idea of how conversation and critique can be interpreted as important parts of an architectural design process. The resulting linkograph, and the associated statistics, resulted in the following conclusions.

The link index of the resulting linkograph is high, indicating many links between the design moves and a rather dense conversation. According to Goldschmidt [20], this is an indication of a productive design process. However, it might also indicate here that designers involved in conversation and critique tend to keep referring to the same ideas, over and over again, in order to persuade the one or the other of a certain element/design move that should be included or excluded. This would indicate that the character of conversation and critique is considerably different from a traditional design session, in the sense that more critical features of the design are questioned, requiring the people involved in the conversation and critique to revisit these critical features over and over and evaluate them again and again. This can make sense in the current context of conversation and critique, as the student design team has been working on their design for about a week, working in a specific direction, and they are now returning to the design teacher, who needs to question the sometimes drastic design decisions taken. This conclusion is in line with the considerations made at the outset of this article (section "Introduction"), where it is presumed that conversation and critique tend to focus more on finding mutual agreement on concepts and ideas. The conclusion is further confirmed by the finding that design moves in the analysed conversation deal most often with structure (S), behavior derived from structure (Bs) and expected behavior (Be). Critical features in the design (S) are continuously evaluated in terms of what they are meant for (Be) and what they achieve in the design (Bs).

Analysing the conversation in terms of entropy measures indicates that the first part of the conversation is 'less productive' in terms of idea development [17], in contrast to the second part of the conversation, which has higher entropy peaks at the points where critical design moves are found. Also, the entropy peaks appear to coincide with the points where critical design moves are found, which are typically those points in the conversation where less agreement is found between the design teacher and the design team. So, considering the interpretation of entropy by Kan and Gero [19], those points where the two partners in the conversation and critique disagree more fundamentally, are actually the points with the highest degree of 'design productivity'.

These two main findings of conversation and critique in architectural design (the high link index, and the importance of significant disagreement) provide very relevant feedback to designers, design students and design teachers. It is namely not only concluded that conversation and critique are highly productive, it is also concluded that they are so productive *because* they provide alternative and important opportunities for profound disagreement and questioning of the most basic concepts. So, first, conversation and critique are media of considerable value in design thinking, and these media should be maximally used instead of avoided by any designer. Second, in order for a design critique to remain as impacting and efficient as possible, not only for students, it is highly important that a critical eye is maintained and that disagreement is almost intentionally sought.

References

1. Pauwels P, De Meyer R, Van Campenhout J (2013) Design thinking support: information systems vs. reasoning. Des Issues 29(2):42–59
2. Bilda Z, Gero J (2008) Idea development can occur using imagery only. In: Gero JS, Goel AK (eds) Design computing and cognition'08, Springer (pp. 303–320).
3. Lymer G, Lindwall O, Ivarsson J (2011) Space and discource interleaved: intertextuality and interpretation in the education of architects. Soc Semiot 21:197–217
4. Goldschmidt G (1990) Linkography: assessing design productivity. In: Trappl R (ed) Cybernetics and systems'90. World Scientific, Singapore, pp 291–298
5. Gero J (1990) Design prototypes: a knowledge representation schema for design. AI Mag 11:26–36
6. Ericsson K, Simon H (1993) Protocol analysis: verbal reports as data. MIT Press, Cambridge
7. Cross N (2001) Design cognition: results from protocol and other empirical studies of design activity. In: Eastman C, McCracken W, Newstetter W (eds) Design knowing and learning: cognition in design education. Elsevier, Oxford, pp 79–104
8. van Someren M, Barnard Y, Sandberg J (1994) The think aloud method: a practical guide to modelling cognitive processes. Academic, San Diego
9. Ennis C, Gyeszly S (1991) Protocol analysis of the engineering systems design process. Res Eng Des 3:15–22
10. Kavakli M, Gero J (2002) The structure of concurrent cognitive actions: a case study of novice and expert designers. Des Stud 23:25–40
11. Goldschmidt G (1995) The designer as a team of one. Des Issues 16:189–209
12. Goldschmidt G (1997) Capturing indeterminism: representation in the design problem space. Des Stud 18:441–445
13. Goldschmidt G (2003) The backtalk of self-generated sketches. Des Issues 19:72–88
14. Van der Lugt R (2003) Relating the quality of the idea generation process to the quality of the resulting design ideas. In Proceedings of the 18th International Conference on Engineering Design (pp. 19–21).
15. Kvan T, Gao S (2005) Examining learning in multiple settings. In: Martens B, Brown A (eds) Learning from the past – a foundation for the future. Springer, Dordrecht, pp 187–196
16. Kan JWT, Gero JS (2005) Can entropy indicate the richness of idea generation in team designing? In Proceedings of the CAADRIA05 Conference (pp. 451–457).
17. Kan JWT, Bilda Z, Gero JS (2006) Comparing entropy measures of idea links in design protocols. In: Gero JS (ed) Design computing and cognition'06, Springer (pp. 265–284).

18. Kan JWT, Gero JS, Tang H (2011) Measuring cognitive design activity changes during an industry team brainstorming session. In: Gero JS (ed) Design computing and cognition'10, Springer (pp. 621–640).
19. Kan JWT, Gero JS (2008) Acquiring information from linkography in protocol studies of designing. Des Stud 29:315–337
20. Goldschmidt G (1992) Criteria for design evaluation: a process-oriented paradigm. In: Kalay Y (ed) Evaluating and predicting design performance. Wiley, New York, pp 67–79
21. Shannon CE (1948) A mathematical theory of communication. Bell Syst Tech J 27:397–423
22. Pourmohamadi M, Gero JS. (2011) LINKOgrapher: an analysis tool to study design protocols based on FBS coding scheme. In International Conference on Engineering Design
23. Gero JS, Kan JWT, Pourmohamadi M (2011) Analysing design protocols: development of methods and tools. In International Conference on Research into Design.
24. Pauwels P, Strobbe T, Verboven J, De Meyer R (2014) Additional Data DCC2014 Article. http://users.ugent.be/~pipauwel/DCC2014_additional-data.html.
25. Schön D (1983) The reflective practitioner: how professionals think in action. Basic Books, New York

Part III
Design Creativity

Mental Models and Creativity in Engineering and Architectural Design Teams

Hernan Casakin and Petra Badke-Schaub

Abstract Mental models play a decisive role when it comes to cooperate and coordinate team activities in complex environments and contexts. However, scientific knowledge about the coordination of mental models in heterogeneous groups, and even more across different disciplines, has not yield much progress in the last decades. Mental models affect design activities on content and process levels. These have consequences for the different phases in the design process, from the first moment of defining the problem till the final decision for detail design. The present paper focuses on the comparison of two different design disciplines, and analyzes how problems demanding creativity are approached. Two meetings of engineering and architectural teams solving a complex domain-specific design problem in the very early stage of idea generation were studied. Utterances of the transcripts from both team sessions have been explored based on the categorisation system developed explicitly for the analysis of group behaviour in complex environments. Qualitative and quantitative analyses are presented, from which conclusions about the differences in design problem solving processes of design teams with different disciplinary backgrounds are offered.

Introduction

Design problem solving can be defined as a complex activity involving a series of adaptive and generative steps such as problem definition, collection of different kinds of information, generation and analysis of solution ideas, selection and implementation of innovative solutions [1], to arrive at a certain specified outcome. For the sake of enhancing opportunities in the early design phase and to produce creative design solutions, design teams should strive for exploring different

H. Casakin (✉)
Ariel University, Ariel, Israel
e-mail: casakin@bezeqint.ne

P. Badke-Schaub
Delft University of Technology, The Netherlands

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_9

alternatives, and avoid discarding ideas prematurely [2]. Most of these activities have to be coordinated and communicated in different social settings such as face-to-face interaction with suppliers, negotiations with the selling department, different kinds of remote interaction with clients and other stakeholders, whereby designers often act as individual designer, but in responsibility for the whole project team and the company. The specific characteristics of the setting determine how information is collected, shared and used. Studying mental models can help gaining insight into basic processes of team coordination and team behavior [3]. Different research methodological approaches, such as comprehensive field studies [4] as well as laboratory settings [5] have explored phenomena of design teams acting in different contexts, e.g., how teams deal with different types of critical situations, and how they use information sources. Still, the detailed process of how mental models develop and influence creativity and decision making processes in design teams need to be addressed in order to support design teams.

Thus, the main goal of the study is to present an approach through which mental models are analyzed with an emphasis on creative design activities in different design disciplines. Here, the focus is set on cognitive and social behavior in engineering and architectural design teams.

General and/or Specific Knowledge Generates Creativity

What is the importance of studying activities within different design disciplines? In essence, we hope that the answer to this question provides knowledge on two different levels. First, we can gain basic knowledge about the phenomenon of creativity and second, we can build on this knowledge, and designers (students as well as practitioners) can be taught accordingly.

There are two concepts to be distinguished, which take up the origin of creativity as either caused by domain-general or by domain-specific abilities. There are empirical studies providing evidence that people with general creative thinking abilities are capable of generating creative ideas across diverse domains [6, 7]. However, other studies show that the generation of ideas requires the availability of domain specific knowledge and skills; see [8] for a comprehensive summary on the subject. Apart from these contradicting approaches, there is a third conceptualization that sees the combination of both, domain-general and domain-specific skills as contributing to the individual creative competence; see Fig. 1. Domain-general creativity encompasses knowledge of general problem solving strategies [9] and general heuristics [10] including reflecting activities. Domain-specific creativity, on the other hand, resort to factual knowledge about content and process (including design methods), as well as design specific skills such as sketching. Apart from knowledge and skills, the final creative product largely depends on the individual ability to generate, communicate, and coordinate own ideas within the social context; see Fig. 1.

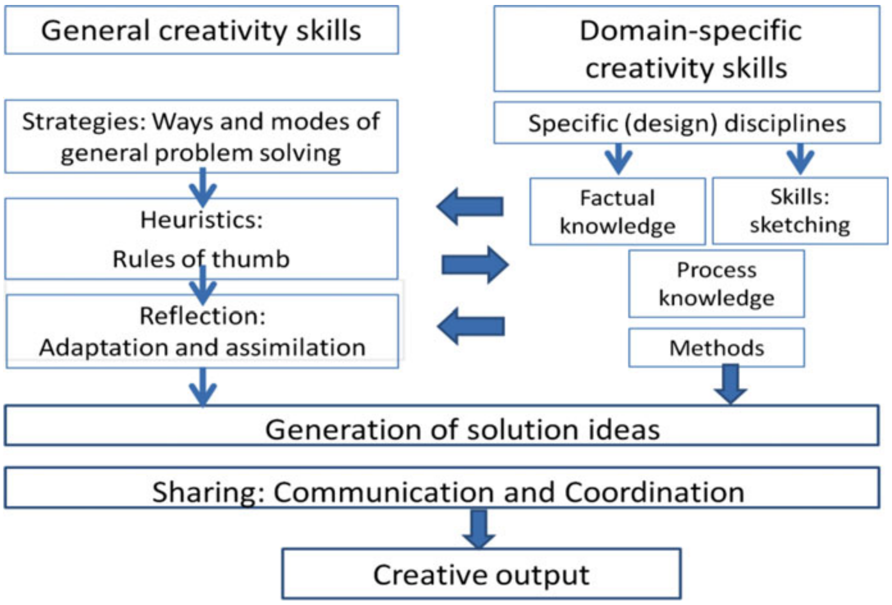

General creativity skills
Strategies: Ways and modes of general problem solving
Heuristics: Rules of thumb
Reflection: Adaptation and assimilation
Domain-specific creativity skills
Specific (design) disciplines
Factual knowledge
Skills: sketching
Process knowledge
Methods
Generation of solution ideas
Sharing: Communication and Coordination
Creative output

Fig. 1 A theoretical framework of the determinants of creativity

Idea Generation and Design Creativity

Designing is characterized by the generation of ideas and solution principles, mainly in the conceptual design phase [11]. The measurement of creativity is – as the definition of creativity – not unanimously defined. There are studies which use only the number of ideas produced. Some studies also rate the originality of ideas, their novelty and usefulness [12] or a combination of them [13, 14]. Shah and Vargas Hernandez [15] propose novelty, variety, quality and quantity as measures of ideation effectiveness.

In terms of creativity, idea generation is seen as the most influential activity since it largely affects the subsequent stages of the design process, including the design decisions being taken [16], and the final outcome [17]. In this process, a designer, or a team of designers work together with the purpose of developing not only functional, but also innovative and creative concepts. Typically, designers generate, give meaning, and interpret several ideas in parallel [18], whereas they explore further directions for clarifying uncertainties and developing potential problem solutions [19]. The generation of new ideas is on a physiological level an outcome of association processes [20]. Idea association is not necessarily promoting a high diversity of ideas [21]. Whereas the design process entails the evolution of a variety of ideas, idea association can help establishing relations [22, 23], and generating new design ideas in the conceptual stage [24]. While looking for alternative design ideas, creative designers enlarge the metaphorical search space of promising idea-solutions [25], enhancing by this the chances of creating better outcomes [26].

According to design methodology [18, 27], designing is the process of working through a defined series of steps that are defined as useful to arrive at the final outcome successfully. Thus, the idea generation process should be preceded by a problem definition stage, in which the design task/problem is structured and framed, and followed by idea-solution analysis, explanation, and evaluation stages, where solutions are developed, clarified, and assessed with regard to their suitability to the design goals. If the idea-solution or a part of it is found to be satisfactory, then design decisions are made. All these stages are considered to be iterative and cyclical, moving from converging thinking stages – in which concepts and ideas are evaluated and selected, to diverging thinking – where alternative ideas are generated, and vice versa [28]. While this process is believed to be characteristic for general design processes, there is no empirical evidence whether it can be considered to be similar across different design domains.

Mental Models and Creativity in Design Teams

A major approach to theorize and study mental representations in design is the concept of mental models. This theoretical construct enables exploring creative cognitive activities. As internal representations, mental models provide the basic conditions for human beings dealing with the environment, and guide their acts [29]. Since mental models can describe and represent thought processes in problem solving, they can aid predicting and offering explanatory power of how individuals will perform and behave in a specific situation [30]. Thus, mental models can be defined as simplified representations of the world [31] that individuals construct, and adapt for attaining fast performing acts, as well as for gaining and processing new information [32]. The manner in which mental models are developed hinges on the context, and social setting in which they are constructed [33].

Individual mental models are the ingredients of the team mental model which is developed in a team and have a strong influence on team communication and performance [3]. The way team members perceive and understand reality can vary according to their personal background, knowledge, expertise, etc., and these have an effect on their mental models. Team mental models are dependent on the individuals' input, and guide the team how to proceed in terms of process and content. A main characteristic of mental models is that they can aid to coordinate and adapt actions, as demanded by the task, and the team members [34]. As the design progresses, team members interact with each other to exchange opinions and ideas, while their mental models are modified, adapted, and eventually shared within the team. These constructs also direct the behavior of a design team when facing new and unknown situations.

Following the assumption that mental models are partly built from domain-specific knowledge, and partly from not domain-specific procedures, the question is what the differences of mental models constructed by teams in different design disciplines such as engineering and architecture are, and what might be their effect on design creativity.

Types of Mental Models

In general, literature about mental models in teams refers to three major types of representations which are the ‘task mental model’, the ‘process mental model’, and the ‘team mental model’ [35]. The task mental model represents aspects about the facts of the problem at hand. Issues affecting the extent to which task mental models are communicated within the team include making representations of the problem task, defining the problem, generation of ideas, production of explanations and clarifications, as well as analyzes and evaluations of solutions, and taking decisions [32].

The successful achievement of a design outcome also embraces appropriate team coordination, which has to do with a comprehension about the process. Mental models of the process are referred to aspects about rules, strategies, and procedures that need to be considered in order to achieve goals, and arrive at a satisfying outcome. A characteristic of creative problem solving is that there are no clear procedures or routines referring to how design teams should work in collaboration, and how they could organize their processes. Consequently, the selection and application of procedures for dealing with a design task has to be decided as the process develops [35]. These involve a need of information exchange about planning strategies (in what moment to proceed and what to do), procedures (in what way to proceed, as well as which methods to use), and reflection (what the team has achieved so far, and how it should proceed in the coming steps) [32].

Finally, team mental models reflect representations applicable to the way that team members work collaboratively as a group. This mental model is an indicator of the extent to which members are motivated to collaborate, and feel part of the team. Badke-Schaub et al. [32] further focus on team cohesion, which represents the mutual positive feeling in the team reached by the team when dealing with a design task. In terms of activities it includes: appreciation and rejection, referring to the approval or disapproval of other team members or their respective contributions; confirmation, which is a positive evaluation supporting the maintenance of a communication channel among team members; and help, which is the assistance provided by team members to each other. This concept of team cohesion mental model was used in the present paper to study design team activities.

Previous studies have shown that mental models can contribute to gain a better understanding of processes related to team coordination and team behavior [3]. Despite a huge amount on empirical studies on how designers think and behave in real [4] and in a laboratory environment [2, 5]. The way that mental models are used in design in general, and in different design domains in particular, as well as the relation of mental models with design creativity is not well understood yet.

The Empirical Study

Goals of the Study

The study aimed to explore possible commonalities and differences of design team activities across two design disciplines: engineering design and architecture. It provides further insights into the basic cognitive and social processes of the observed teams, by analyzing the transitions of the task, process, and team cohesion mental models developed along the design meetings. In particular, it centers on possible differences regarding the frequencies of the transitions of these mental models to discover whether the design groups show a strategic or a more opportunistic approach. Another goal is to gain a more detailed insight about creativity in engineering and architectural teams. In order to understand the interrelation between mental models and design creativity, the focus is set on the analysis of the frequencies of the transitions between design ideas produced in each team, and the specific design activities corresponding to the different mental models. In this way, it is intended to unveil whether new design ideas could be either preceded or followed by predictable design steps showing a systematic pattern of behavior of the team.

Data Collection and Data Coding

The data material has been collected from two case studies – an architectural and an engineering team – in the context of the Design Thinking and Research Symposium, DTRS [36]. The meetings were videotaped, transcribed, parsed into utterances, and coded with regard to a categorization system presented in Table 1. The analysis explored the different types of communication exchanges developed among the team members. The categorization system was organized into three main groups: task, process, and team cohesion where each of these was divided into subcategories. Mangold InterAct (version 9.3.5 http://www.mangold.de) software was used for information coding. This software program supports the coding and rendering of behavioral data per time unit. The first author acted as the main coder. In order to check coding consistency, the second author assessed 30 % of the data independently. All coding categories received acceptable levels of reliability (i.e., Kappa coefficients for inter-coder reliability larger than 0.72).

Architectural Design Meeting

The architectural task dealt with the design of a new municipal crematorium to be located close to an existent one. The brief included a series of functions such as

Table 1 Categorization system for verbal activities in engineering and architectural teams

	Task mental model
Problem definition	Definitions that are mentioned in order to define the problem
New solution idea or new solution aspect	Stating a new idea or a new solution for a problem or a sub-problem, or new aspects of an earlier solution idea
Solution analysis	Analysis of characteristics and potential application of a solution idea
Solution evaluation	Evaluation of a solution idea by assessing its value and feasibility
Explanation	Clarification of aspects and questions related to design issues, i.e., user, technical, budget
Solution decision	A final and definitive decision
	Process mental model
Planning	Aspects related to when to proceed, what to do, and who does it
Procedure	How to proceed to approach the task, strategies which methods may be used
Reflection	What the team has been doing so far and what variables have shown influence
	Team cohesion mental model
Appreciation	Approval of other team members supporting an idea, an explanation or a problem definition
Confirmation	Positive statements confirming other team members' statements
Rejection	Disapproval of other team members about an idea, an explanation or a problem definition
Help	Aid or assistance provided to other team members

cremation facilities, waiting rooms, vestry, parking areas, and a chapel intended for up to 100 people. The architectural team was composed of a municipal architect, the manager of the existing facility, and an officer from the local government on behalf of the municipality.

Engineering Design Meeting

The engineering task was concerned with the design of a new digital pen using novel print-head technology. The pen had to be devised as a kind of artist's tool or as a toy. The design issues discussed in the meeting centered on functional aspects dealing with electronics and software, as well as with features of the pen. Seven members from a technology development company formed the engineering team: a business consultant in the role of a group moderator, an expert in electronics and business development, and another in ergonomics and usability issues, three mechanical engineers, and an industrial design student.

Results

In this section, we present results of the data analysis. First, the analysis of communication provides insights into main differences of mental models in the architectural and engineering groups. Second, transitions of design activities in the two disciplines, and their relation to creativity are analyzed.

Analysis of Frequencies and Communicative Acts in Regard to the Three Mental Models

Figure 2 shows the cumulative frequencies of design activities per design team related to the defined mental models: *Task*, *Process*, and *Team cohesion*. There were a total of 2,256 utterances, 54 % of which corresponded to *Task* activities, 10 % to *Process*, 27 % to *Team cohesion*, and 9 % to other activities. This overview shows that the design activities relating to the *Task* mental model play a key role in both teams, followed by activities in *Team cohesion*.

Table 2 summarizes the cumulative frequencies of design activities per design team according to the mental models sub-categories. From the table can be seen that *Task* activities were mainly characterized by *Solution analysis* and *Explanations* in both teams, and by *Evaluations* in the engineering one. *Procedures* and *Reflections* were dominant in the *Process* activities of the architectural team. Regarding *Team*

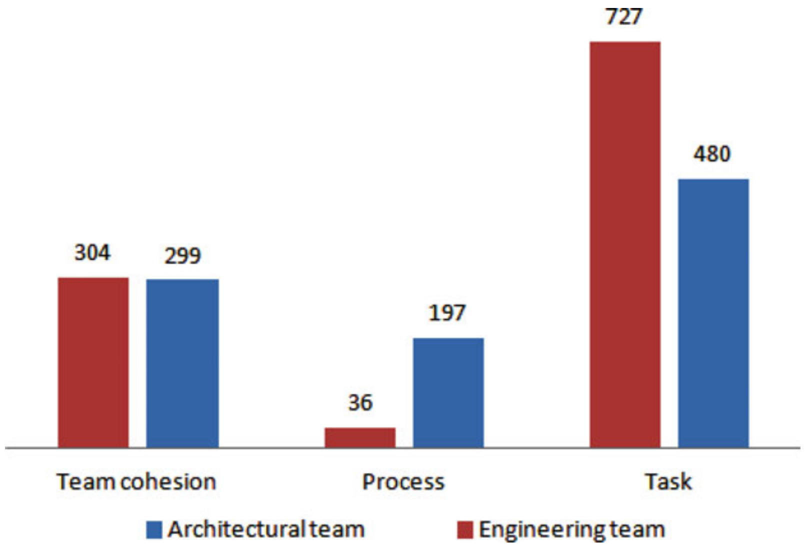


Fig. 2 Cumulative frequencies of design activities per design team

Table 2 Cumulative frequencies of design activities per design team according to mental models sub-categories

Categories		Architectural team	Engineering team
Task mental model	Problem definition	20	90
	New solution idea	36	111
	Solution analysis	285	218
	Solution evaluation	9	122
	Explanations	114	172
	Solution decision	16	14
Process mental model	Planning	51	0
	Procedure	74	19
	Reflection	72	17
Team cohesion mental Model	Appreciation	28	5
	Confirmation	262	287
	Rejection	5	9
	Help	4	3

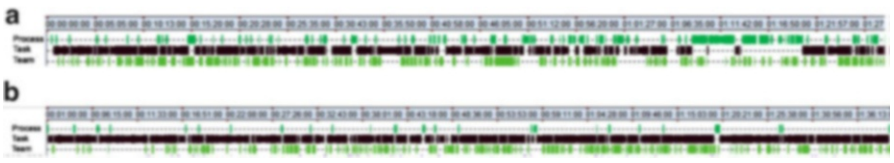


Fig. 3 Activity focus related to utterances in regard to mental models developed over the course of the design meetings– (**a**) architectural team; (**b**) engineering team

cohesion, *Confirmations* were central in both groups, whereas *Appreciations* were dominant in the architectural one.

Figure 3 depicts the sequence of acts in the two teams with regard to the communication issues across the three mental models of *Task*, *Process*, and *Team cohesion* over the complete period of the design meeting. A qualitative analysis indicates that the engineering team largely focused on activities concerned with *Task* and *Team cohesion* along the whole design process. While contributions related to the *Task* were the most dominant activity of this group, much less attention was spent on aspects dealing with the *Process*.

The architectural team, on the other hand, also dedicated their efforts to the design *Task* – albeit to a less extent than the engineers- and to the design *Process*. This activity was constant along the whole period of work, only increasing near the final stages of the meeting. This group also maintained a good *Team cohesion* all over the process. In sum, both teams progressed in their work mainly by focusing on design problem content, supported by a positive atmosphere aiming at mutual understanding. Procedural aspects were also relevant in the architectural session, probably to advance the exchange of communication acts between team members for implementing design ideas and solutions in practice.

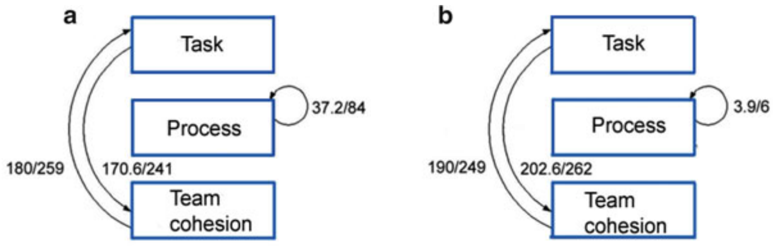


Fig. 4 Transitions between mental model categories – (**a**) architectural team; (**b**) engineering team

Transition Steps in Architectural and Engineering Teams

Transitions between design activities in relation to *Task*, *Process*, and *Team Cohesion mental models* were examined to find out whether combined phases of activities – patterns – could be observed. It would be interesting to see which group shows a more strategic approach; or whether the groups choose a more opportunistic approach, and change often between the different design activities. Thus, this analysis might also provide evidence in how far the design process is characterized by an unpredictable sequence of design activities, or whether there are certain patterns of behavior that appear systematically. In order to answer this question, the transition probabilities between all utterances were calculated, and then compared to the related baselines. As a way of exemplification: if *Task* utterances occur in 50 % of all team communication, but after a *Task* verbalization in 62 % of all cases a *Process* verbalization takes place, this implies that there will be a higher probability that sequences of *Task-Process* verbalization will occur more often than *Task-Task* utterances. A chi-squared test was used for calculating whether the observed transition probability is significantly higher in comparison to the baseline of the categories.

Figure 4 depicts transition probabilities of the three mental models of *Task*, *Process*, and *Team cohesion*, in the two teams. A connection that ends with an arrow represents a transition that is significantly more likely to occur compared to the expected count. The first number aside the connection represents the expected count, and the second number refers to the transition count. As it can be seen in Fig. 4, in the two groups a transition within the same pattern of behavior is highly likely.

Mental Models and Creativity in Architectural and Engineering Teams

We further explored the mental models in each design team, with a particular focus on design creativity. In the present study, we defined and measured creativity as the

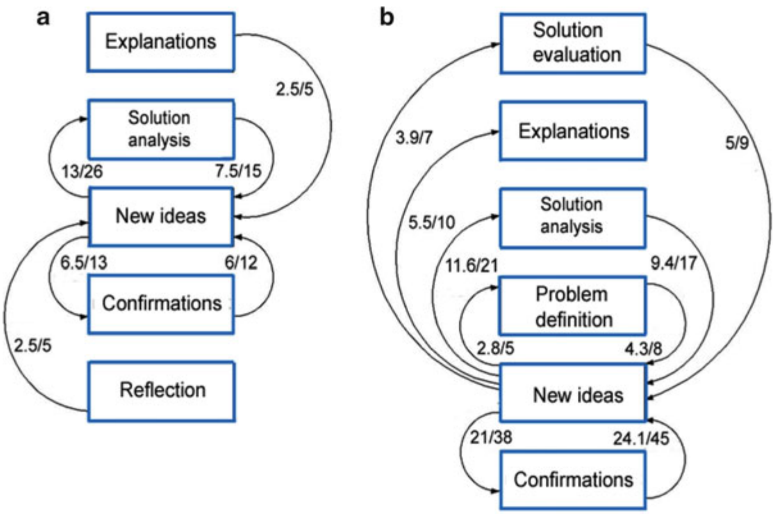


Fig. 5 Transitions between design activities– (**a**) architectural team; (**b**) engineering team

number of ideas generated by the team during the design meeting. Accordingly, we investigated if the generation of design ideas might be either preceded or followed by predictable design steps reflecting certain systematic pattern of behavior. Thus, we analyzed the transitions of utterances between the *New ideas* generated during the design activity, and the other design activities. The transition probabilities between all of the activities were calculated, and compared to the baselines of the activities. For the entire sessions, a chi-squared test of independence showed that the observed frequencies were significantly different than the expected ones for both design teams, chi^2 (14, 2256) = 212, $p < 0.001$, two tailed).

Figure 5 shows transition probabilities between utterances related to *New ideas* and design activities. In the architectural team, *Confirmations*, *Solution analysis*, *Reflections*, and *Clarifications* were the most frequent steps that preceded the generation of *New ideas* ($p < 0.05$*, and* $p < 0.001$, two tailed). On the other hand, *New ideas* were followed by *Confirmations* and *Solution analysis* ($p < 0.001$, two tailed).

In the engineering team, *New ideas* were preceded by *Problem definitions*, *Solution analysis*, *Solution evaluations*, as well as *Confirmations* ($p < 0.05$*,* $p < 0.01$, *and* $p < 0.001$, two tailed). Likewise, *New ideas* were continued by further *Problem definitions*, *Confirmations*, *Solution analysis*, *Explanations*, and *Solution evaluations* ($p < 0.05$*,* $p < 0.01$, *and* $p < 0.001$, two tailed); see Fig. 6.

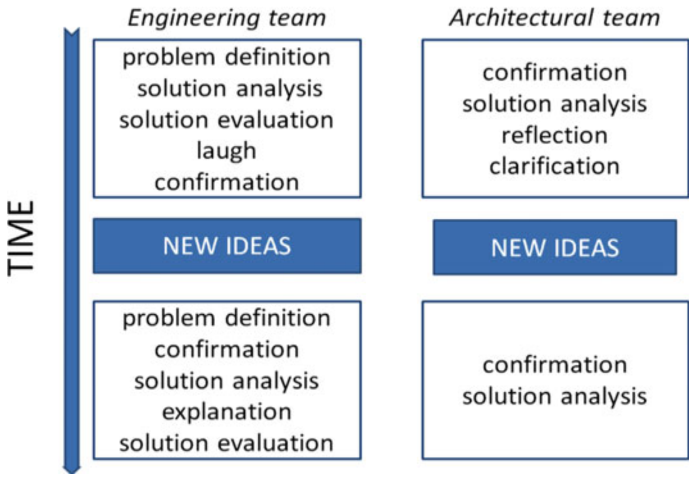


Fig. 6 Transitions of design activities before and after the generation of design ideas: engineering and architectural teams

Discussion

Mental Models in Architectural and Engineering Teams

Considering the exploratory nature of the study and the small number of groups, we do not intend to generalize the outcomes observed in the two groups of designers. However, there were some interesting results in regard to the differences in the distribution of the design activities about the mental models in both teams. These suggest that in each team certain design activities were more important than others.

Results about the distribution of design actions of the mental models showed further remarkable differences between the teams. Particularly, *Task* utterances occurred more often in the engineering team, and more *Process* utterances took place in the architectural team. This result indicates that the engineering team dedicated most of their communication efforts to advance activities related to the successful completion of the problem at hand by transmitting task related content. In contrast, the architectural team mainly focused on team coordination [3], and therefore they attempted to gain a better understanding about the process, which included issues related to strategies and procedures [35]. An alternative explanation is that the engineering design process was more structured than the architectural one, and thus larger agreement was attained among team members. On the other hand, the engineering task could have been less clear than the architectural one, and as a result the engineering team needed to focus more on the task, while the architectural team mainly explored and agreed upon the process with which to proceed. Generally speaking, *Task* mental model was the most developed model in both groups as compared to the other ones, indicating the importance of investing on task contents over any other design aspect. On the other hand, both teams showed to strive for reaching a general understanding while dealing with the design problem. Therefore, *Team cohesion* was the second dominant mental model in both design groups.

Further results concerned with the distribution of design activities belonging to the different mental models showed interesting similarities and differences. *Analysis of solutions* and *Explanations* were the most dominant activities in both teams. Whereas *Solution analysis, Appreciations*, *Reflection, Planning, and Procedure* utterances occurred more often in the architectural team, *Problem definition*, *New ideas*, *Explanations*, and *Solution evaluations* were more frequent in the engineering team. These results indicate that the pattern of behavior of the engineers which was the more creative team in terms of the number of ideas produced was largely characterized by the framing of *Design problems*, and the generation of *Explanations*, *Analysis,* and *Evaluations* of solution ideas. Nevertheless, the high number of evaluations in this group is unexpected considering that designers were requested to generate and discuss ideas mainly by brainstorming techniques [37]. Creativity techniques such as brainstorming warn against earlier evaluations of ideas without previous analysis. It is possible and desirable that an even higher number of design ideas would have generated if fewer evaluations have been made at so early stage [38]. What concerns the architectural team, the high number of activities related to *Analysis*, *Reflections*, *Planning*, and *Procedural* aspects can be seen as a core channel used for communicating and exchanging information in this group. This might be due that an aim of the meeting was to present and discuss the development and modifications of the design project.

Mental Models and Transition Steps in Architectural and Engineering Teams

The analysis carried out on the transition of design activities related to *Task*, *Process*, and *Team Cohesion* mental models showed a similar pattern of behavior in the design teams. This finding is remarkable considering the differences in terms of background, domain interest, and creativity between the groups. It was found that once designers in each team intertwined between *Task* and *Team Cohesion*, they tend to remain engaged to this pattern of behavior for several communicative acts, before switching to another behavior. The repeated loop of task and team cohesion seems to reflect a structuring pattern of behavior in the observed teams. These findings reveal that a good understanding between team members is necessary to progress with the design task. A successful completion of the task needs to be both preceded and followed by positive feedback supporting communication among the team members [32]. This may serve to explain the reason that task actions were not recursive (that is not within the same step). In contrast, design steps concerned with process were dominantly recursive, and not directly related to task or team cohesion actions. This means that with regard to *Process*, teams tend to stick to the same communicative behavior and use to spend more than one communicative act on the same activity before switching to another.

Mental Models, Transition Steps, and Creativity in Architectural and Engineering Teams

Given the additional interest of this study on creativity, further analyses were carried out for the transitions of design steps established between design ideas generated during the design activity, and the different design activities of the mental models. One interesting result common to both design teams is that the generation of *New ideas* was related to a loop of transition steps of *Confirmations* and *Solution analyses*. It is noteworthy that when designers in each team intertwined *New ideas* with *Solution analyses* and *Confirmations*, they use to stay engaged to these patterns of behavior for a number of communicative acts before changing to other patterns. These repeated loops, which seem to represent a structuring pattern of creative behavior is known to allow enlarging the metaphorical space of possible solutions [39], and leading at the end to more creative solutions. Such structuring pattern can be associated with domain-general creativity, which encompasses knowledge of general problem solving strategies that are common to both disciplines [9]. In addition to these, a pattern of behavior related to domain-specific creativity was observed in the engineering team, which included repeated loops of *New ideas* with *Problem definitions* and *Evaluations*. From the viewpoint of creativity, the framing of problems can help to an understanding of the design situation, contributing to promote the production of ideas. In turn, the generation of new ideas can lead to the restructuring of the problem from new perspectives, and again to new ideas. Nevertheless, this design behavior seems to be in contradiction with the early evaluation of design ideas. As noted previously, the immediate judgment after the generation of ideas can prevent the production of additional ideas. The premature evaluation of design ideas was noticed by Stempfle and Badke-Schaub [40] in their study on design team communication. These researchers argued that precipitated evaluations can lead to early rejection of ideas that in later stages could prove to be appropriate to solve the problem. Another mistake that can occur as a result of too early evaluation of ideas is the early implementation of solutions that in later stages may show to be unsuitable. Finally, it was also observed that the architectural team spent more time on *Reflections* and *Explanations* that led to design ideas. Considering the heterogeneous background of the team, these types of design activities are necessary to enhance the communication and understanding, and as a trigger to reach agreement between team members.

Conclusions

A main goal of this study was to explore how design teams from different disciplines deal with design problems, mainly in the early phases of design and with special emphasis on creativity. Thus, all utterances during the design process were defined as cognitive design activities belonging to different mental models: task, process, and team cohesion, and their relation to creativity as a collective ability [42]. Due to the small sample size used in this study we do not intend to generalize the outcomes. Despite the limitations, findings showed interesting results regarding the design activities about the mental models in each design discipline. The study also contributed to introduce and illustrate a new approach to analyze mental models in design teams, and their relation to creativity.

Differences but also similarities in design activities identified in the engineering and architectural groups shed light on how teams with different disciplinary background behave and use their knowledge to solve problems [34], and [41] during the design activity. Remarkably, whereas on a higher level many similarities were found between the design teams, large differences existed on a detailed level when focusing on the design activities corresponding to the mental models. Moreover, it was possible to gain insight into what mental models and what design activities can be characterized by patterns of behavior that appear systematically along the design process, and what activities were more opportunistic, and therefore more difficult to predict. On the other hand, further understanding into the relationship between the mental models and design creativity was gained. While a number of similar design actions occurred as new ideas were identified in each team, major differences were also observed in each domain. This reopens the question of whether creativity is domain-general, or domain-specific [6–8]. Results suggest that while some design activities leading to creativity might be shared across the two design disciplines, some patterns of activities were related to each specific design domain.

These findings are important for educational programs, in particular those aiming to promote creativity in teams across the design disciplines. Training teams would enable to deal with those design activities that were not found to be prolific. Similarities and differences in the pattern of behavior observed in the mental models of the two disciplines have to be considered when couching teams with different design backgrounds. Whereas existing differences in design activities should be considered by intervention programs aiming to enhance domain-specific creativity, similarities in design activities would be also important for the promotion of general creativity across the disciplines. Moreover, the predictable sequences will help to understand how teams behave and act when dealing with problems demanding creativity.

References

1. Akin O, Akin C (1998) On the process of creativity in puzzles, inventions, and designs. Autom Constr 7:123–138
2. Badke-Schaub P, Neumann A, Lauche K, Mohammed S (2007) Mental models in design teams: a valid approach to performance in design collaboration? CoDesign 3:5–20
3. Klimoski R, Mohammed S (1994) Team mental model – construct or metaphor? J Manag 20:403–437
4. Badke-Schaub P, Frankenberger E (1999) Analysis of design projects. Des Study 20:465–480
5. Bierhals R, Schuster I, Kohler P, Badke-Schaub P (2007) Shared mental models – linking team cognition and performance. CoDesign 3:75–94
6. Amabile TM (1996) Creativity in context: update to the social psychology of creativity. Westview, Boulder
7. Casakin H, Davidovitch N, Milgram R (2010) Creative thinking as a predictor of creative problem solving in architectural design. Psychol Aesthet Creat Arts 4:31–35
8. Kaufman JC, Baer J (eds) (2005) Creativity across domains: faces of the muse. Erlbaum, Mahwah
9. Dörner D (1996) The logic of failure: recognizing and avoiding error in complex situations. Metropolitan Books, New York
10. Gigerenzer G (2007) Gut feelings: the intelligence of the unconscious. Penguin, New York
11. Pahl G, Beitz W (1996) Engineering design: a systematic approach. Springer–Verlag, London
12. Sarkar P, Chakrabarti A (2011) Assessing design creativity: measure of novelty and usefulness. Des Study 32:348–383
13. Segers N, de Vries B, Achten HH (2005) Do word graphs stimulate design? Des Stud 26:625–647
14. Torrance EP (1974) Torrance tests of creative thinking. Personnel Press, Princeton
15. Shah JJ, Vargas-Hernandez N (2002) Metrics for measuring ideation effectiveness. Des Stud 24:111–134
16. Berliner C, Brimson JA (1988) A cost management for today's advanced manufacturing: the CAM-I conceptual design[M]. Harvard Business School Press, Boston
17. Ulrich KT, Eppinger SD (2003) Product design and development, 3rd edn. McGraw-Hill, Boston
18. Lawson B, Loke SM (1997) Computers, words and pictures. Des Stud 18:171–183
19. Goldschmidt G, Tatsa D (2005) How good are good ideas? Correlates of design creativity. Des Stud 26:593–611
20. Mednick SA (1962) The associative basis of creative process. Psychol Rev 69:220–232
21. Lai IC (2005) Dynamic idea maps: a framework for linking ideas with cases during brainstorming. Int J Archit Comput 3:429–447
22. Casakin H (2005) Design aided by visual displays: a cognitive approach. J Archit Plan Res 22:250–265
23. Casakin H, Goldschmidt G (1999) Expertise and the use of visual analogy: implications for design education. Des Stud 20:153–175
24. Nagai Y, Taura T, Mukai F (2009) Concept blending and dissimilarity factors for creative concept generation process. Des Stud 30:648–675
25. Gero JM (2000) Computational models of innovative and creative design processes. Technol Forecast Soc Chang 64:183–196
26. Chakrabarti A, Bligh TP (1996) An approach to functional synthesis of design concepts: theory, application, and emerging research issues. AI EDAM 10:313–331
27. Badke-Schaub P, Buerschaper C (2001) Creativity and complex problem solving in the social context. In: Allwood CM, Selart M (eds) Decision making: social and creative dimensions. Kluwer, Dordrecht, pp 177–196
28. Liu YC, Bligh T, Chakrabarti A (2003) Towards an 'ideal' approach for concept generation. Des Stud 24:341–355

29. Gentner DA, Stevens AL (1983) Mental models. Lawrence Erlbaum, Hillsdale
30. Norman DA (1983) Some observations on mental models. In: Gentner DA, Stevens AL (eds) Mental models. Erlbaum, Hillsdale, pp 7–14
31. Smyth MM, Collins AF, Morris PE, Levy P (1994) Cognition in action, 2nd edn. Psychology Press, East Sussex
32. Badke-Schaub P, Neumann A, Lauche K (2011) An observation-based method for measuring the sharedness of mental models in teams. In: Boos M, Kolbe M, Kappeler PM, Ellwart T (eds) Coordination in Human and Primate Groups. Springer-Verlag, Berlin, pp 177–197
33. Marshall N (2007) Team mental models in action: a practice-based perspective. CoDesign 3:29–36
34. Cannon-Bowers JA, Salas E, Converse S (1993) Shared mental models in expert team decision making. In: Castellan NJ Jr (ed) Individual and group decision making: current issues. Lawrence Erlbaum, Hillsdale, pp 221–246
35. Edmondson AC, Nembhard IM (2009) Product development and learning in project teams: the challenges are the benefits. J Prod Innov Manag 26:123–138
36. McDonnell J, Lloyd P (eds) (2009) About: designing – analysing design meetings. Taylor and Francis, London
37. Osborn AF (1957) Applied imagination — principles and procedures of creative thinking. Scribner, New York
38. Newell A, Simon HA (1972) Human problem-solving. Prentice- Hall, Englewood Cliffs
39. Sawyer RK (2003) Group creativity: music, theatre, collaboration. Lawrence Erlbaum Associates, Mahwah
40. Stempfle J, Badke-Schaub P (2002) Thinking in design teams – an analysis of team communication. Des Stud 23:473–496
41. Langan-Fox J, Anglim J, Wilson JR (2004) Mental models, team mental models, and performance: process, development, and future directions. Hum Factors Ergon Manuf 14:331–352
42. Guilford JP (1981) Potentiality for creativity. In: Gowan JC, Khatena J, Torance EP (eds) Creativity: its educational implications. Kendall Hunt, Dubuque, pp 1–5

Brainstorming vs. Creative Design Reasoning: A Theory-Driven Experimental Investigation of Novelty, Feasibility and Value of Ideas

Akin O. Kazakci, Thomas Gillier, Gerald Piat, and Armand Hatchuel

Abstract In industrial settings, brainstorming is seen as an effective technique for creativity in innovation processes. However, bulk of research on brainstorming is based on an oversimplified view of the creativity process. Participants are seen as idea generators and the process aims at maximizing the quantity of ideas produced, and the evaluation occurs post-process based on some originality and feasibility criteria. Design theories can help enrich this simplistic process model. The present study reports an experimental investigation of creativity process within the context of real-life design ideation task. Results lead to the rejection of the classical 'quantity breeds quality' hypothesis. Rather, we observe that successful groups are the ones who produce a few original propositions that hold great value for users while looking for ways to make those propositions feasible.

Research Problem: Creativity Beyond Idea Generation

Considering current economical and social challenges at the global scale, innovation is widely considered as a vital component of today's industry. It is generally assumed that creativity is a main component in innovation processes. Based on these premises, an astounding amount of research has been undertaken on creativity techniques and processes since 1950s [1–8]. A technique that has received particular attention and acknowledgement is brainstorming [3]. Brainstorming is considered as an effective technique by professionals in overcoming particularly hard

A.O. Kazakci (✉) • A. Hatchuel
Ecole Nationale Supérieure des Mines de Paris, Paris, France
e-mail: akin.kazakci@minesparistech.fr

T. Gillier
Grenoble Ecole de Management, Grenoble, France

G. Piat
Electricité de France (EDF), Paris, France

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_10

tasks that require creative insights [9]. There is a large room for debates on this issue considering the conflicting results from research [1, 2]

Since the pioneering work of Osborn [3], a significant amount of work has been undertaken to decipher this creative process and to provide control parameters and various extensions [4–6]. Our main claim about the existing research work is that they are working with an oversimplified (and implicit) process model for creativity process. Not only do they lack rich definitions of creative behavior, thus reducing the phenomenon mainly to a simple idea generation phase, they also implicitly assume that the evaluation process can be dissected from the generation phase. Under such hypotheses, naturally, brainstorming research tends to evaluate output of a creative process mainly by the quantity of ideas produced, making the assumption that quantity breed quality [3].

Compared to brainstorming research, design research has focused in understanding the dynamics of creativity within design processes [10–14]. Arguably, design processes are most significant creative processes driving economic and social innovations [15]. As design research progresses towards maturity, several formal models and theories has been produced describing design reasoning leading to creativity [16–24]. The paper defends the thesis that such models would enable better-designed experiments to understand creative processes and would allow new predictions to be tested.

To demonstrate this concept, the current work reports an on-going exploratory investigation of a real-world design experiment. Ten teams of three people have been given an innovative design task (the design of an Antarctica-like museum) within the context of an industrial setting. Their reasoning processes have been analyzed using verbal protocol analysis. Two hypotheses, regarding the quantity of properties produced and the quantity of novel properties produced have been tested – both of which have been rejected, as predicted based on a particular model of design, namely, C-K theory. After reviewing brainstorming research in section "Brainstorming Research", we introduce a creative design process description based on C-K design theory in section "Design Theory and Models: Rich Description of Creativity". Section "Research Design and Methodology" discusses methodology and data collection. Section "Results" presents results, before concluding in section "Conclusion".

Brainstorming Research

Overview

Focus of brainstorming research has been to increase number of ideas generated by groups [25, 26]. Initial research wanted to show that group creativity is better then same number of individuals (called nominal groups) working separately. Experiments have shown that nominal groups are more productive. A second episode of research on brainstorming tried thus to identify causes for this phenomenon. A number of factors such as the production loss, blocking, social loafing, fear of evaluation have been identified [25]. The natural tendency followed was to rectify this situation by eliminating these factors. Numerous techniques such brainwriting and anonymous evaluation has been proposed.

Evaluation of a Creativity in Brainstorming Literature

Brainstorming research has accepted, often unquestioningly, a major hypothesis initially introduced by Osborn himself:

> *Quantity breed quality hypothesis* (*QBQ*): "*It is almost axiomatic that quantity breeds quality in ideation. Logic and mathematics are on the side of the truth that the more ideas we produce, the more likely we are to think up some that are good.*" (*Osborn 1963, p. 131*)

In the literature, this hypothesis has manifested itself as an evaluation process that is based essentially on the *number* of ideas. Literature often discusses criteria introduced early on [27, 28] such as fluency, originality, and flexibility. Fluency is a quantity of the number of non-redundant ideas generated during the process Originality represents the uncommonness of an idea, given a problem, sometimes measured as the relative rarity of an idea given the pool of ideas produced by several participants for the same task [28, 29]. Flexibility measures the ability to produce ideas that belong to uncommon categories of solutions given to the task at hand [27, 28]. Research indicates that these dimensions can be co-related but it is not always the case. Some studies indeed report that fluency and originality are co-related (i.e. that QBQ is valid [1] but this is not always the case [30] reports that previous work where both flexibility (number of categories used) and within-category fluency were present, no systematic correlation between the two was found [31] indicating these measures might be independent. Although the literature consistently argues that total production is correlated with high-quality output.

Feasibility vs. Originality

Despite the previously mentioned criteria, in experiments, the production of an ideation process is usually measured based on a dichotomy of originality and feasibility. Rietschel et al. [25] state that "*individuals or groups that generate the most ideas, also generate the highest number of good ideas* (*with good ideas usually defined as ideas that score high on both originality and feasibility*). As Rietschel and colleagues [32] puts it there is a general agreement among brainstorming researchers that quality in creativity tasks is some combination of *originality* and feasibility. Measuring idea quality by having external judges rating the originality and the feasibility of the generated ideas, it is claimed that quantity breed quality hypothesis has been verified [1, 32].

Critics of Idea Evaluation in Brainstorming

Contrary to mainstream brainstorming research, concerns have been raised in various research projects regarding the quantity breed quality hypothesis and the evaluation process used in ideation tasks. After a comprehensive review of brainstorming research, [26] concludes that available evidence is not conclusive, or even conflicting.

Quantity Is Not the Issue in Real-Life Innovation Processes

Williams and Sternberg [33] instructed teams of participants to produce a best idea ratter than as many ideas as possible. Contrary to usual work comparing nominal groups and interactive groups, they found out that teams were more successful in generating an overall superior idea than individuals. Their instruction clearly violated the quantity breed quality hypothesis, while approaching a more realistic setting. As we shall argue later on, the objective of finding a best idea enforces the evaluation of ideas while they are being generated, rather than a pure generative process, the activity becomes a *reasoned* process. Rowatt et al. [34] demonstrated with a series of experiments that people have indeed a preference for *quality over quantity*. They interpreted their finding as a reason for revising brainstorming instructions to downplay the importance of quantity and emphasize the importance of quality. Paulus [35] interpret this preference as fear for novelty and judging somewhat unfortunate that people tend to focus more on usefulness and validity.

Phased Separation Between Idea Generation and Evaluation

Rietschel et al. [25] acknowledge that idea generation is only a part of the creative process and not a goal in itself. However, their diagnostic for situating the dynamics of idea generation within the broader process share the same flaw many other brainstorming researchers: they make an implicit assumption that idea generation and evaluation are two separate processes: *A question largely unaddressed by brainstorming research thus is how exactly the production of ideas contributes to creative solutions or innovations after the idea generation stage* [25].

Paulus [35] argues that a lot of potential that is built up during the idea generation phase is wasted because people do not know how to evaluate ideas. Citing Rietzschel and colleagues [36], he states that idea generation is still an elusive issue for brainstorming research. *Typically it is suggested that generation and selection should be two separate phases, but evidence thus far is not clear on this issue. Alternatively, it might be best to mix short idea generation sessions with evaluation sessions. This will be a puzzle for future research to resolve*. As we shall see, this is far from being a puzzle but an elementary property of creative processes in design research.

Design Theory and Models: Rich Descriptions of Creativity

Contrary to brainstorming research having strong relationship with experimental psychology literature, design research combined several methodological approaches to understand complex and real-life creativity situations that are design processes. One of the methodologies that have been used is theoretical modeling [16–24]. In the present work, we are going to consider a particular theory of design, namely the C-K theory [20] – and contrast it with the brainstorming underlying process model.

An Overview of Value, Feasibility and Originality in C-K Theory

C-K theory describes design based on the interaction between two spaces. In the knowledge space, propositions about the known world exist. They are either true or false. In the concept space, there exist definitions for classes of objects. The theory claims that, in innovative design processes, those definitions are undecidable: it cannot be stated that corresponding objects can or cannot exist – until the end of design. In C-K theory creative propositions (and thus, originality) stems from a particular type of operation a designer applies in order to elaborate an undecidable propositions. This kind of operation, called expansive partitions or conceptual expansions, adds to a concept an unusual or unknown property in order to build new and unprecedented object definitions (e.g. an Antarctica-like mobile museum). A second type of operation, called restrictive partitions, adds to a concept a usual and known property (e.g. a history museum). Restrictive partitions does not necessarily create easy-to-realize object definitions, since creative design already starts with a conceptual expansion – and the known property added by the restriction stills need to be connected with the unusual properties within the concept (e.g. an Antarctica-like mobile history museum). Although not explicitly stated by the theory, in practice it is often assumed that concepts hold value. Value is constructed progressively, extended if necessary, using partitioning (expansion or restrictions). Thus, conceptual expansions create or add new values to a type of object (e.g. an

Antarctica-like mobile museum provides opportunity for children to be immersed into an Antarctica-like universe, not far from where they live).

There is no guarantee that a creative concept is going to be validated (i.e. acknowledged as feasible): concepts are validated if, during the process, knowledge warranting the existence of such an object is produced, activated or found. Expansive partitions, by definition, introduce originality into a design, but makes more difficult to validate a concept (i.e. to make it feasible).

Thus, in design, separation between the originality and the feasibility of a concept is not a problem; it is an opportunity to achieve both: a concept is unfeasible by nature (it contains creative expansions that makes it unfeasible). It is by the process of design that those concepts are made to exist. Thus, unfeasible yet original ideas cannot be discarded as invaluable or of poor quality. They allow the exploration of both value and feasibility.

Theory-Driven Predictions About Brainstorming Hypotheses

Based on the dynamics described by C-K theory, it is possible to produce and predict the outcome for a myriad of hypotheses about the dynamics of a design process and its impact on performance in terms of feasibility (F), originality (O) and value (V) of ideas. To demonstrate the approach, we shall introduce a second hypothesis. As mentioned earlier, the conceptual separation between idea generation and evaluation is a simplistic model that is bound to induce interpretation problems regarding the creative processes. Paulus [35] states: *People tend to focus more on usefulness and validity. There is clearly fear of novelty*. This seems to assume that originality should prime over feasibility and value of ideas. We might formulate this hypothesis as quantity of novel (unknown) properties breeds quality:

Novelty breed quality hypothesis (NBQ) Higher the number of novel properties, better the quality in ideation.

C-K theories and more generally, work on design research goes clearly against this hypothesis. For instance, Girorta et al. [37] argue that organizations prefer a single outstanding idea to several good ideas. In design processes, the objective of finding a best idea enforces the evaluation of ideas while they are being generated – but the evaluation steps do not necessarily eliminates ideas. As C-K theory points out, new *knowledge* allowing further elaboration and improvement of those ideas are generated as well. As can be expected from such process, designers tend to generate sufficient number of original ideas that embodies value but they also need to make sure the feasibility of those ideas by elaboration. Brainstorming research, seeing the creative process as the generation of a sequence of unconnected ideas, misses this crucial insight. Thus, based on C-K theory's description of design process, novelty breed quality hypothesis should prove to be false.

Research Design and Methodology

Overview

In order to address our research problem, we realized verbal protocol analyses of several design teams' creativity sessions. Our groups were composed of three designers with experiences in R&D activities. The creativity sessions were part of an innovative real-life project, supported by a French cross industry innovation partnership: In 2012, MINATEC IDEAs Laboratory® decided to organize a series of innovative workshops involving multiple external professional designers. The sessions took place in a laboratory setting. Data were recorded and analyzed following the principles of verbal protocol analysis [38, 39].

Our research question was to explore how design teams ideate when they need to elaborate breakthrough concepts. Contrary to research that mostly used case-study methodologies for understanding group creativity at the firm level, verbal protocol analysis permits tracking the team cognitive process in a more fine-grained level. More details about our protocol are being presented elsewhere.

Data Collection and Research Protocol

Participants and Formation of Design Teams

The 30 participants were all either engineering designers or industrial designers with an average of 12 years of professional experience in R&D and innovation. Participants came from various industrial sectors and held different positions at the moment of the experiments: 10 were innovation managers, 11 engineering designers, 4 industrial designers, 3 industrial buyers and 2 B-to-B marketeers. No participant had previous experience in the design of museum or in the creation of important public social events.

Presentation of the Design Brief

The ten design teams were assigned to elaborate a breakthrough museum concept that gives the visitor an immersive experience in an Antarctica-like world. The participants were asked to elaborate both the form and the functions of the museum, the architectural aspects and the possible museum activities. Additional requirements were the following:

- The museum aims to make people sensible to the imperative of protecting Antarctica.
- The museum is mobile – it is a touring museum that could be deployed everywhere in the world whatever the conditions.

- Practical and easy to install and transport.
- Eco-friendly as much as possible (ecological materials; energy harvesting solutions. . .)
- The museum size is approximately 3,600 m2.
- The topic of museum is simple and easily appropriable for the subjects.

Because museums are a commonplace, all participants have some experience with them. The design brief is sufficiently open-ended; it offers the opportunity to investigate how design teams think in very different ways about different domains. Also, the object to be designed, by its very definition, is outside the scope of any known instance of its category.

Organization of the Design Sessions

At the start of each session, the design task was given to the design team. They were informed that they had 1 h and 30 min to formulate one single concept of innovative museum. The design teams were asked to summarize their final proposal on an A3 sheet of paper with sketches, user scenarios, texts and motto. All experiments were launched in the same large room; table, white papers/papers, pencils were provided. Participants did not have any access to external documents (no computer, no internet connections, no books, no phone. . .).

At the end of each session, the participants were given 10 min to present their final product concept to the organizers. In order to cover all the aspects of the design proposals and to provide reliable and comparable qualitative data between the ten designs, a semi-structured guide was used to question them about their designs. All the experiments and interviews were video and audio recorded.

Research Protocol of the Ratings of the Final Designs

A panel of 14 professionals assessed the ten final designs – 6 of them were experts in museum (2 directors of museum, 2 curators, 1 public programmer and 1 exhibition designer) and 8 of them were specialized in the organization of public events (1 director and 7 project managers). Judges were asked to rate the ten final designs with three criteria according to a five level Likert scale:

- **Novelty** of the product concept compared to the existing museums – all kinds of museum could be considered (from traditional museums to innovative ones (planetarium, 3D-relief movies. . .));
- **Feasibility** of the product concept in terms of how it can be implemented – economical and technical feasibility were considered;
- **Value** for users; value of the product concept for potential visitors.

All judges were blind to the research. In order to increase the reliability of judges, the evaluation process was divided into three steps according to an adapted version of Delphi technique. In the first step, the rating criteria were presented and discussed by the judges. This aims to reduce the possible differences in the interpretation of the three criteria. Then, the final sketches of the ten final design concepts were presented and each judge was asked to rate independently. Finally, the results of ratings were discussed in an unstructured form by the judges who could modify (or not) their initial ratings. Inter-rater reliability between the judges was measured. We use the inter-judge agreement reliability formula, called the Proportional Reduction Loss (PRL) of Rust and Cooil's [40]. PRL is proved to be of superior performance of several other reliability approaches for qualitative or quantitative data since it does not allow reliability to appear inflated. The inter-rater reliability was calculated for each of the criteria during both the first and second round of evaluation: *PRL-originality* (*2nd* = 0.*74*; *1st* = 0.*71*); *PRL- feasibility* (*2nd* = 0.*74*; *1st* = 0.*71*); *PRL- value* (*2nd* = *0.70*; *1st* = 0.*71*). All statistics show PRL $\geq$ 0.7 and so the internal consistency is largely acceptable.

Data Coding and Analysis

Verbal protocol analysis has been used to study the conversations and the different interactions between the participants. Teams activities were recorded and transcribed. Information regarding the identification of the speaker and the specific actions and reactions of participants (drawing, handling objects, jokes, laughing, mime...) were integrated into the transcripts. In total, the ten transcripts covering 15 h of audiovisual data were used in the present analysis. The degree of verbalization varies between teams, number of words ranges from 12.500 to 23.000.

Coding of Design Properties: Classic Versus Novel Properties

Utterances from design teams have been used to capture various design properties they dealt with. All the design properties were then either coded as "known" or "unknown" by two experts. Known properties refer to classical properties of museums (i.e. restrictive partitions). Unknown properties are properties imagined by designers that do not correspond to any traditional properties of museums. During coding, a design property was considered to be known if it could be trivially observed in existing or past museums. Several documents about various types of museums (scientific, artistic, cultural, or historical) were considered – a specific attention was given to the website of the major museums dedicated to Antarctica, the museum of Artic and Antarctic in St Petersburg (Russia). In the case where an uttered design property was not usually encountered in museum, it was coded as an unknown property. For instance, the design properties "a museum that is a zeppelin" or "museum with ice footsteps sound effects" fall in this category.

Inter-rater Reliability Measures: Identification of Properties

Two of the authors completed the coding of all the data independently and without communicating. Design properties were identified by highlighting each time they appeared in the transcribed data; the two coders individually named each design property, with its timestamp and the name of the designers. For coding, the software Atlas Ti (version 6.2, www.atlasti.com) was used. Afterwards, the two coders analyze and compare their coding segments side-by-side in order to validate (or not) their identification. The Percentage Agreement (PA) between the two coders for identifying the design properties was calculated for each team. While this method does not exclude agreements occurred by chance, it is simple and appropriate for exploratory conditions. For each transcript, the PA was calculated. The naming of design properties was discussed by the two coders. All the differences in the interpretations were resolved between the two coders. The overall average PA is 0.77. Such level shows that the identification of design properties can thus be considered fully acceptable and satisfactory (PA $>$ 0.7). Inter-rater reliability between the two authors was measured for the full set of design properties. The PRL for each team reach a satisfactory level ($>$0.7), in overall, the PRL was 0.75.

Results

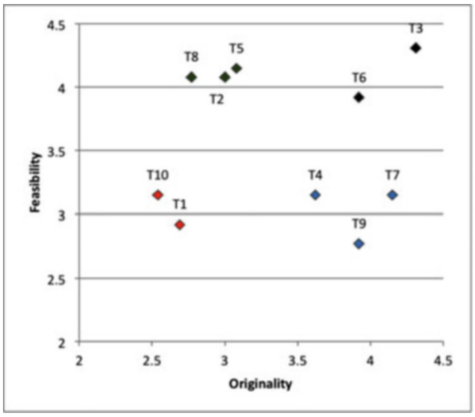
An Overview of Team Performances

Table 1 and Fig. 1 give some insights into the nature of team performances. Here, we can observe that there are four distinct categories of team performances. Teams #3 and #6 are the best teams overall. In particular, they have performed well both on criteria O and F. Teams #1 and #10 are the worst performing teams on both criteria compared to others. The remaining teams let appear two contrasted profiles.

Teams #2, #8 and #5 are teams that have performed reasonably well (worst than the bests, better than the worsts). Their performance profiles are similar; they are all

Table 1 Team performances on O, F and V

Team	Originality (O)	Feasibility (F)	Value (V)
Team #1	2.69	2.92	2.62
Team #2	3	4.08	3.31
Team #3	4.31	4.31	4.31
Team #4	3.62	3.15	2.77
Team #5	3.08	4.15	3.15
Team #6	3.92	3.92	3.31
Team #7	4.15	3.15	2.85
Team #8	2.77	4.08	3.46
Team #9	3.92	2.77	3.00
Team #10	2.54	3.15	2.69



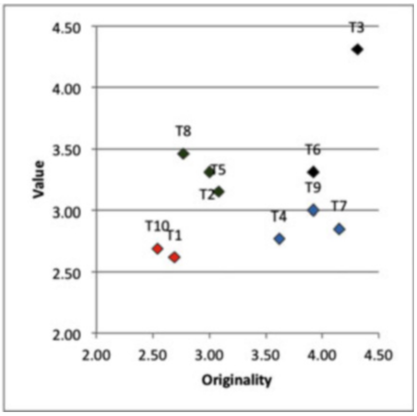


Fig. 1 Comparison of team performances, on the *left*, feasibility vs. originality criteria; on the *right*, value vs. originality. The first chart suggests there are four clusters according to the traditional criteria in brainstorming studies (originality-feasibility). The second chard suggests, although the general grouping is somewhat preserved, the consideration of the value criteria puts Team#3 further apart then the rest

Table 2 Number of properties, unknown properties, known properties and their rations per teams

Team	# of Ps	# of UPs	# of KPs	UPs/KPs ratio	UPs/Ps ratio
Team #1	64	31	33	0.94	0.48
Team #2	68	39	29	1.34	0.57
Team #3	60	39	21	1.86	0.65
Team #4	57	36	21	1.71	0.63
Team #5	95	75	20	3.75	0.79
Team #6	125	79	46	1.72	0.63
Team #7	88	56	32	1.75	0.64
Team #8	95	51	44	1.16	0.54
Team #9	85	59	26	2.27	0.69
Team #10	85	47	38	1.24	0.55

rather better on the feasibility criterion compared to originality. By contrast, teams #9, #7 and #4 (again worst than the bests, better than the worsts) did perform better on originality than on feasibility. When we consider the performances of these four categories of teams on the value criteria, we see that the bests and the worsts remain stable. Between the two remaining groups, the feasibility oriented group has performed slightly better with respect to value criteria than the originality oriented group. Different hypotheses that we are going to test give us a better understanding of these performances.

In addition, Table 2 provides information about the number of properties generated by the teams. We found no significant correlations between the quantity of properties generated and the three evaluation criteria. For example, team #6 generated the highest number of design properties (125 design properties, ranking second overall), but the team #3 (best team, ranked first in all criteria was among the teams that generated lowest number of properties.

Our data seem to suggest that generating a large number of unknown properties increases the novelty of the final concepts ($r = 0.077$, $p < 0.05$, $n = 822$) and that design teams naturally talk much more and much more often about unknown properties than about the known properties (respectively, $r = 0.107$; $r = 0.138$; $p < 0.01$; $n = 822$). We found that the number of unknown properties does not necessarily impact the feasibility and the value of the final concepts. Thus, a design team can generate a high amount of unknown properties with a low overall score (e.g. team #10, ranking 9) or inversely, a design team can generate a low amount of unknown properties with a high overall score (e.g. team #3, ranking 1). According to our data set, it may be the case that what is important for breakthrough concept elaboration is not the quantity of properties (or unknown properties) but rather the proportion between the number of unknown properties and the number of known properties.

Our dataset seem also to indicate that the timing of generation has different impacts on the three criteria. First, the unknown properties produced early in the process seem to have a higher impact on the novelty of the final concepts than the unknown properties produced late in the process (correlation between time of unknown properties emergence and novelty $r = -0.103$; $p < 0.05$, $n = 512$). Second, the properties that are generated late in the process positively impact the feasibility score of the final concept (correlation between time of properties emergence and feasibility $r = -0.08$; $p < 0.05$, $n = 822$). The design properties generated at the end of the sessions can reinforce the feasibility aspect of the final concept, but the properties can be unknown or known properties. Finally, the known properties generated late in the process increase the value score better than the known properties proposed earlier in the process (correlation between time of known properties emergence and feasibility $r = -0.13$; $p < 0.05$, $n = 310$).

(QBQ) Quantity Breed Quality Hypothesis: Bigger the #Ps, Better the Team Performance

There are several interesting consequences that can be drawn from Table 1 and Table 2. First, we see that the relationship between the number of (non-redundant) properties and team performance is questionable. The top three generating teams are team #6, #5 and #8. Team #6 generated the highest number of design properties (125 design properties), well above the average (>82.2). Its performance appears to be well on originality and value criteria (third in both). On feasibility criteria it's ranked fifth. Its average ranks are rather good (second on both O + F and O + F + V).

This might tempt us to think that QBQ is correct: the group that has generated the most ideas is a good group overall. Looking at the top second and third most generating teams (respectively, team #5 and #8), however, gives a clearer picture.

Both have generated 95 ideas (again, well above the average). Team #5 has an unsteady performance profile (sixth on O, second on F and fifth on V). This team has achieved particularly well on the feasibility criterion; however, its overall rank on O + F is just above average (fourth). Team #8 has also an irregular performance (eight on O, fourth on F and second on V). This team has achieved particularly bad on the originality, but surprisingly, it has one of the best ranks in value criterion (second). However, both of its overall ranks are average at best (sixth on O + F and fifth on O + F + V). Looking to the sporadic performances of these top-generating teams, QBQ cannot be confirmed clearly.

The least generative teams shed further light on QBQ. In fact, one of the two least generative teams, namely team #2, is the team with the highest performance on all criteria, thus, arguably the best team. *They have only generated 60 ideas*. This goes clearly against the QBQ. Let us remark that the opposite of the QBQ (the less the number of ideas, the better the quality) cannot be stated either. Team #1 and #4, two of the three least generative teams have arguably bad performances. Team #1, which generated 64 properties, has ranked ninth or tenth on all criteria. Team #4 has ranked fifth, seventh and eighth on O, F and V. In sum, according to our dataset, there is no relationship between the number of generated Ps and the quality of Ps. Thus, we reject QBQ.

(NBQ) Novelty Breed Quality Hypothesis: Bigger the #UPs, Better the Team Performance

Let us consider NBQ and the number of unknown properties (UPs) generated by the teams. The team that generated the most UPs is team #6, which is the same as the team that generated most Ps. It is indeed one of the best teams (second both on O + F and O + F + V).[1] The second best group on UPs ranking is team #5, which is also a good team in overall performance (fourth on O + F and third on O + F + V). Going to the opposite side, the two teams that generated the least UPs (team #4 and team #1) have poor performances overall (team #1 being the least good team in the experiment, and team #4 being seventh on O + F and eighth on O + F + V). Considering these, it is tempting to suspect a positive relationship between the number of UPs generated and the team performance. However, if we consider the third most and third least generative teams, the argument breaks down. In fact, once again team #1 who is the uncontested best on performance criteria is one of the least UPs generating team (ranked third). Likewise, one of the worst teams, team #9 (eighth on O + F and seventh on O + F + V), has ranked third according to the number of UPs generated, again going against the NBQ.

[1] It should be noted, however, that their score on feasibility is average (fifth) and thus, their being second in overall is mostly due to the compensatory nature of the averages (e.g. a bad score can be compensated by a good one, and vice versa).

Much as it is hard to confirm NBQ with the available data, some variations of NBQ deserve some attention. Let us consider:

- NBQ’: if a group’s generation of UPs is below some threshold with respect to the average, it will have a bad performance.
- NBQ”: if a group’s generation of UPs is above some threshold with respect to the average, it will have a good performance.

Although not conclusive from the current data, there is some evidence supporting these hypotheses. If we consider that the average UPs production is 51.20, we see that both most and least productive teams are beyond the edges of standard deviation (±16.25). Team #5 and #6 have both produced above 67.45 UPs and both have good performance. Team #4 and #1 have both produced below 40 UPs (slightly above μ-σ) have performed poorly. Considering the size of our sample and the performance of team #3 (which has also produced slightly above μ-σ but has best performance), this is still inconclusive. However, there is reason to check for NBQ’ and NBQ” in future iterations.

Another way to consider NBQ is from the performance charts (Fig. 2). Here, we see that the top-performing group (Team #3 and #6) has a significant difference in terms of number of UPs produced. Likewise, the difference between the worst performing teams is considerable (Teams # 1 and #10). This variability in terms of number of UPs is also valid for the middle performance groups. Both the feasibility and the originality groups span widely along the number of UPs axis. In the light of these observations, it cannot be stated that NBQ is a direct predictor of team performance.

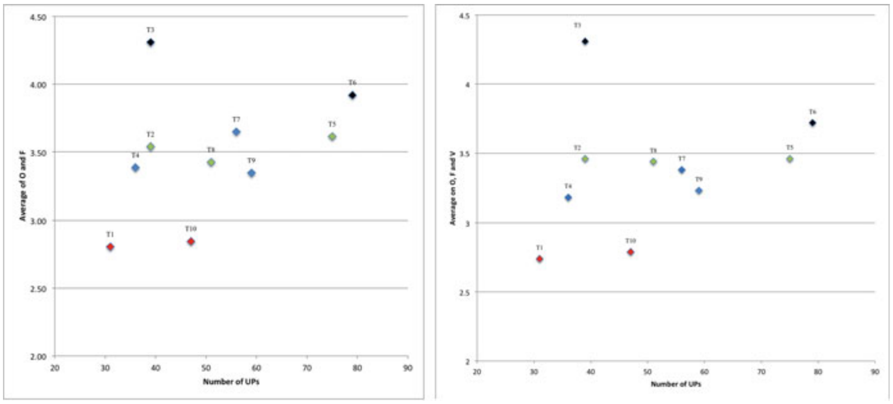


Fig. 2 Comparison of team performances based on #UPs. On the *left*, an average performance measure based on feasibility and originality criteria were used; on the *right*, team performance based on originality, feasibility and value criteria have been used. Among other things it can be noted that the number of UPs produced by a team is not a predictor of its performance (e.g. best and worst teams have about the same number of UPs produced; and best and second best teams have produced respectively very low and very high number of UPs)

Conclusions

The full set of results from our experiment is still being processed. The major result presented in the current work is the invalidity of quantity breed quality hypothesis of brainstorming in our experimental setup based on a design creativity task. We also demonstrated that the number of original (unknown) propositions generated by a team is not predictor of high performance. An inspiring observation is that, groups that have achieved high on feasibility (from among the middle performance groups) are also better on value than the groups that have achieved better on originality. A possible explanation is that the groups that tend to have a few original UPs and that managed to ascertain feasibility of those outperformed groups that have only searched for originality. In our data, it would seem that a more balanced UPs and KPs generation allows higher originality. All the teams that ranked close to average (Teams #3, #7, #6, #9 and #4) have high originality scores. By contrast, teams that have ranked well on feasibility criterion are either a low or high ratio of UPs/KPs. It would seem, that having high originality with a balanced unknown generation strategy is not sufficient to obtain a high value or feasibility score. In future work, we shall analyze whether successful teams identify and work on the valuable and original ideas to guarantee their feasibility. This will require going beyond quantity-based metrics by introducing and analyzing process based metrics and their relationship with output proposals.

References

1. Diehl M, Stroebe W (1987) Productivity loss in brainstorming groups: toward the solution of a riddle. J Pers Soc Psychol 53(3):497
2. Stroebe W, Diehl M, Abakoumkin G (1992) The illusion of group effectivity. Personal Soc Psychol Bull 18(5):643–650
3. Osborn AF (1953) Applied imagination, principles and procedures of creative thinking. Charles Scribner's Sons, New York
4. Paulus PB, Brown VR (2003) Enhancing ideational creativity in groups. In: Paulus PB, Nijstad BA (eds) Group creativity: innovation through collaboration. Oxford University Press, New York, pp 110–136
5. Paulus PB, Nijstad BA (2003) Group creativity: innovation through collaboration. Oxford University Press, New York
6. Mannix EA, Goncalo JA, Neale MA (2009) Creativity in groups, vol 12. Emerald Group Publishing, Bingley
7. Summers I, White DE (1976) Creativity techniques: toward improvement of the decision process. Acad Manag Rev 1(2):99–108
8. De Bono E (1969) The mechanism of mind. Wiley Online Library
9. Hurt F (1994) Better brainstorming. Train Dev 48(11):57–59
10. Goldschmidt G (1991) The dialectics of sketching. Creat Res J 4(2):123–143
11. Gero JS, Maher ML (1973) Modeling creativity and knowledge-based creative design. Lawrence Erlbaum, New Jersey
12. Logan B, Smithers T (1992) Creativity and design as exploration. In: Gero ML (eds) Modelling creation and knowledge-based design, J.S.a.M. Lawrence Erlbaum, New Jersey

13. Goel AK (1997) Design, analogy, and creativity. IEEE Expert 12(3):62–70
14. Dorst K, Cross N (2001) Creativity in the design process: co-evolution of problem–solution. Des Stud 22(5):425–437
15. Le Masson P, Weil B, Hatchuel A (2010) Strategic management of innovation and design. Cambridge University Press, Cambridge
16. Yoshikawa H (1981) General design theory and a CAD system. In: Sata T, Warman E (eds) Man-machine communication in CAD/CAM. North Holland Publishing, Amsterdam
17. Takeda H, Veerkamp P, Tomiyama T (1990) Modeling design processes. AI Mag 11(4):37–48
18. Salustri FA (1996) A formal theory for knowledge-based product model representation. in 2nd IFIP Knowledge Intensive CAD workshop.
19. Hendriks L, Kazakci A (2010) Imagining future knowledge, logic and interactive rationality. The Institute for Logic, Language and Computation, University of Amsterdam
20. Hatchuel A, Weil B (2009) C-K design theory: an advanced formulation. Res Eng Des 19 (4):181–192
21. Maimon O, Braha D (1996) A mathematical theory of design. Int J Gen Syst 27(4–5):275–318
22. Braha D, Reich Y (2003) Topological structures for modelling engineering design processes. Res Eng Des 14:185–199
23. Gero JS (1990) Design prototypes: a knowledge representation schema for design. AI Mag 11:26–36
24. Kazakci A (2013) On the imaginative constructivist nature of design: a theoretical approach. Res Eng Des 24(2):127–145
25. Rietzschel EF, De Dreu KW, Nijstad BA (2009) What are we talking about, when we talk about creativity? Group creativity as a multifaceted, multistage phenomenon. In: Mannix E, Neale M, Goncalo JA (eds) Creativity in groups. Emerald Group Publishing, Bingley, UK
26. Reinig BA, Briggs RO (2008) On the relationship between idea-quantity and idea-quality during ideation. Group Decis Negot 17(5):403–420
27. Torrance EP (1962) Guiding creative talent. Prentice Hall, Englewood Cliffs
28. Guilford J (1967) Creativity: yesterday, today and tomorrow. J Creat Behav 1(1):3–14
29. Amabile TM (1996) Creativity and innovation in organizations. Harvard Business School, Cambridge
30. Förster J, Friedman RS, Liberman N (2004) Temporal construal effects on abstract and concrete thinking: consequences for insight and creative cognition. J Pers Soc Psychol 87 (2):177
31. Nijstad BA, Stroebe W, Lodewijkx HF (2003) Production blocking and idea generation: does blocking interfere with cognitive processes? J Exp Soc Psychol 39(6):531–548
32. Rietzschel EF, Nijstad BA, Stroebe W (2007) Relative accessibility of domain knowledge and creativity: the effects of knowledge activation on the quantity and originality of generated ideas. J Exp Soc Psychol 43(6):933–946
33. Williams WM, Sternberg RJ (1988) Group intelligence: why some groups are better than others. Intelligence 12(4):351–377
34. Rowatt WC, Nesselroade K, Beggan JK et al (1997) Perceptions of brainstorming in groups: the quality over quantity hypothesis. J Creat Behav 31(2):131–150
35. Paulus P (2013) Intervievw on group creativity. Creat Innov Manag 22(1):96–99
36. Rietzschel EF, Nijstad BA, Stroebe W (2010) The selection of creative ideas after individual idea generation: choosing between creativity and impact. Br J Psychol 101(1):47–68
37. Girorta K, Terwiesch C, Ulrich KT (2008) Idea generation and the quality of the best idea. Working paper
38. Dunbar K, Blanchette I (2001) The in vivo/in vitro approach to cognition: the case of analogy. TRENDS Cogn Sci 5(8):334–339
39. Ericsson KA, Simon HA (1985) Protocol analysis. MIT Press, Cambridge
40. Rust RT, Cooil B (1994) Reliability measures for qualitative data: theory and implications. J Mark Res 31:1–14

Modeling Expectation for Evaluating Surprise in Design Creativity

Kazjon Grace, Mary Lou Maher, Douglas Fisher, and Katherine Brady

Abstract This paper describes methods for characterizing expectation as it applies to calculating surprise for evaluating the creativity of designs. Building on a model of creative designs as being novel, surprising and valuable we develop a typology of the kinds of expectations that, when violated, produce surprise. Contrasting computational models of two of kinds of expectation are presented in the domain of mobile devices and their respective advantages for creativity evaluation are described.

Surprise and Creativity Evaluation

Building on previous work [1, 2] in which we define a creative design as one that is novel, valuable and surprising, this paper extends methods for computing surprise. In this paper we use the following explanations for each of the three characteristics of a creative design. Novelty is the degree to which a design is different from similar designs that have come before it. Value is the degree to which a design is recognized as useful by the society in which it is embedded. Surprise is the degree to which confident expectations about a design are violated by observing it. Each of these notions individually may or may not indicate creativity – many designs are surprisingly bad, and novelty does not guarantee that the design is valuable – but we argue that the occurrence of all three together is a strong indicator of creativity.

Novelty captures the originality of a product compared to other products within the domain, while surprise captures the unexpectedness of a product to the observers of that domain. We distinguish novelty and surprise because it is possible for a design to be novel and expected. The ability to be original in ways that are not expected by experts is a hallmark of creative products, and thus both notions are necessary in computational creativity evaluation. This paper discusses what kinds

K. Grace (✉) • M.L. Maher
University of North Carolina at Charlotte, Charlotte, NC, USA
e-mail: K.Grace@uncc.edu

D. Fisher • K. Brady
Vanderbilt University, Nashville, TN, USA

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_11

of expectation are pertinent to surprise for design creativity and how they can be modeled.

Surprise has been described as the violation of a confident belief [3], an automatic reaction to a mismatch [4], an input-expectation discrepancy [5], and an emotional response to novelty [6]. Cognitive characterizations of the surprise process have focused on active (an explicit expectation formed prior to the surprising event) vs. passive (a post-hoc expectation arising from previous experience only after the unlikely perceptual datum is encountered) expectations [3, 4, 7], on low-level sensory vs. higher-level symbolic expectations [8], on whether the mental operation required to produce the expectation was inferential or retrieval-based [4], or on whether the expectation failed due to ignorance/unreliable evidence as opposed to was violated by genuinely surprising observation(s) [3]. These distinctions are significant in the investigation of surprise as pertains to design creativity, but of more relevance is the kind of information being expected [9].

Expectations are by nature temporal – they are based on using the past to predict what will occur. Some models represent this explicitly, conceptualizing surprise as time series prediction, with the sequence of past events being expected to continue, as in [10]. Sequence prediction is a necessary component of surprise for creativity evaluation, but we argue it is not sufficient. Expectations can be based on categorical inference from past experiences, such as "as it is a sports car, I expect this vehicle to handle well", or relational inferences, such as "this car has very low ground clearance, I expect it to handle well". Other forms of reasoning, such as analogy ("this car looks like a fighter jet, I expect it to be fast!") or meta-cognition ("I know a lot about cars, I assume this next car will fit with what I know"), could also be used in expectation construction. All of these strategies for forming expectations infer a future state from known past states, but they do not necessarily predict a sequence.

We formulate surprise probabilistically, as the degree of unexpectedness of the observation – the complement of the observer's a-priori expected likelihood of making that observation given previous experience. The less likely that an event would occur, the more surprising it is when it does. This is consistent with the formulations in [7, 8, 11]. We also specify a confidence value for each expectation, which is the observer's estimate of the reliability of the expected likelihood. This confidence is derived from previous experience and can be calculated statistically, logically, or otherwise, but only when confidence exceeds a contextually-determined threshold will surprise be registered. This ensures that we distinguish surprise from "mere expectation failure" [3].

Kinds of Expectation

What are the causes of surprise that are relevant to creativity evaluation and how can they be computationally modeled? A number of constructs within an observer's knowledge are relevant to expectation construction: a *design*, its *designer*, the design *domain*, the observer's own conceptual *knowledge* of the domain, and the *society* within which all four occur. While expectations about all those constructs can and are produced by observers, the causes of surprise relevant to the evaluation of the creativity of a design are those where the unexpected factor emerges from the design itself. When the designer, the domain or the observer's knowledge behave unexpectedly in ways that do not directly involve the design, the surprise elicited is not creativity-relevant. Expectations that are relevant to creativity fall into four categories:

1. *Categorical expectations* about a design given a classification either inferred by the observer or attributed by an external source.
2. *Trend expectations* about a design given domain trends (the change in designs over time).
3. *Relational expectations* about a design given an aspect of the design.
4. *Comprehensiveness expectations* that the observer's knowledge of the domain will suffice to describe new designs.

Type 1, or "categorical" expectations are formed as a result of previous experiences with designs that were classified similarly to the one being currently observed. The classification may be simple, complex, exclusive or non-exclusive, and it may be specified internally, deduced or inferred. Type 1 expectations include the expectation that a smartphone be less than 180 mm in length, an expectation that was violated in 2010 by the release of phone/tablet hybrids, known as "phablets". This kind of expectation was captured in the computational model S-EUNE [7, 11]. In the most general case, where the classification involved is membership in the design domain in question, this form of expectation is equivalent to novelty. Such a design is "unexpected for the domain given previous experience", effectively a measure of its difference from known designs, which we model as novelty. This accounts for the "surprise is a response to novelty" stance in [6]: our position is that *some* surprise is a response to novelty.

Type 2, or "trend" expectations are formed as a result of inferring trends from previous experiences and expecting that they will continue. The trend is expressed as a description of how one design attribute is expected to vary over time. Examples of such descriptions of a variable include a bounded range, a median, a distribution, or a set of inequalities. An example of surprise generated from this kind of expectation is the length of mobile phones in the mid-2000s: for the previous decade mobile phones had been getting shorter, but this trend was surprisingly reversed with the introduction of large-screen multi-touch displays. This kind of expectation has been the focus of our previous models of surprise [1, 2, 10].

Type 3, or "relational" expectations are formed as a result of correlations between attributes of designs. The description of this correlation is similar to that of Type 2 expectations, except the design attribute varies over the range of another attribute rather than over time. An example of this kind of surprise is the relationship between storage capacity and CPU speed in mobile devices: there is a strong positive correlation between these two variables, but Palm Pilots from the 1990s violated this expectation with higher storage capacities than other devices with similar CPU speeds. This surprising trait reflects the emergence of a different kind of device, the PDA. We have developed a model of this type of expectation for design creativity in detail in [12], which we outline here for comparison.

Type 4, or "comprehension" expectations are formed from an agent's beliefs about their own conceptual structuring of the domain and how it will change as a result of observing a new design. After an observer has become sufficiently confident in their understanding of a domain they come to expect that their understanding is comprehensive if not complete. New designs that result in a restructuring of the observer's conceptual structure for that domain of designs are surprising. This conceptual structure can be generated using hierarchical clustering, an approach we have previously described in [1] elaborate upon in this paper. An example of this kind of expectation is the emergence of the "smartphone" in the mid-2000s as a device that amalgamated both mobile phones and PDAs/digital organizers, resulting in a restructured space of possible phone designs.

Each of these four kinds of expectation can be formed actively from deliberate scrutinized expectations or passively from post-hoc background expectations. An active Type 1 expectation involves knowing about the classification of a design before observing the design itself, the approach used in [7]. An active Type 2 expectation involves hypothesizing about a design while knowing its release date. An active Type 3 expectation involves partial observation of a design, with the perceived attributes being used to expect something about the remainder. This is particularly relevant to temporally-experienced designs, such as music (as in [13]) or fiction, although partial experiences can also be a result of causes like sensory occlusion. An active Type 4 expectation involves the rare but theoretically possible situation that an agent is deliberately scrutinizing the comprehensiveness of its own conceptual knowledge.

With the exception of the fourth type, all expectations are of the form "X is expected given Y". These involve both the subject of the expectation (X) and the driver of the expectation (Y). The subject is expected by previous experience to occur in the context of the driver. In this paper we only consider when subject of Type 1–3 expectations is the design itself, such as in the expectation "this mobile phone is very heavy given its release date".

Expectations where the driver is the designer, such as "this house is very angular given that the architect is known for organic forms", say nothing about whether the design is surprising in the space of possible and known designs. The surprise in this case comes from a deviation of the designer's previous style. Other kinds of expectation, such as those involving society's response to a product, are also not considered here in evaluating that product's creativity. While expectations about the success of a product (and surprise from the violation of those expectations) are common, this dimension of a product's creativity is better captured by popularity-based measures of value rather than by surprise. Expectations of value judgments by others and surprise based on those judgments have been theorized as relevant to autonomy of creative preference (as they permit an agent to socially adapt its preferences as in [14]), but not to creativity evaluation directly.

Computational Models of Expectation

We develop two models of surprise, one based on predicting distributions of design attributes and capable of Type 2 and Type 3 creativity-relevant expectations, and one based on hierarchical clustering of designs and capable of Type 4 expectations. Both models adhere to our definition of surprise as an observation's unlikelihood given expectations derived from experience, but both are based on different kinds of expectation. The predictive model uses regression to predict the distribution of one attribute in terms of another, and the clustering model uses a hierarchical categorization of the design space to model the similarity between designs at multiple levels. Code for both models can be found at *github.com/kazjon/SurpriseEval*.

The predictive model concentrates on single attributes in the newly observed design while the clustering model takes a more holistic view of each design. Both models are sensitive to the order in which designs were introduced in time, as they are unaware of future designs when performing an evaluation and would produce different results if the temporal order of designs was changed. Both models are also unaware of whether the surprise registered in response to a design would be positive or negative – that is the domain of the value component of creativity evaluation.

Each observed design is described by a set of real valued attributes. These include descriptive attributes such as length and width, performance attributes such as RAM Capacity and CPU clock speed, and production attributes such as the release date of the device. In these models we are concerned only with passive expectation, although both models are extensible to an active expectation context.

Predictive Model

The predictive model builds expectations about the relationships between attributes. These expectations can be expressed in English by sentences such as "devices with a later release date will be thinner" or "the RAM capacity of a device is inversely proportional to its size". When a new design violates the expectations about how attribute A (the "predicted" attribute) varies as a function of attribute B (the "observed" attribute) we express this with the statement "It is surprising that the value of attribute A for this device is x given that its value for attribute B is y". We model the relations between each pair of variables in this way, taking the highest pairwise unexpectedness variable for each design as its surprise. This is based on the assumption that all variables are equally important, but could be customized for domains where that was not the case.

Expectations are represented in the predictive model as an expected cumulative distribution function. This represents the range of expected values of the predicted attribute given the value of the observed attribute. The (un)likelihood of each attribute of a new design is measured using the distributions returned by these functions. In this model the observed attribute can be release date in order to construct a Type 2 expectation, or it can be any attribute of the design to construct Type 3 expectations. A surprising design by this model is one that violates expectations about how its attributes will relate to each other.

The stochastic functions used to make predictions in this model are built using non-linear regression algorithms trained on each additional design in sequence. To address the distinction raised by [3] between expectation failure arising from incomplete data and true surprise we implement a confidence factor. The confidence factor is based on the past accuracy of the predictive model for designs near the one being observed. This serves as an indicator of how strongly the observer should expect the predicted distribution to be accurate. When constructing Type 3 expectations the density of observations is rarely uniform, making accurate predictions across the range of the observed variable difficult. We calculate a local confidence factor based on the density of the data near an observation. The level of confidence is used to scale our surprise values: in areas where there are no previous designs with which to compare the surprise value is very small.

Since the expected value of a design variable is not typically deterministic, the standard assumption in machine learning of an underlying deterministic function does not apply. We instead predict contours of the expected distribution (e.g.,: the median or the 75th percentile) using regression models and collate them into a histogram. A cubic function is then fit to the predicted contours to interpolate an expected cumulative distribution. The underlying assumption of this model is that the distribution of each attribute will vary in mean, standard deviation, skewness and kurtosis, but all these changes will be continuous and the distribution will always be unimodal. See [12] for details of the expectation construction process.

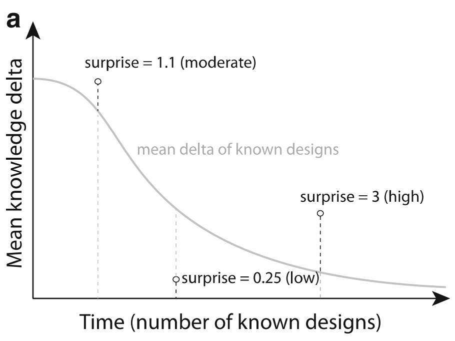
Surprise is measured as the probability of observing a value of the predicted attribute that was at least as unlikely as the one that was observed – that is, an expectation that is closer to the predicted median. For example if the attribute "width" of a mobile device is w and the predicted cumulative distribution function for width given the value of the observed attribute of the design is f then the measure of surprise for that device's width is $abs(1-2f(w))$. This value is then multiplied by the confidence factor to calculate the measure for surprise.

Clustering Model

Expectations in the clustering model are based on the observer's constructed hierarchy of designs within the domain. The model builds categories of designs and is then surprised if these categories must be altered to accommodate new designs. A description of being surprised in this way would sound something like "I thought all mobile devices could be categorized as either small devices without much memory, or large devices with a lot of storage, until I saw device x which is large but has little memory". This is a more holistic examination of the design's attributes than the predictive model (in that it considers relationships between all attributes concurrently, rather than considering pairs of attributes separately), but the results are complicated to interpret.

This approach to modeling surprise is based on building a knowledge structure from the data about previous designs and measuring the degree to which that structure is modified by a new design. We express the observer's knowledge structure about the domain using clusters of designs. A cluster is a group of designs that represent a region in the domain's conceptual space [15, 16]. The implicit assumption of this model is that its knowledge structure is comprehensive and stable, and that any significant perturbation of that structure as a result of observing a new design reflects an unexpected (and thus surprising) event. If the rate of observing new designs exceeds the rate at which the design domain evolves over time then this assumption becomes more accurate over time. To determine a confidence-adjusted measure and again distinguish between expectation failure (in this case the premature assumption of comprehensiveness) and true surprise, we normalize by the average surprisingness of recent observations, weighted by time. Figure 1a shows how the surprise value of three designs would be calculated relative to this running weighted average: the surprise for a design is the ratio of the change in the system's knowledge it caused (the "knowledge delta") to the average of recent changes. Figure 1b shows the actual mean delta over time.

The knowledge structure used in this model is a hierarchy of clusters developed through unsupervised learning. This structure is built using a divisive hierarchical clustering algorithm known as COBWEB [17]. Designs are added to this hierarchy one at a time in the order of their release, and their effect on the structure of the hierarchy is measured. New designs are introduced at the root of the hierarchy and then allowed to "percolate" downwards through clusters of increasing specificity until they are satisfactorily classified. Goodness of fit in each cluster is assessed using a utility function. Once this process completes the design is a member of all



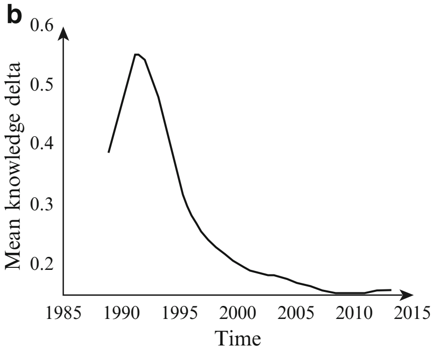


Fig. 1 (**a**) An illustration of the normalization process for producing confidence-adjusted cluster-based surprise. Surprise values are expressed as a multiple of the mean knowledge delta at the time of release. Three examples are shown along with their normalized surprise. (**b**) The average cluster-based surprise of each design as a function of time. This was calculated by a Gaussian-weighted average of the designs around each observed release date

clusters that are its parents in the hierarchy. This "percolation" happens by adding the device to a parent cluster (originally the root of the hierarchy) and then finding the two sub-clusters of that parent which would have the highest utility if the new device were added to them. The algorithm also considers the utility of merging these two sub-clusters, splitting the sub-cluster that had the greatest utility when the device was added to it, making the device its own separate sub-cluster and simply adding the device to the sub-cluster which had the greatest utility. Whichever of these options gives the most utility is carried out and the process is carried repeated recursively until the device reaches a cluster with less than two elements.

This model uses a utility function for real valued attributes from [18] (shown in Eq. 1). This function clusters devices without much variance in their attributes such that each cluster is differentiated from the norm of their parent cluster and that the number of sub-clusters of parent cluster is as small as possible. A more thorough explanation of this function has been omitted for brevity, but can be found in [17]. This utility function is more useful for the clustering of similar designs than other approaches which either divide all attributes near their average (using mean only) or divide designs based only on one attribute (using standard deviation only).

$$Utility(C) = \frac{\sum_{c\in\ C.children} P(c|C) \sum_{a\in\ Attributes} |\text{mean}(a|C) - \text{mean}(a|c)| \times (\text{std}(a|C) - \text{std}(a|c))}{|C.children|} \tag{1}$$

This method of clustering is less stable than the agglomerative algorithms usually used to make cluster hierarchies and is very sensitive to the order in which designs are observed. Unlike in most machine learning contexts this is a desirable attribute for our purposes since surprise evaluation is contingent on the order in which observations are made. There are methods of stabilizing cluster hierarchies created using COBWEB which we hope to investigate in future work, but as shown in Fig. 1 the hierarchy will stabilize over time in the current model.

In this model, surprise is defined as the degree to which a knowledge structure is changed upon observation of a new design. This is evaluated by a delta function, defined as the difference between the knowledge structure pre- and post- observation of the new design. The general principle of this delta function is to combine the number of changes made in the hierarchy with the inverse of the depth at which the newly observed design is clustered, as both are indicators of how well the observation sat with existing clusters. We use a delta function that divides the number of changes caused in the cluster hierarchy by the depth of the leaf the device ends up in, but determining an optimal delta function for cluster-based surprise is an area of ongoing research. We calculate the number of changes in the cluster hierarchy as the number of merges, plus the number of splits, plus one for the leaf or "singleton cluster" each device makes when it has settled into a category. While this value usually falls between zero and one, it is not guaranteed to do so. These raw delta values are then compared to the average for recent designs (as explained in Fig. 1), giving a measure of relative unexpectedness that is used as the surprise rating of a design.

Results

We have implemented both models in the domain of mobile devices using gathered from PDAdb.net, with 3,355 devices in total. These devices include mobile phones, PDAs and tablets released between 1989 and 2013. The mobile device domain was chosen as it is ubiquitous in modern life and has undergone several significant transformations in recent memory (smart phones, touch screens, tablets, etc.). Each device is represented by a set of 12 real valued attributes: CPU speed, display diagonal, display length, display width, display pixel density, height, length, width, mass, volume, RAM capacity and storage. The date of release for each device was also recorded.

Our models for surprise are independent of the features that are used to represent each design. In this paper we represent these designs using physical attributes because they were readily available and they demonstrate that our method can identify when significant changes have occurred in a domain. For example, surprise is recognized after observing the first touch-screen phones in the mid-2000s, despite the representation containing no information about touch capability.

Predictive Model Results

We predicted relationships between all pairs of attributes, with each attribute considered as both observed and predicted. To model both Type 2 and Type 3 expectations we consider release date as an additional observed attribute, for 144 models in total (132 attribute pairs and 12 release date comparisons). The surprisingness of each new observation was measured on all of these models and the highest observed surprise was used to score that observation. Some models were found to have strong predictive correlations (e.g., RAM capacity and pixel density) while others had only weak, partial or no predictive utility (e.g., pixel density and mass). The confidence measure ensured that false positives were not reported in cases of weak expectations.

Expectations for each design are constructed for each models using a training set of all previously released designs. Our model returns a distribution for each predicted attribute given the value of an observed attribute. The predicted likelihood of a less extreme value than the true one (the “unexpectedness”) is then multiplied by the predictive success of nearby previous designs (the “confidence”) to give surprise.

To illustrate a surprise calculation using this data we look at the LG KC1 as an example. This phone was released in August 2007 and runs the Windows Mobile operating system. This device was found to have a highly surprising CPU speed for its release date (a Type 2 expectation). For illustrative purposes we depict the full set of 3,355 phones, although calculations for the KC1 are based on the subset of designs released before it was released in 2007. Figure 2 shows the first step of the

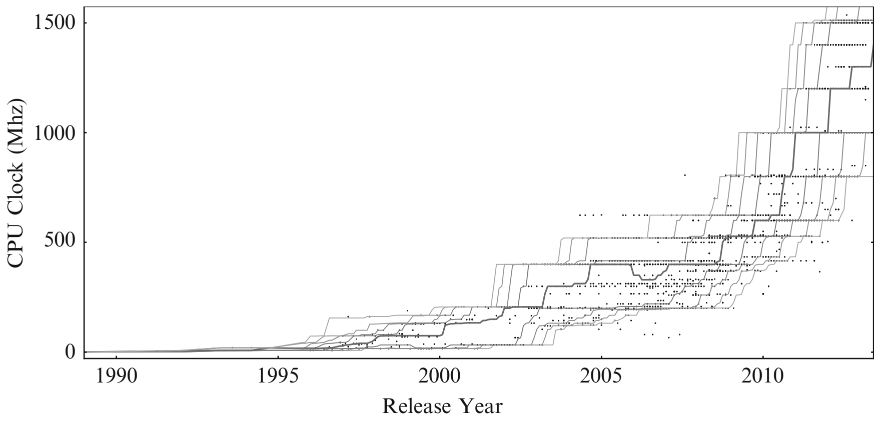


Fig. 2 The training data for the predictive model of CPU speed as a function of release date. The thick line in the center is the median of the observed distribution

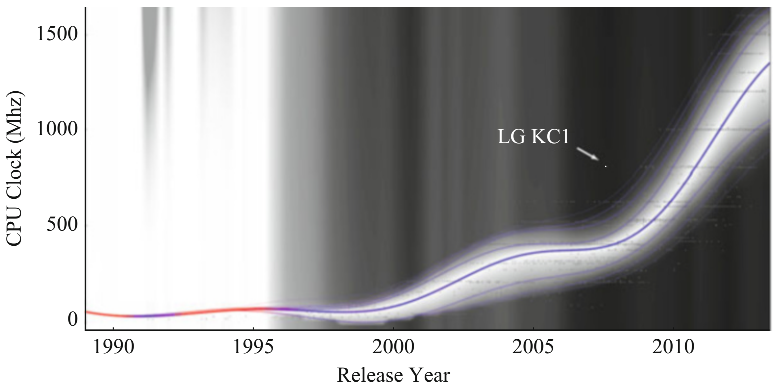


Fig. 3 The hypothetical surprise of all possible new designs as a gradient, overlaid with contours indicating the predicted distribution of CPU speeds over time. The LG KC-1 is highlighted, indicating its highly surprising CPU speed (806 Mhz)

predictive model process in which contours of the observed distribution of designs are constructed. The general upward and accelerating trend of increasing mobile device CPU speeds can be seen, with two notable periods: 2004–2007, where previous gradual increases seemed to partially stall with the introduction of many low-cost low-speed phones, and 2007 onwards, where that trend reversed and CPU speeds began increasing in step with computing power in general (Moore's Law), which they had previously not done. This reversal coincides with the release of multi-touch smart phones and the widespread adoption of general-purpose mobile computation.

Figure 3 shows the predictive model resulting from the CPU speed/release date data in Fig. 2. The contours represent the expected distribution of designs at that

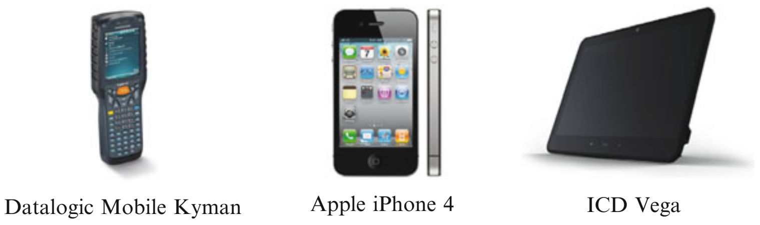


Fig. 4 Three devices rated as highly surprising by the predictive model

time, and the gradient represents the hypothetical surprisingness of a new design if it were observed at that point, with black being more surprising. Vertical bands of lighter tone reflect areas of low model confidence, while the horizontal band of low surprise reflects the channel where devices were expected to be. The LG KC1 is highlighted and it can be seen that it has a very surprising CPU speed for its release date.

The LG KC1's CPU speed with respect to release year was its most surprising attribute-attribute (Type 3) or attribute-time (Type 2) relation, resulting in an overall surprisingness of ~0.9. The KC-1's fast processor was mentioned in reviews of the product upon its release. While critical reviews were positive the device was only moderately popular within its home market of Korea. Its processor would have been considered mainstream two years later, making the KC-1 a design ahead of its time. We hypothesize that the pre-multitouch numeric keypad interface of the KC-1 did not allow users to make full use of its processing power, limiting the utility of its surprising feature. Three other highly surprising devices are shown in Fig. 4, with more complete results described in [12].

The Datalogic Kyman is immediately noticeable as unusual for a mobile device: it is actually a retail barcode scanner that was included in our dataset because it runs a mobile OS. It was surprisingly long, deep and heavy for its screen size, demonstrating that our surprise evaluator can identify surprising differences in form and function among designs (such as that of a barcode scanner among a database of phones and tablets) using relationships between simple physical attributes. The Apple iPhone 4, a device touted by the popular press for being innovative, was rated as surprising due to its high pixel density and storage capacity for its release date. Apple's marketing for this device revolved around its high-DPI screen, the "retina display", demonstrating that our surprise evaluator can identify popular and relevant trends. The ICD Vega had a surprisingly large display size (15″) for its release year, but also had a surprisingly low pixel density for its year. Both occurred because the Vega was a "tablet for the home" built around a laptop-like display.

Clustering Model Results

Mobile devices were added to the cluster hierarchy and evaluated for cluster-based surprise in the order of their release. Release date was not used as an attribute in this

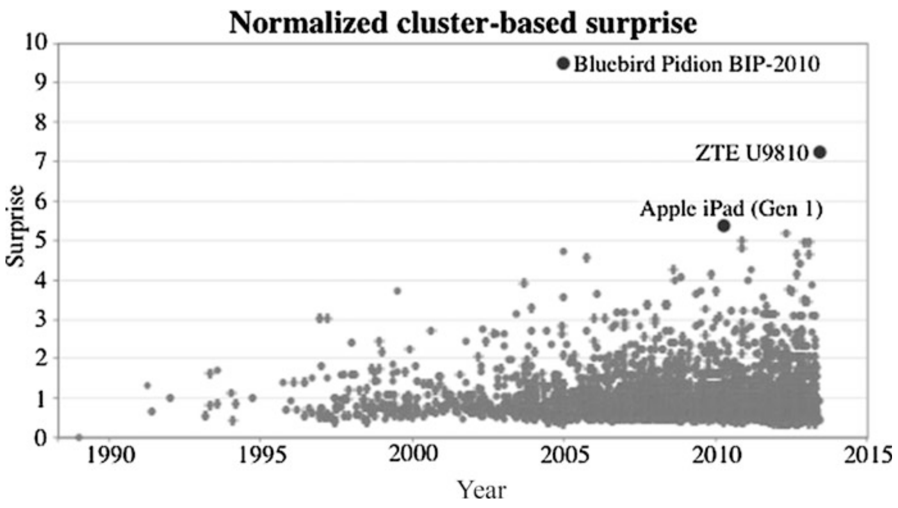


Fig. 5 The cluster-based surprise for all 3,355 mobile devices, with the three most surprising devices shown in *black*

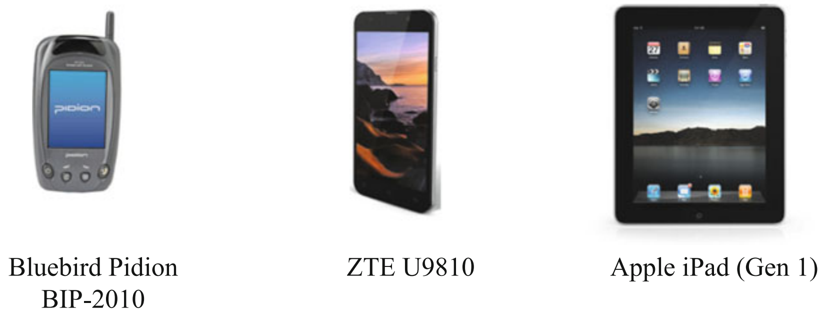


Fig. 6 Three devices rated as highly surprising by the clustering model

experiment explicitly (although designs were observed in order), giving a total of 12 attributes. The confidence-adjusted surprise values for each of the phones in the database can be seen in Fig. 5, where surprise is expressed as unexpectedness multiplied by the average unexpectedness of recent designs (see Fig. 1).

The three most surprising designs according to this model are visible in Fig. 6. We describe how and why these devices were evaluated as surprising to illustrate the cluster-based surprise calculation process.

The most surprising mobile device design according to our model of Type 4 expectation was the Bluebird Pidion BIP-2010, a 2005 phone that caused a significant high-level restructuring of the conceptual model. The clustering model's reaction to observing the BIP-2010 can be seen in Fig. 7, showing the relevant section of the hierarchy of clusters. The Pidion followed the green trajectory,

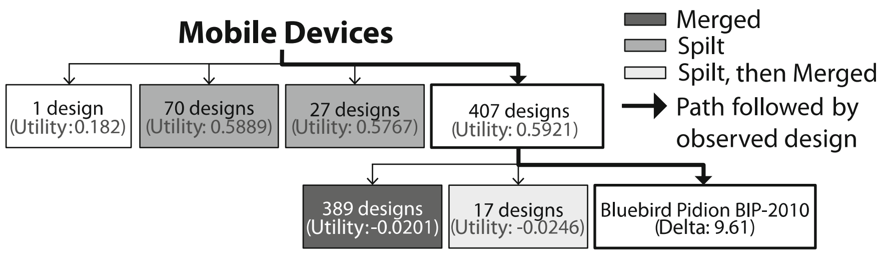

Mobile Devices
Merged
Spilt
Spilt, then Merged
Path followed by observed design
1 design (Utility:0.182)
70 designs (Utility: 0.5889)
27 designs (Utility: 0.5767)
407 designs (Utility: 0.5921)
389 designs (Utility:-0.0201)
17 designs (Utility: -0.0246)
Bluebird Pidion BIP-2010 (Delta: 9.61)

Fig. 7 A visual representation of the clustering model's restructuring after observing the Bluebird Pidion BIP-2010. The new design followed the bolded path through the hierarchy and affected the shaded clusters

became a singleton cluster after only two levels, and caused merges and splits to four other categories while doing so. The number of designs within each cluster is listed, along with the utility of placing the Pidion in that category. Note that the utilities are less different at lower levels of the hierarchy as the function is a global one and categories are made up of more similar devices lower in the hierarchy.

When it was being classified at the first level of the hierarchy the BIP-2010 was found to have physical dimensions that fit with larger, tablet-like devices but the screen size and other attributes of a smaller phone-like device. This discrepancy caused 17 devices previously thought of as "smallish large" (a category of relatively smaller devices within the category of large devices) to be reorganized into a new category of "largish small" devices (a category of relatively larger devices within the category of small devices). These 17 devices (and the Pidion) also had a CPU speed and display density more inline with the smaller of the two high-level categories, and the Pidion tipped the balance sufficiently for them to all be reclassified. This effectively redrew the boundary between tablet and phone, although the cluster hierarchy is not aware of such labels. The Pidion range are ruggedized mobile phones, which explains their larger physical dimensions. The BIP-2010 did not fit within established categories, forcing a restructure of the observer's knowledge and signaling presumptive surprise.

The predictive model also evaluates the Bluebird as surprising, although there are almost 100 devices rated higher than it. According to the predictive model the length of the Bluebird is very surprising (over 0.8) given its width, CPU speed or display size. This suggests that the two models agree in kind, although not degree, about how the Pidion was surprising.

The second-most surprising design using the clustering method was the ZTE U9810, a recently-released (mid 2013) smartphone from Zhongxing Telecommunications Equipment, which although largely unknown outside Asia is the world's fourth largest mobile phone vender. The U9810 is a high-end Windows 8 phone with a large display, fast CPU and lots of RAM and storage. As can be seen in Fig. 8, at the higher levels of the conceptual hierarchy it fit well, first into the category of small, phone-like devices and into the subcategory of thin, modern phones with above-average storage. Within these clusters, however, the ZTE began

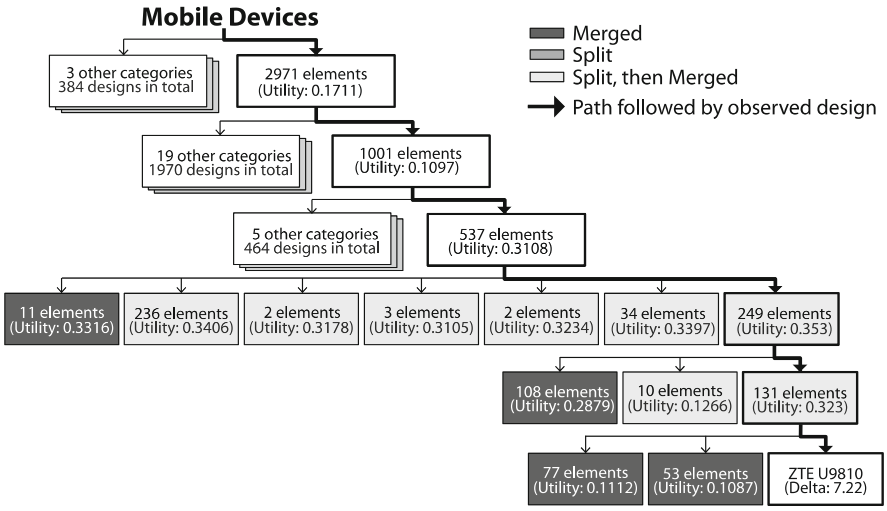

Mobile Devices
Merged
Split
Split, then Merged
Path followed by observed design
3 other categories 384 designs in total
2971 elements (Utility: 0.1711)
19 other categories 1970 designs in total
1001 elements (Utility: 0.1097)
5 other categories 464 designs in total
537 elements (Utility: 0.3108)
11 elements (Utility: 0.3316)
236 elements (Utility: 0.3406)
2 elements (Utility: 0.3178)
3 elements (Utility: 0.3105)
2 elements (Utility: 0.3234)
34 elements (Utility: 0.3397)
249 elements (Utility: 0.353)
108 elements (Utility: 0.2879)
10 elements (Utility: 0.1266)
131 elements (Utility: 0.323)
77 elements (Utility: 0.1112)
53 elements (Utility: 0.1087)
ZTE U9810 (Delta: 7.22)

Fig. 8 A visual representation of the clustering model's restructuring after observing the ZTE U9810. The new design followed the bolded path through the hierarchy and affected the shaded clusters. The stacked boxes are truncated depictions of multiple unaffected categories

to significantly affect the conceptual structure. All seven of the subcategories at this level were affected, with six suffering multiple splits and merges (the blue nodes on the fourth level of Fig. 8) to produce a structure that better represented the distinctions between smartphones after the U9810 was released.

After this reorganization the conceptual structure reflected that some small phones have large, high-resolution displays and higher CPU speed, attributes which previously were found only among larger devices. This shows that observing the ZTE broke down the system's belief that small devices were not as fast as larger ones, a Type 4 expectation that when violated signalled surprise. This also shows the benefit of a hierarchical structure: at more abstract levels of categorization there was nothing out of the ordinary about the ZTE-U9810, but it completely reorganized the observer's understanding of its specific region within the domain.

The predictive model evaluated the ZTE U9810 as reasonably surprising, among the top 500 of the 3,300+ devices. The device had a surprisingly high RAM capacity for its release year, and a surprisingly high mass for its volume, depth and display diagonal. This fits with the ZTE's reception amongst the tech press as being a high-powered high-end smartphone heavily marketed for its technical specs. ZTE's plan to equip a smartphone with a larger screen, more powerful processor and higher resolution camera comes at the cost of surprisingly high weight.

The third most surprising device was the original Apple iPad, a device released in 2010 which – based on an admittedly informal investigation of technology blogs from that period – appears to have elicited significant comprehensiveness-of-understanding expectation violation upon its release. As can be seen in Fig. 9, the iPad was placed (with relatively low utility) into the category of smaller devices

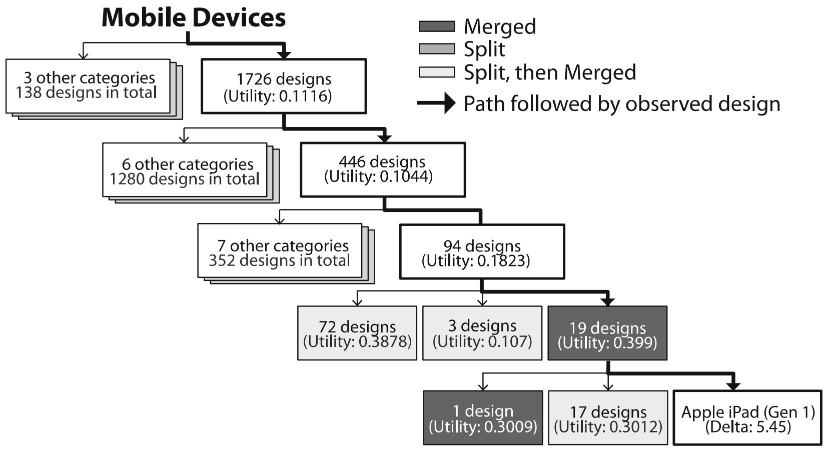

Mobile Devices
Merged
Split
Split, then Merged
Path followed by observed design
3 other categories 138 designs in total
1726 designs (Utility: 0.1116)
6 other categories 1280 designs in total
446 designs (Utility: 0.1044)
7 other categories 352 designs in total
94 designs (Utility: 0.1823)
72 designs (Utility: 0.3878)
3 designs (Utility: 0.107)
19 designs (Utility: 0.399)
1 design (Utility: 0.3009)
17 designs (Utility: 0.3012)
Apple iPad (Gen 1) (Delta: 5.45)

Fig. 9 A visual representation of the clustering model's restructuring after observing the first generation iPad. The new design followed the bolded path through the hierarchy and affected the shaded clusters. The stacked boxes are truncated depictions of multiple unaffected categories

that we recognize as being smartphones because of its fast CPU, RAM, storage and thinness, and because this category was already quite broad and varied due to drift in phone specs over time. It then disrupts that category because of its length, width and volume, splitting and merging its 94 elements to make three new clusters. The iPad itself then settles on the next level. This reorganization occurs because the iPad had more in common with previously observed smaller devices than with large ones. Making this observation violated the model's expectations that mid-sized devices have poor display resolution and low computing power, and that mid-sized devices tend to be thicker.

The predictive model evaluated the original iPad as more surprising than over 80 % of devices. The iPad was thin for its display width, its CPU speed and its release year, and had a very low display resolution for its release year, as the model was more familiar with smaller, ultra-high-DPI displays. While the clustering model found that the iPad had a high DPI display for its category, the predictive model – comparing to the whole domain – found that the iPad has a surprisingly low DPI display, but a surprisingly high display resolution.

Note that these three devices tend not to have one clear category at each level of the hierarchy into which they can be easily placed. This results in a "best" classification that is not significantly better than the next alternative. This occurs for highly surprising devices as by definition they do not fit well within the established cluster hierarchy, and for the typical device there is significantly more difference between utility values.

These results suggest that the clustering model correlates both with popular opinion in this design domain and with the predictive model. The two differ in the kinds of expectation they produce, and thus offer related but complementary perspectives on surprise for creativity evaluation.

Discussion

We have developed a framework for measuring surprise for creativity evaluation. We organize the expectations that cause surprise when violated into four types: categorical, trend, relational and comprehensiveness. We have developed a predictive model of trend and relational expectations and a cluster-based model of comprehensiveness expectations. Proof of concept implementations of our models in the domain of mobile devices have been presented. The models, and the framework that inform them, can serve as the basis for future computational creativity evaluation systems that adopt the novelty/value/surprise approach. While we have demonstrated the application of our models of expectation on descriptive attributes of mobile devices, the same models and algorithms can be used for other designs using a different representation for existing and expected designs.

The predictive model has the advantage over previous predictive approaches to surprise that it is based on expected distributions over values rather than expected values. This quality makes it possible to compare surprise between attributes and to model an expected range rather than a single value. The model constructs both Type 2 expectations about trends over time and Type 3 expectations about correlations between attributes.

The clustering model has the advantage that it measures the degree of reorganization required by an observer to adapt their knowledge to the presence of the new design. This is motivated by Boden's notion of creativity as transformation of conceptual spaces [15]. Boden divides creativity into "H-creative" events that are creative for an entire society, and "P-creative" events that are creative for an individual. We introduce a parallel distinction between "H-transformative" creativity, that which transforms a society's shared understanding of a design domain's conceptual space, and "P-transformative" creativity, that which transforms an individual's understanding of a design domain's conceptual space. The cluster-based model of surprise measures surprise indicating "P-transformative" creativity – designs that require an observer to restructure their own conceptual knowledge. While it is complex for an individual agent to assess the H-transformativity of a design, methods like our clustering model indicate the possibility of computational assessment of P-transformativity.

In both the predictive and clustering model, Apple products produced higher surprise levels than their peers. Of the six largest phone manufacturers (Samsung, Nokia, Apple, ZTE, LG and Huawei), Apple products were the most surprising on average and Samsung products were the least, despite the latter being the largest of the manufacturers.

The LG KC1, rated as highly surprising in the predictive model, does not have many surprising attribute relationships outside the CPU/release date one mentioned. Thus it is not surprising that the clustering model, in which release date is not used as an attribute, did not find this device surprising. The LG KC1 didn't cause any merges or splits in the cluster hierarchy and thus got a delta value around average for its release date.

A previously encountered limitation of the predictive model is that it is not well suited to evaluating the surprise of designs that are extremely novel [12]. The exemplary case is the Apple Newton range, tablet computers predating the iPad by over a decade. These devices were of low surprise because model confidence is extremely low in the very sparse regions of the attribute space that characterize highly novel designs. While the predictive model's "raw" surprise values unadjusted by confidence were high for the Newtons, the clustering-based model performed much better. The original 1993 Newton MessagePad and the 1997 Newton eMate 300 both ranked very highly in the cluster-based model as their unusual form factors distinguished them from the majority of contemporary devices.

The selection of an optimal delta function for the clustering-based surprise model remains an area of active research. The delta function measures the difference between the conceptual structure before and after a new design is observed. Our current approach operationalizes two concepts: goodness-of-fit and change in structure. Goodness-of-fit is the degree to which a design fits within the existing hierarchy (designs which do not fit will settle into a singleton cluster close to the root of the hierarchy, i.e., they will be only shallowly clustered). Change in structure is the effect of introducing the new design, as measured by how many clusters merged, split or moved. Initial experimentation with these two measures individually indicates that goodness-of-fit may be better thought of as a measure of novelty than surprise. Future research into comprehensiveness expectation measures will investigate both new delta functions and the relationship between cluster-based measures of surprise and hierarchical novelty.

The two models identify different kinds of creativity-relevant surprise. Both identify designs which do not follow established expectations about how attributes relate to each other and to time, but they do so in divergent ways. The predictive model tends to identify designs that violate a strong trend in a small number of attributes as measured against their other, less surprising attributes. By comparison, the clustering model tends to identify designs that provide the "missing link" to catalyze a reorganization of categories that were no longer the best representation of an evolving domain. The similarities between the results of these two approaches to modeling expectation indicate the promise of computational models of surprise for identifying creative designs, while the differences between them highlight the need for further research in expectation and surprise.

References

1. Maher ML, Fisher DH (2012) Using AI to evaluate creative designs. 2nd Int. Conf. Des. Creat. ICDC. Glasgow, pp 17–19
2. Maher ML, Brady K, Fisher DH (2013) Computational models of surprise in evaluating creative design. In: Proc. Fourth Int. Conf. Comput. Creat. Sydney, pp 147

3. Ortony A, Partridge D (1987) Surprisingness and expectation failure: what's the difference? In: Proc. 10th Int. Jt. Conf. Artif. Intell., Los Angeles, Vol. 1. pp 106–108
4. Lorini E, Castelfranchi C (2007) The cognitive structure of surprise: looking for basic principles. Topoi 26:133–149
5. Partridge D (1985) Input-expectation discrepancy reduction: a ubiquitous mechanism. IJCAI, Los Angeles, pp 267–273
6. Wiggins GA (2006) A preliminary framework for description, analysis and comparison of creative systems. Knowl-Based Syst 19:449–458
7. Macedo L, Cardoso A (2001) Modeling forms of surprise in an artificial agent. Structure 1:C3
8. Kahneman D, Tversky A (1982) Variants of uncertainty. Cognition 11:143–157
9. Brown DC (2012) Creativity, surprise & design: An introduction and investigation. 2nd Int. Conf. Des. Creat. ICDC2012 Glasgow, UK
10. Maher ML (2010) Evaluating creativity in humans, computers, and collectively intelligent systems. In: Proc. 1st DESIRE Netw. Conf. Creat. Innov. Des., Aarhus, Denmark, pp 22–28
11. Macedo L, Cardoso A, Reisenzein R, Lorini E, Castelfranchi C (2009) Artificial surprise. Handb. Res. Synth. Emot. Sociable Robot. New Appl. Affect. Comput. Artif. Intell. IGI Glob. Hershey USA
12. Grace K, Maher ML, Fisher DH, Brady K (2014) A data-intensive approach to predicting creative designs based on novelty, value and surprise. Int. J. Des. Creat. Innov., Taylor & Francis, Accept. Publ
13. Pearce MT, Wiggins GA (2006) Expectation in melody: the influence of context and learning. Music Percept Interdiscip J 23:377–405
14. Jennings KE (2010) Developing creativity: artificial barriers in artificial intelligence. Minds Mach 20:489–501
15. Boden MA (1990) The creative mind: myths and mechanisms. Basic Books, New York
16. Gärdenfors P (2004) Conceptual spaces: the geometry of throught. MIT Press, Cambridge
17. Fisher DH (1987) Knowledge acquisition via incremental conceptual clustering. Mach Learn 2:139–172
18. Leouski AV, Croft WB (2005) An evaluation of techniques for clustering search results. Technical report IR-76, Department of Computer Science, University of Massachusetts, Amherst

Computational Design Creativity Evaluation

David C. Brown

Abstract This paper presents a simple framework for computational design creativity evaluation, presenting its components with rationale. Components are linked to recent computational creativity research in both art and design. The framework assumes that the product, not the process, is being evaluated, and that evaluation is done by comparison with descriptions of existing products using a set of aspects that each suggest creativity. Not every evaluation will use all of the components of the framework. It can be used to guide or assess design creativity research.

Introduction

This paper is concerned with the Computational Design Creativity (CDC) of engineered products and, specifically, the evaluation of creativity. That is, how does a computer system know when it "sees" a product that people will tend to label it as creative? A key issue is what the appropriate evaluation methods and measures are. Another is to identify the possible evaluators. Yet another is to describe what their knowledge might be.

In general, the issue is to determine all the types of ingredients involved in such an evaluation: hence the development of the ideas in this paper. The design creativity evaluation framework presented here consists of components, but not every evaluation will use all of the components.

There is no such thing a priori as a "creative computational design system", only one that produces artifacts that are evaluated as creative. This suggests that any CDC system must 'design *with* evaluation' and 'design *for* evaluation'. That is why this topic is so important.

Products, design descriptions, and design processes, are labeled as creative based on evaluation [1–3]. This framework is *not* concerned with processes, but it is possible that it might apply to them.

D.C. Brown (✉)
Worcester Polytechnic Institute, Worcester, MA, USA
e-mail: dcb@cs.wpi.edu

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_12

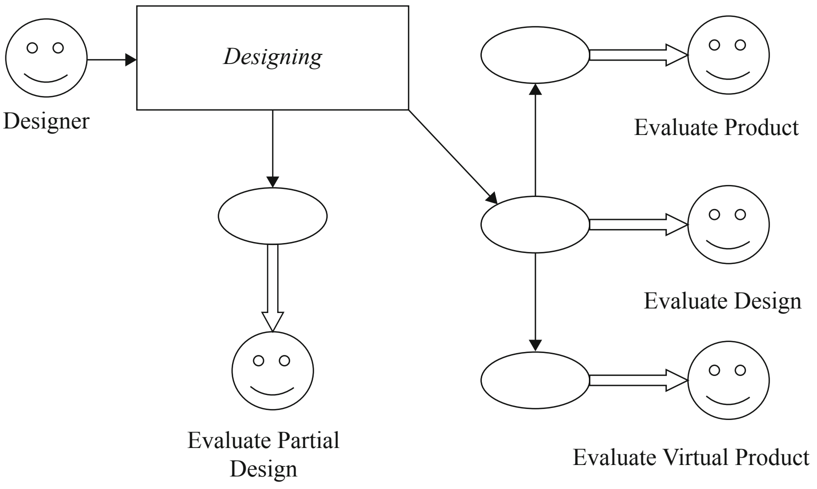


Fig. 1 The participants in evaluation

The main assumptions are that evaluation is done *by comparison with descriptions of past or existing designs/products*, using a set of evaluative *aspects*, such as ‘novelty’, where each aspect may suggest creativity.

At this point it is still safe to say that humans are better creativity evaluators than machines [3, 4], and that (as with much of AI) the best initial approach to full computational evaluation of designs is to firmly base it on whatever we can determine about the way that humans do evaluation.

Figure 1 outlines the participants and their roles in the evaluation process. The design is assumed to be a description, while the product is a thing. The design might also be rendered virtually, and then evaluated. Despite being drawn with faces, some of these evaluations can be carried out by a CDC system: for partial designs and complete designs especially. An important issue is what each evaluator knows about the designer and vice versa.

There are many different factors that play a part in evaluation. For example, the time at which the evaluation is done is important for a CDC system. What varies then is how much of a design description is available. During designing it may be partial. After designing it should be complete. However, when presented with just the complete design description (or the actual product), the requirements may not be available, causing it to be much harder to evaluate relative to original intentions.

Evaluation of partial designs or of design decisions made during designing will need to be in terms of their *likely contribution* to the eventual perceived creativity of the final product. As this is difficult to predict, and such evaluation requires accurate expectations [5], partial designs are hard to evaluate. This may be made harder by the “new term” problem [6] where some previously unknown thing, property, or relationship is introduced during designing and must be recognized in order to make an effective evaluation.

Evaluation for creativity after the product has been designed is the norm. However, creativity evaluation of sub-parts and sub-systems during designing seems necessary in order to help drive the process towards a creative conclusion. Consequently, evaluations both *during* and *after* designing are needed for CDC systems.

The framework for evaluation that is proposed here is *simple* in that it provides a framework tuned to designing that has relatively few components, but it is *challenging* because to do computationally all that it suggests is currently very difficult. Galanter [7] states that "evaluation victories have been few and far between". We expect it to remain difficult for quite a while. However, the framework should encourage researchers to try to implement all of its parts, rather than just a few. By specifying how each component of the framework is realized, it should allow researchers to classify how evaluation is done in existing and planned CDC systems. The framework addresses level 8, CC processes, in Sosa and Gero's [8] Multi-level Computational Creativity model, with a nod towards levels 4 and 6, Product, and Cognition.

The references provided in this paper are a resource that should allow easy access to the current literature on design creativity evaluation, focusing primarily on the product, hardly at all on the process [2], and not at all on the designer's personality [9, 10].

The field of Computational Creativity has probably advanced most in the area of the arts, with computer systems that paint, draw, write poetry, and interact with visitors/viewers. There appears to be very little reference by the design researchers to artistic creativity research, and vice versa. That provides additional motivation to do so here.

Computational artistic creativity researchers refer to "Aesthetic Evaluation", as the arts are more concerned with beauty, and with taste [7]. The design area focuses on use and function, in addition to novelty. In the arts, novelty is a 'given' and function is usually secondary: however, that's not to say that an artistic work has no function.

Romero et al. [11] argue that aesthetic judgment should apply to 'form' not 'content'. So, for example, an artwork with the intended effect (function) of providing propaganda (content) should be judged aesthetically solely by the way it looks. However, this position needs to be softened for interactive digital art (see below). In contrast, for engineering design, usefulness and functionality are usually included in any evaluation.

We continue by sampling work on the evaluation of artistic creativity, in order to augment existing concepts from computational design creativity research. We then present the simple framework for design creativity evaluation, followed by an explanation of its components.

The Evaluation of Artistic Creativity

We proceed by reviewing *some* of the work in computational artistic creativity that relates to evaluation, evaluation knowledge, types of participants in the creative process, and creativity models. The hope is that we will find ideas to inform a framework for evaluation in design. Note that this is not intended to be a comprehensive review.

Evaluation Knowledge

Cohen et al. [6] presents a discussion of many aspects of evaluation in creativity. They introduce the idea of the evaluator's perspective or role, and the notion that what gets evaluated, and how, may change because of that role. They propose that a role may be "creator or designer, viewer, experiencer, or interactive participant".

They point out that much evaluation is concerned with "prediction": i.e., what impact a decision made during designing might make on the reaction to the final artwork, or how the *whole* artwork might be evaluated by others. Evaluation during designing is "directed to how to proceed".

Cohen et al. point out that prediction of an *emotional* response might be needed, not just an evaluation based on some "aesthetic principles" (what we call "aspects" in the framework below).

They also indicate the importance of the knowledge needed for evaluation: the artist's knowledge, knowledge about the artist, about cultural norms, the factors driving the creative act (which would include Requirements for designing), and the observer's knowledge. Note that the knowledge of the various evaluators may overlap but it is not likely to be the same. Some knowledge might be required to turn quantities (e.g., product dimensions) into qualities that can form part of an evaluation (e.g., stylish shape). They conclude that "Knowledge and experience emerge as decisive factors in producing artifacts of high creative value" (p. 98).

Evaluating Creativity

In Candy's "Evaluating Creativity" [12] she briefly discusses "Creativity in Design" but with almost no mention of the Engineering Design issues or the references introduced in this paper. Candy contrasts "digital arts" with Engineering Design with the former having the "designer and implementer" being the same person.

She uses the matrix for evaluating creativity proposed in Candy and Bilda [13] that is tuned to Art, and Interactive Digital Art in particular. Their model uses the well-known People, Process, Context (also known as "Press"), and Product divisions [14], with evaluation criteria added for each.

For Product evaluation she proposes Novel, Original, Appropriate, Useful, Surprising, Flexible, Fluent, and Engaging as the evaluation aspects. Interestingly, she also proposes measuring the *interaction* with the product with additional aspects: Immediate, Engaging, Purposeful, Enhancing, Exciting and Disturbing. Note the use of both functional and emotional terms. With less emphasis on interactive artwork, Candy and Bilda [13] also propose evaluating based on

Composition, Aesthetic, Affect, Content and Technique. Note that here too the suggestion that an attempt be made to evaluate the emotional impact being made on the viewer/user. This is not just important for art, but for design too [15].

Candy also lists the people involved as artist/designer, participant/performer, and sometime "jury". For engineering design this reduces to designer and user, but rarely a jury.

Interaction with Art

Interactive art is often seen as providing "creative engagement" [13] by the viewers, or participants. In engineering design, there is much less concern about the creative nature of the 'use' of a product (i.e., whether it can be used or interacted with *creatively*).

However, the user or some other external evaluator will interact with a product in order to evaluate it, either by *viewing* it, *touching* it, *lifting* it, *manipulating* it, or *using* it for some task [16]. These interaction-based aspects are already well represented in most published sets of evaluation criteria (e.g., [17]): in fact without such interaction there can be limited evaluation of an implemented product. This is true even if the interactions are visualized or mentally simulated based on design descriptions or CAD models.

In design research we usually consider "a design" to be a description, and not the actual product. Descriptions do not usually exist as a deliverable in most artistic endeavors. For designing then we might evaluate at either the partial or full description stage, at the virtual implemented (e.g., CAD model) or the real implemented product stage. Hence we may need to be quite precise when talking about interaction enabling evaluation.

Candy and Bilda [13] have very distinct meanings for "interaction". They refer to "static", "dynamic-passive", "dynamic-interactive" and "dynamic-interactive varying" types of interaction with art. These correspond roughly to the "viewing" to "using" interactions mentioned above for designs.

They offer some guidelines for supporting/enhancing creative engagement with an interactive system. For example, "set expectations of the audience before they start to interact", suggests the possibility of improving eventual evaluation by invoking knowledge of familiar, existing products with the chosen structure, function or behavior of the new product. It also suggests taking advantage of perceived affordances or providing "signifiers" [18].

Computational Creativity Theory

Colton et al. [2] present a framework that is the basis of their Computational Creativity Theory, intended to "describe the processing and output of software designed to exhibit creative behaviour." Clearly this work is relevant for CDC systems even though there is some bias towards artistic creativity, as well as bias towards process and not just product. Note that their framework is not a framework about evaluation, and evaluation of the kind needed for product design is not stressed.

Colton et al. divide generative/creative "acts" into types g="ground" (producing new artifacts) and p="process" (producing new processes). These are coupled with what type of thing is being manipulated: expressions of concepts (E), concepts (C), aesthetic measures (A), and framing information (F): referred to as "FACE" tuples. This allows a rich set of creative actions to be described: in particular, and most original, actions that produce methods for generating concepts, or that produce methods for generating aesthetic measures. Actual things, such as concept descriptions, "expressions" (i.e., instances) of concepts, or measures, are seen as the input and output of these acts. Tuples of acts, such as $<A^g, C^g, E^g>$, together indicate a more complex creative act consisting of generative acts with information flowing between them.

From the examples provided it appears that evaluations apply to the whole new expressions of concepts (i.e., "designs" in our terms). If so, this misses the possibility that intermediate designs or design decisions might be evaluated.

With regard to evaluation, this paper suggests that there might be a minimal acceptable aesthetic (evaluation) level below which a result can be considered as "too low quality". This seems very context-dependent: certainly in terms of the experience/knowledge of the evaluator. However, it does permit meta-level measures such as "average", "best ever" and "precision", which are an important way to evaluate a CDC system *over time*.

Colton et al. provide little evidence for which aesthetic measures may be appropriate (apart from novelty); nor how they might be combined. They instead propose that an "audience" judge both how much they "like" a creative act, and how much "cognitive effort" they were prepared to spend understanding that act.

This proposal is supposed to prevent an audience from having to "evaluate creativity directly". However, by using a 'profile' or 'fingerprint' of evaluation aspects a CDC system could judge ("directly") when a partial or complete design would be likely to be evaluated as creative. If necessary, it might even be able to calculate a creativity score based on the intended use of the design or the characteristics of the intended users. Note that use of evaluation aspects in our framework does not presuppose such a calculation by human or computer.

Colton et al. also propose the existence of a *distance measure* that can be applied to creative acts (including output), and a *similarity threshold* for distance, below which one act is deemed too similar to another, and therefore of less worth. An *upper threshold* can be used to determine whether two creative acts are similar enough to even be sensibly compared. These are useful concepts, but are probably aspect-specific, as different aspects will focus on different features and attribute of the design.

These thresholds do allow for some interesting hypotheses about the stages of development of a creative software system, as well as some general metrics that apply to groups of evaluators, such as judging whether a system's creative act has an impact that provokes "shock", provides "instant appeal", or is prone to "triviality".

Their paper argues for not using measures of the "value of solutions" (how well it solves a problem) in favor of using the "impact of creations". The authors appear to be using a very specific meaning for "value" so this use is consistent with their proposals. However, for design solutions, how well the problem is solved can be determined relative to requirements and to actual usage scenarios. Not only can a design solve a problem (satisfy a need) it may also have perceived or real "value". With regard to "impact" they also have a specific meaning in mind, referring more to the impact of creative acts, rather than products.

Colton et al. do not separate out the knowledge needed by a system for an artifact that will be evaluated by different types of evaluators. Having a model of the evaluator can change the action of a designer or design system. Similarly, an evaluator's judgments will change depending on his/her/its model of the designer. Thus any framework of design evaluation needs to include these knowledge possibilities.

There is a rich history of considering types of knowledge and their roles during designing, such as the roles for knowledge in design reasoning [19], knowledge level descriptions of designing [20, 21], and types of knowledge during learning while designing [22].

Colton et al.'s work provides an excellent beginning to a theory of computational creativity, with strong bias towards creative processes, and some bias towards art. Even though it may provide a framework in which evaluation can occur, it is not a framework for evaluation itself. Their "creative acts" should provide a way to evaluate the development of creativity in a complete software system, which is their goal. For designs it seems self-evident that people can evaluate the creativity of a product without knowing anything about the design or manufacturing process. This is the normal situation for products, hence this paper's focus on the product.

Work on aesthetic evaluation in computational artistic creativity provides ideas about who might be evaluating, some particular views about how a user might interact with a "product", and suggestions about methods for evaluation. These inform the framework proposed in the next section.

A Creativity Evaluation Framework for CDC Systems

In this section we present a set of components involved in design evaluation, focusing on the actions, the knowledge needed, and the context for evaluation. In this framework we refer to "evaluator", considering it mainly to refer to a single evaluator that is not the designer, but in some circumstances they will be the same agent.

Note that it is assumed that evaluation is done by comparison with descriptions of past or existing designs/products: hence this does not appear as a component of the framework. We assume that appropriate design descriptions can be searched for, found, organized, selected, or recreated when needed. Apart from suggesting that the components given below influence this activity, no claims are being made about how this basis for comparison is actually produced. We assume that the description languages for the items in the basis are appropriate and comparable.

The Framework

The proposed framework for creativity evaluation for CDC systems has the following components:

1. *a description of the complete or partial artifact being judged, and/or the actual artifact*;
2. *the agent judging* (*i.e., person, computer program, or group*);
3. *the temporal basis for comparison* (*e.g., the point in time or the period*);
4. *the source of the design basis for comparison* (*e.g., personal, group, industry, global*);
5. *the set of "aspects" to include in the evaluation* (*e.g., novelty, surprise, style, utility, etc.*);
6. *the method of evaluation for each aspect*;
7. *the method used to combine the evaluations of the aspects* (*if one exists*);
8. *domain knowledge used by the evaluator* (*i.e., their amount of domain expertise*);
9. *the evaluator's knowledge about the designer* (*e.g., performance norms for the designer's level of expertise*);
10. *knowledge about the audience at whom the evaluation is aimed*;
11. *knowledge of the design requirements*;
12. *knowledge of resource constraints* (*e.g., materials, or available design time*);
13. *the evaluator's knowledge of the artifact due to the type and duration of experience with it*;
14. *the evaluator's knowledge of the design process*;
15. *the emotional impact of the design on the evaluator*;
16. *other contextual factors that may have an impact* (*e.g., culture*).

An Explanation of the Framework

Creativity evaluation depends on the components listed above. We will add some explanation about each one in turn. No detailed consideration will be given here as to how easily each might be adopted, adapted and implemented for CDC system use. The author is fairly convinced that they all could be implemented, with varying degrees of ease and precision.

Clearly, not every component of the model needs to be included in every CDC evaluation, and not every attribute of an artifact needs to be included in an evaluation.

A Description of the Complete or Partial Artifact Being Judged, and/or the Actual Artifact

The evaluator will judge a design or a partial design. A CDC system deals with descriptions, although it is possible that, in the future, CDC systems might be 'grounded' by visual and tactile ability that could be applied to (perhaps computer generated) prototype artifacts. Humans are more likely to deal with artifacts, but can also judge descriptions. For complete evaluation it is necessary to have multi-level descriptions (e.g., showing subsystems), and descriptions in terms of Function, Behavior and Structure (see Erden et al. [23] for a review).

Some work on creativity evaluation considers a *set* of designs from a single designer (e.g., in response to the same requirements). However, even though the judgment is about the set, the essence of this approach is still comparing a single design against others.

The Agent Judging

A 'judge' of some sort evaluates a design for creativity: that could be a person, a group, or a computer program. A CDC system might have knowledge and reasoning based on any of these. In a multi-agent design system, for example, both the designer and the judge might be computer programs.

The Temporal Basis for Comparison

The temporal basis is a point in time, or a period, on which to base the samples of related objects, prototypes, or standards [24] that are used for comparison with the design being judged [25].

The judgment of creativity is a moving target, as any new artifact could be added to the basis for comparison, which changes any subsequent judgment of the same (or similar) artifact. Of course, that depends on the judging agent having access to the modified basis [26]. Note that any changes to the basis (e.g., its organization) due to the addition of a new design may have meaning, such as indicating novelty [27].

Creativity evaluation is always a judgment at a time. It can be, and usually is, set to "now", but it could be set in the past, yielding a hypothetical evaluation about whether an artifact might have been seen as creative at some past time. For a CDC system we're considering "now" to be at the time of designing. By setting both the temporal and the source bases appropriately, evaluations of "rediscoveries" can be made [28].

The basis is often sourced from a time period. The normal period tends to be the maximal one of all history: at least back to the point where the technology makes comparisons irrelevant (e.g., laser cutters compared to flint knives) (*cf.* Colton et al.'s "upper threshold"). The temporal basis can be especially important for evaluating novelty [29].

The Source of the Design Basis for Comparison

This component refers to from where the design basis is gathered. It might be strictly personal; in which case the basis is only designs produced by the designer (see [30]). This corresponds to evaluating for Boden's P-Creative designs, where P stands for Psychological [1]. By widening it to a group, industry, or global, and by using "all history" as the temporal basis, we are evaluating for H-Creative designs, where H stands for Historical.

This makes it clear that P- and H-creativity are labels for very particular areas of the *time-and-source space* of possible bases for comparison: i.e., just referring to P-Creative and H-Creative is much too simple.

As already mentioned, the actual basis for comparison is not considered to be part of this framework as it is considered to be 'generated by selection' depending on the time-and-source space specified.

In contrast to the evaluation of a single design against past designs, which might be called "absolute" creativity, some researchers evaluate a design, or a set of designs, against designs produced (often at the same point in time, and from the same requirements) from other designers in the same cohort [31–33]. This is often associated with the evaluation aspects of quantity and variety of the ideas generated. This limited comparison might be called "relative" creativity. However, both types can be accounted for by using the time and source components in this framework.

The Set of "Aspects" to Include in the Evaluation

There are a very wide variety of different aspects mentioned in the literature that might be included for creativity evaluation, such as novelty, surprise, style, functionality, and value [29, 32–41]. The field of artistic creativity evaluation has alternative (but overlapping) sets of aspects.

Besemer [17] has one of the most long-lived (from 1981) and well tested lists of aspects organized into categories. She includes Novelty (Surprising, Original), Resolution (Logical, Useful, Valuable, Understandable), and Style (Organic, Well-crafted, and Elegant).

Cropley and Kaufman [42] go even further, proposing 30 indicators of creativity that they experimentally reduced to 24. Their categories of aspects include Relevance and Effectiveness (Performance, Appropriateness, Correctness), Problematization (Prescription, Prognosis, Diagnosis), Propulsion (Redefinition, Reinitiation, Generation, Redirection, Combination), Elegance (Pleasingness, Completeness, Sustainability, Gracefulness, Convincingness, Harmoniousness, Safety), and Genesis (Vision, Transferability, Seminality, Pathfinding, Germinality, Foundationality).

The Method of Evaluation for Each Aspect

Whichever aspects are included in a CDC system, an actual evaluation needs to be made using those aspects [43]. For example, an artifact needs to be judged for its novelty/originality [29, 38, 40, 44, 45] or for whether it is surprising [46, 47]. Different evaluation methods are possible for both of these aspects.

For example, novelty can be evaluated using a frequency-based approach that detects how many other designers have produced a similar design: the fewer the better. Novelty can also be estimated by accumulating the distance between the new design and the most similar design(s). If past designs are clustered, with some stereotypical design representing each cluster, the distance between the new design and the closest stereotype might also be used to evaluate novelty. Alternatively, if the new design causes re-clustering then this might indicate novelty. Finally, novelty might be measured by the amount of variation from the path of changes to features that designs with this functionality have exhibited over time: large variation suggests novelty. We conjecture that different methods will also exist for other aspects besides novelty.

In addition, depending on the design description used, it may be possible to apply the evaluation of aspects to different levels of abstraction in the description [32, 48, 49], and to descriptions that include Function, Behavior and Structure [40].

The Method Used to Combine the Evaluations of the Aspects

Overall evaluations have strengths; therefore artifacts may be seen as more, or less, creative – i.e., it isn't a Boolean decision. However, if many aspects are evaluated this will produce a 'profile' of the amount of creativity demonstrated across all those aspects, not a single result [17]. Evaluation in a single, combined dimension results from the evaluator's biases about how to combine different aspect evaluations [31, 32, 37, 40].

Even if a particular evaluating agent is being modeled (e.g., an actual user or group of users), this combination method may not exist explicitly. Evolutionary methods have been used to produce combinations of aspects with some success [7] but often the methods of combination they produce "seem alien". Learning systems exist that extract and use features to do "aesthetics-based evaluation" [11, 50]. Fuge et al. [51] describe a method that is able to learn to mimic expert creativity ratings, such as "variety" scores.

A complex issue regarding combining evaluations that needs addressing is how the separate evaluations of creativity in the Function, Behavior and Structure levels affect each other and the evaluation of the whole artifact. For example, a candle that produces sparks on the hour to indicate time provides a standard function by behaving in a novel way, with only a slightly new structure: how creative is that?

The Domain Knowledge Used by the Evaluator

It is well established in the literature (see [43]) that the amount of domain expertise that the designer has makes a big difference to their potential for creativity. However, to *fully* appreciate a design the evaluator needs to (at least) match their level of sophistication. For example, expert evaluators may know about complex electromechanical devices: less expert designers may only know about Legos. Hence the nature and amount of the evaluator's domain knowledge will make a big difference to the evaluation [42]. Note that this need not be put explicitly into a CDC system – in fact it may not be able to be – but it might be accumulated using machine learning.

The Knowledge About the Designer

Knowledge of the capabilities of the designer may play a role in creativity evaluation: for example, the evaluator might be able to recognize Transformational creativity [1, 52]. Also, knowing the performance norms for the designer's level of expertise is important. Consider a design description of a building from a 10 year old child versus a design description from an excellent Architect. An excellent child might be very creative relative to what they've already done (P-Creative), while an excellent architect is more likely to be judged as very creative relative to what everyone else has already done (H-Creative).

The Knowledge About the Audience at Whom the Evaluation Is Aimed

The evaluation must be understandable by the recipient of the evaluation. What you'd tell a child would be different from what you'd tell an expert. The conjecture is that this is not just a matter of the type of language used for the evaluation report, but that the actual evaluation might vary. For example, if a simple Yes/No or numeric position on a scale answer is desired then a powerful general technique such as CSPs, Neural Nets, or Evolutionary Computing might be used for the evaluation, as rationale for either the design or the evaluation is not needed, nor available. If the evaluation is for an expert, then it might be provided in technical terms, and mention product features, for example: whereas an evaluation of a process for an expert might mention ingredients such as selection, planning, evaluation, constraint testing, patching, failure handling, etc.

The Knowledge of the Design Requirements

Do the 'requirements' for the product, possibly including the intended function, need to be known to evaluate creativity? We argue that it is not necessary, but it should be helpful, as it allows the basis for comparison to be more precisely selected.

The Knowledge of Resource Constraints

If an evaluator understands how a designer dealt with resources constraints, such as limits on material availability or limited design time, it can affect their creativity evaluation.

The Evaluator's Knowledge of the Artifact Due to the Type and Duration of Experience with It

An evaluator might read the design description, see the artifact, touch the artifact, manipulate the artifact, or actually use the artifact [16]. This affects the completeness of their understanding of the artifact, and therefore their evaluation. Ludden's example involves 'surprise', but other aspects could also be affected.

Cohen et al. [6] conjecture about the computer 'experiencing' a design, and whether perception is required in order to do evaluation that matches what humans do. A CDC will need to have the equivalent of the ability to 'imagine' a design when given a design description, in order to evaluate its look or feel, its organic qualities, or its use.

The Evaluator's Knowledge of the Design Process

Colton [53] argues that, especially for artistic products, knowledge of the process is extremely important for the evaluation of creativity. However, it is clear that a very novel and interesting process might result in a not very creative design. We include this component of the framework for completeness. For many of the researchers referenced in this paper, this component is not essential for the evaluation of designed artifacts.

The Emotional Impact of the Design on the Evaluator

There is an increasing amount of interest in the emotional impact of designs [54]. But what is the role of emotion in the evaluation of creativity? The "impact" on the evaluator does play a role [34] but how do fun, cuteness, cleverness, memories, or jokes play a role in evaluation? Horn and Salvendy [36] claim that arousal and pleasure influence the evaluation of product creativity. Cropley [55] points out that "departure from the usual arouses discomfort" and perhaps "departure from the usual arouses excitement". Datta et al. [50] relate emotional impact to aesthetics evaluation.

Norman [15] proposes that initial design evaluation takes place at a sensory/visceral level, where 'appearance' can evoke an emotional response. The behavioral level of evaluation is concerned with usability: a very good or very bad experience can evoke corresponding emotions (e.g., frustration). The reflective level of evaluation is about prestige and desirability: i.e., how having the product makes one feel, and the degree of good taste that it might convey. It's clear that some of the aspects introduced above (e.g., Besemer's "Style" dimension) might act as a proxy for some of the emotional response, while the prestige associated with a particular product or designer could be estimated.

In general, emotional impact is clearly a difficult component to include in a CDC system. However, it might be detected or estimated in a variety of ways: *direct* methods such as eye movement/dilation, galvanic skin response, and brain wave changes; *indirect* methods such as measures of similarity to products that have known emotional impact, or classifiers trained using machine learning from user reporting.

Other Contextual Factors That May Have an Impact

This, we must admit, is a catch-all category. However, there are factors, such as culture, that may play a role in evaluation that could go here, as it isn't clear that they always apply or are a main influence for CDC systems.

One such factor is whether a past artifact has been acknowledged as creative: perhaps to the point of it being a disruptive product, changing the direction of future artifacts in the same category. Sternberg et al. [28] describe this as "propelling" a field. This knowledge might be used to suggest that a new artifact might be creative by analogy: if the new artifact (X) has 'similar' characteristics to an existing artifact (Y), and Y was seen as creative and influential in the past, then perhaps X will be seen as a creative influence. Such an evaluation would be helped by having similarity information [56] available, and knowledge about the design time. Of course, too much similarity decreases novelty.

Some evaluation schemes include "usefulness" and the "importance" of the use as evaluation aspects (e.g., [40]). This might be measured in terms of actual use, or potential use. As the artifact has just been, or is still being, designed, evaluating "actual" use will not be possible. There needs to be enough knowledge included during the design creativity evaluation process to estimate how much it might be used, and weight it by "importance" or potential "impact".

Summary and Conclusion

The framework presented here differs from other work by focusing on artifact design, the different types of participants in the evaluation process, and the types of knowledge needed by the designer and the evaluator: in particular what each needs to know about the other.

This framework is intended to be used to guide or assess design creativity research, with the hope that it will eventually apply to CDC systems. The references should allow easy access to the current literature on design creativity evaluation. Given the number and difficulty of the components in the framework it is obvious that CDC systems still need a lot of work. The framework also makes it clear that how creative an artifact is may only be properly stated if the full context of the evaluation is included.

References

1. Boden MA (1994) What is creativity? In: Boden MA (ed) Dimensions of creativity. The MIT Press, Cambridge, MA, pp 75–117
2. Colton S, Charnley J, Pease A (2011) Computational creativity theory: the FACE and IDEA descriptive models. In: Proc. 2nd Int. conf. on computational creativity, Mexico City, pp 90–95

3. Hennessey BA, Amabile TA (2010) Creativity. Ann Rev Psychol 61:569–598
4. Amabile TM (1996) Creativity in context. Westview Press, Boulder
5. Grecu DL, Brown DC (2000) Expectation formation in multi-agent design systems. In: Gero JS (ed) Proc. artificial intelligence in design'00. Kluwer, Dordrecht, pp 651–671
6. Cohen H, Nake F, Brown DC, Brown P, Galanter P, McCormack J, d'Inverno M (2012) Evaluation of creative aesthetics. In: McCormack J, d'Inverno M (eds) Computers and creativity. Springer-Verlag, Berlin, pp 95–111
7. Galanter P (2012) Computational aesthetic evaluation: past and future. In: McCormack J, d'Inverno M (eds) Computers and creativity. Springer-Verlag, Berlin, pp 255–293
8. Sosa R, Gero JS (2013) Multilevel computational creativity. In: Proc. 4th Int. conf. on computational creativity, Sydney, pp 198–204
9. Eysenck HJ (1994) The measurement of creativity. In: Boden MA (ed) Dimensions of creativity. The MIT Press, Cambridge, pp 199–242
10. Charyton C, Jagacinski RJ, Merrill JA (2008) CEDA: a research instrument for creative engineering design assessment. Psychol Aesthet Creat Arts 2(3):147–154
11. Romero J, Machado P, Carballal A, Correia J (2012) Computing aesthetics with image judgment systems. In: McCormack J, D'Inverno M (eds) Computers and creativity. Springer-Verlag, Berlin, pp 295–321
12. Candy L (2013) Evaluating creativity. In: Carroll JM (ed) Creativity and rationale: enhancing human experience. Springer-Verlag, London, pp 57–84
13. Candy L, Bilda Z (2009) Understanding and evaluating creativity. In: Bryan-Kinns N (ed) Creativity & cognition 2009. ACM, Berkeley, pp 497–498
14. Rhodes M (1961) An analysis of creativity. Phi Delta Kappan 42:305–310
15. Norman DA (2004) Emotional design: why we love (or hate) everyday things. Basic Books, New York
16. Ludden GDS, Schifferstein HNJ, Hekkert P (2008) Surprise as a design strategy. Design Issues 24(2):28–38
17. Besemer SP (2006) Creating products in the age of design. How to improve your new product ideas! New Forums Press, Stillwater
18. Norman DA (2008) THE WAY I SEE IT: signifiers, not affordances. Interactions magazine. ACM 15(6):18–19
19. Brown DC (1992) Design. In: Shapiro SC (ed) Encyclopedia of artificial intelligence, 2nd edn. Wiley, New York
20. Smithers T (1996) On knowledge level theories of design process. In: Gero JS, Sudweeks F (eds) Artificial intelligence in design'96. Kluwer, Dordrecht, pp 561–579
21. Smithers T (1998) Towards a knowledge level theory of design process. In: Gero JS, Sudweeks F (eds) Artificial intelligence in design'98. Springer, Netherlands, pp 3–21
22. Sim SK, Duffy AHB (2004) Knowledge transformers: a link between learning and creativity. AIEDAM 18(3):271–279
23. Erden MS, Komoto H, van Beek TJ, D'Amelio V, Echavarria E, Tomiyama T (2008) A review of function modeling: approaches and applications. In: Goel A, Davis R, Gero JS (eds) Special issue on multi-modal Design, AIEDAM 22(2)
24. Redelinghuys C (2000) Proposed criteria for the detection of invention in engineering design. J Eng Design 11(3):265–282
25. Wiggins GA (2006) A preliminary framework for description, analysis and comparison of creative systems. J Knowl Based Syst 19(7):449–458
26. Sosa R, Gero JS (2005) A computational study of creativity in design: the role of society. AIEDAM 19(4):229–244
27. Maher ML, Brady K, Fisher D (2013) Computational models of surprise in evaluating creative design. In: Proc. 4th Int. conf. on computational creativity, University of Sydney, pp 147–151
28. Sternberg RJ, Kaufman JC, Pretz JE (2002) The creativity conundrum: a propulsion model of kinds of creative contributions. Psychology Press, Philadelphia

29. Maher ML, Fisher DH (2012) Using AI to evaluate creative designs. In: Proc. 2nd Int. conf. on design creativity (ICDC2012), Mexico City, pp 45–54
30. Jagtap S, Larson A, Hiort V, Olander E, Warell A (2012) Ideation metrics: interdependency between average novelty and variety. Int. Design Conf., DESIGN 2012, pp 1881–1892
31. Oman SK, Tumer IY, Wood K, Seepersad C (2013) A comparison of creativity and innovation metrics and sample validation through in-class design projects. Res Eng Des 24(1):65–92
32. Shah JJ, Vargas Hernandez N, Smith SM (2003) Metrics for measuring ideation effectiveness. Des Stud 24:111–134
33. Kudrowitz BM, Wallace DR (2012) Assessing the quality of ideas from prolific, early-stage product ideation. J Eng Design, Special Issue on Design Creativity, (first online preview), 1–20
34. Christiaans HHCM (1992) Creativity in design: the role of domain knowledge in designing. Uitgeverij Lemma BV, Utrecht
35. Dean DL, Hender JM, Rodgers TL, Santanen EL (2006) Identifying quality, novel, and creative ideas: constructs and scales for idea evaluation. J Assoc Inf Syst 7(10):647
36. Horn D, Salvendy G (2006) Product creativity: conceptual model, measurement and characteristics. Theor Issues Ergon Sci 7(4):395–412
37. Ritchie G (2007) Some empirical criteria for attributing creativity to a computer program. Minds Mach 17:67–99
38. Srivathsavai R, Genco N, Hölttä-Otto K, Seepersad CC (2010) Study of existing metrics used in measurement of ideation effectiveness. Proc. ASME 2010 Int. IDETC & CIE Confs., DETC2010-28802
39. Liikkanen LA, Hämäläinen MM, Häggman A, Björklund T, Koskinen MP (2011) Quantitative evaluation of the effectiveness of idea generation in the wild. Human Cent Design Lect Notes Comp Sci 6776:120–129
40. Sarkar P, Chakrabarti A (2011) Assessing design creativity. Des Stud 32:348–383
41. Lu C-C, Luh D-B (2012) A comparison of assessment methods and raters in product creativity. Creat Res J 24(4):331–337
42. Cropley DH, Kaufman JC (2012) Measuring functional creativity: non-expert raters and the creative solution diagnosis scale. J Creat Behav 46(2):119–137
43. Brown DC (2013) Guiding computational design creativity research. Int J Design Creat Innov 1(1):1–42
44. Lopez-Mesa B, Vidal R (2006) Novelty metrics in engineering design experiments. Int. Design Conf., DESIGN 2006, 557–564
45. Shelton KA, Arciszewski T (2007) Formal innovation criteria. Int J Comp Appl Technol 30 (1/2):21–32
46. Macedo L, Cardoso A, Reisenzein R, Lorini E, Castelfranchi C (2009) Artificial surprise. In: Vallverdu J, Casacuberta D (eds) Handbook of research on synthetic emotions and sociable robotics: new applications in affective computing and artificial intelligence. Information Science Reference (IGI Global), Hershey
47. Brown DC (2012) Creativity, surprise & design: an introduction and investigation. In: Proc. 2nd Int. Conf. on Design Creativity (ICDC2012), Mexico City
48. Nelson BA, Yen J, Wilson JO, Rosen D (2009) Refined metrics for measuring ideation effectiveness. Des Stud 30:737–743
49. Farzaneh HH, Kaiser MK, Schroer B, Srinivasan V, Lindemann U (2012) Evaluation of creativity: structuring solution ideas communicated in groups performing solution search. Int. Design Conf., DESIGN 2012, 1871–1880
50. Datta R, Joshi D, Li J, Wang JZ (2006) Studying aesthetics in photographic images using a computational approach. Lecture notes in computer science, 3953, In: Proc. European Conf. Computer Vision, Part III, Graz, Austria, pp 288–301
51. Fuge M, Stroud J, Agogino A (2013) Automatically inferring metrics for design creativity. ASME IDETC & CIE, DETC2013-12620
52. Ritchie G (2006) The transformational creativity hypothesis. N Gener Comput 24:241–266

53. Colton S (2008) Creativity versus the perception of creativity in computational systems. In: Proc. AAAI Spring Symp. on Creative Systems, Palo Alto, CA
54. McDonagh D, Denton H, Chapman J (eds) (2009) Special issue on ‘design and emotion’. J Eng Design 20(5)
55. Cropley DH (2009) Fostering and measuring creativity and innovation: individuals, organisations and products. In: Proc. Conf. Can Creativity be Measured? Section 17, Education and Culture DG, European Commission, http://ec.europa.eu/education/lifelong-learning-policy/creativity-conference_en.htm
56. Minsky M (2006) The emotion machine: commonsense thinking, artificial intelligence & the future of the human mind. Simon & Schuster, New York

Part IV
Design Processes – 1

A Multiagent Approach to Identifying Innovative Component Selection

Carrie Rebhuhn, Brady Gilchrist, Sarah Oman, Irem Tumer, Rob Stone, and Kagan Tumer

Abstract Though there is a clear correlation between profitability and innovativeness, the steps that lead to designing a "creative" product are elusive. In order to learn from past design successes, we must identify the impact of design choices on the innovativeness of an entire product. This problem of quantifying the impact of design choices on a final product is analogous to the problem of 'credit assignment' in a multiagent system. We use recent advances in multiagent credit assignment to propagate a product's innovativeness back to its components. To validate our approach we analyze products from the Design Repository, which contains thousands of products that have been decomposed into functions and components. We demonstrate the benefits of our approach by assessing and propagating innovation evaluations of a set of products down to the component level. We then illustrate the usefulness of the gathered component-level innovation scores by illustrating a product redesign.

Introduction

Innovation can revolutionize the way we approach and think about modern designs. Innovative designs can corner or even create markets, and can lead to higher profits for the companies that develop them. Companies that can develop more innovative solutions can have a definite advantage over companies that focus on more incremental approaches to design. But how can design engineers learn to innovate? This is an active field of research, and promoting *creativity* in the early stages of design has shown promise in promoting *innovativeness* in the final product [1]. In this work, we refer to *creativity* as the quality of the ideation process that leads to *innovation*.

There have been numerous studies linking design diversity with the use of solution-generation tools [2, 3]. These tools use design information from a Design Repository to offer component suggestions to fulfill a desired function. But often

C. Rebhuhn (✉) • B. Gilchrist • S. Oman • I. Tumer • R. Stone • K. Tumer
Oregon State University, Corvallis, OR, USA
e-mail: rebhuhnc@engr.orst.edu

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_13

there is a large selection of components available, offering no indication of whether a component solution has been used in innovative designs or commonplace ones. The designer is left to sort through *all* of the solutions to find the creative solutions. This is not a problem while the Design Repository holds a manageable number of design breakdowns, but because the Design Repository will grow over time it is essential to develop metrics that will allow designers to manage such suggestions.

To increase the potential for creating a market-changing product, we want to focus on evaluating *innovation* in stored design choices. The problem with using innovation as a metric is that it has a complex definition and no common mathematical form. Market success, product rareness, quality, company prevalence, usability, time on the market, and many other factors can impact the perceived innovativeness of a product. Although this quality has a varying definition and no mathematical formula, people can still identify innovation in products. Experts in design and marketing have created lists of identified innovative designs, published in *Time Magazine*, the IDSA IDEA Award, and *Popular Science*.

Innovation within a product can be identified by experts or by popular opinion, but it is much more difficult to quantify the amount that particular design decisions led to the product's innovativeness. Accurately quantifying this parameter for each design decision would allow us to sort design suggestions in such a way that designers would be able to quickly identify and leverage innovative design solutions. Previous work has focused on propagating innovativeness down to the component level through comparative analysis of standard and expert-identified innovative products [1]. In this approach, an 'innovation' parameter was calculated based on the rareness of the component in a population of designs and this was weighted in order to calculate the creativity of a product.

The set of decisions made by the designer and the resulting innovativeness of the product are correlated in some complex manner. But how can we evaluate the impact of each design decision? The field of multiagent systems faces a similar problem, referred to as *credit assignment*. In a cooperative multiagent system, each agent contributes in some way to the performance of all the agents. If there is a multiagent system coordinating traffic, it is easy to measure the performance of the entire system through total throughput of traffic. However it is difficult to quantify the impact of the individual decisions of each traffic-controlling agent on the total traffic throughput. Similarly, it may be easy to identify the innovativeness of a product, but difficult to quantify the impact of each design decision.

One way to handle this credit assignment is by using the *difference reward*. This reward evaluates the contribution of each agent by comparing the performance of the system to the performance of a hypothetical system in which the agent did not contribute. Because the difference reward can more specifically encapsulate the effects of an agent than a system evaluation, this reward has shown increases in learning speed, convergence, and solution quality over using the system reward. This accurate identification of impact is exactly what we need in order to identify the innovative impact of specific components.

The contributions of this work are to:

1. Form a method for the propagation of product-level innovation scores to individual design decisions based on techniques present in multiagent learning.
2. Present a new decomposition of innovation scores based on the component frequency of appearance within the dataset, which we use as the basis for our difference reward.
3. Present results from using this technique with real design data and three separate sets of human-generated innovation scores.
4. Demonstrate the potential use of this new component-level innovation data in a proposed redesign of a product from the Design Repository.

Background

This work draws on principles from design engineering as well as multiagent learning. On the design side, we explain the general structure and benefits of the Design Repository housed at Oregon State University and we introduce *novelty*, a metric developed by Shah et al. [3] for quantification of the uniqueness of a design. On the multiagent side, we identify difference rewards as a learning-based method for innovation score propagation.

Storing Innovation Information

The Design Repository at Oregon State University contains a wealth of information about modern products [4]. It currently contains over 180 products with more than 6,000 artifacts. The purpose of the repository is to store previous design information. Products are disassembled into their individual components. Information about each component is recorded such as function, mass, dimensions, materials, manufacturing processes, failure modes. Functional models are created for every product, identifying the way that materials, signals, and energy change as they flow through the product. The functions assigned to the components are done so using the functional basis [5]. This allows for a common language for functionality that is universally understood.

The output from the repository is a morphological matrix containing all possible solutions for a given function (example: "Export") and flow (example: "Electricity"). A morphological matrix is a matrix of component solutions to given functions. This matrix can be used by the designer to select components and assemble them into a complete concept. Within the repository, in addition to the component the frequency that a component solves the given function is also provided. However, the repository does not yet have a way to capture the innovativeness of products and their components. With inclusion of innovative scores, the creative potential of concepts can be determined earlier on in the design process. In order to promote innovation-oriented designs, in this work we develop a mechanism to propagate innovation scores down to the component level.

Novelty Calculations and Creativity

The attempt to ascribe metrics to creativity and innovation is not a new study, and is a source of recent research in the design community [1, 6]. Quality, quantity, novelty, and variety were identified by Shah et al. [3] to evaluate a set of concepts. In particular, the metric of novelty has been implemented in recent work by Oman et al. [1] along with a weighting system in order to identify the creativity score of particular functions. The metric for novelty we use in this paper is similar to the one used in Oman et al., and is given by the equation:

$$S_{Nj} = \frac{T_j - R_j}{T_j} \tag{1}$$

where S_{Nj} represents the novelty, T_j is the number of designs in the set being examined and R_j is the number of times the component solution for function j is seen in the set. This novelty metric has been traditionally applied only over sets of products with a similar type or goal, however in our work we use it over a set of products of varying types and goals. This may mean that the elements performing functions in one device may not feasibly perform the same functions in another device, but a user may mitigate this potential problem by obtaining lower level functional basis descriptions from the Design Repository.

Previous work has focused on comparison of the functions of ‘common’ products to ‘innovative’ products [1]. In this way, functions unique to innovative products along with their frequency of appearance within the design set could be used to characterize their general impact of the innovation within products. In this work, we take this comparison idea and draw on the reinforcement learning concept of a *difference reward*, which has been shown in many domains to effectively quantify the impact that an agent has on a multiagent system [7].

Multiagent Learning and Difference Rewards

Multiagent reinforcement learning is a field of artificial intelligence that involves the coordination of distributed learners. Control of a complex system may be modeled as a set of less complex interacting entities which make autonomous decisions. These entities are called ‘agents’, and they interact with their environment through a process of sensing the world state, taking actions available within that state, and receiving rewards. Reinforcement learning in a cooperative multiagent system focuses on finding a set of actions that most benefits the collective system. Learning agents adapt their strategies through repeatedly taking actions and getting a reward for these actions. The Design Repository provides a large database over which to train agents in a *supervised learning problem*, which is a problem where an agent is trained based on expert guidance [8]. In our case the expert guidance is provided by the design examples within the Design Repository.

An agent learns a *policy*, which holds information on an agent's perception of the value of taking a particular action. This policy is updated each time the agent receives a reward for an action it has taken. We perform Q-learning adapted for a stateless game [9], which adjusts the policy according to the rule:

$$V(a) \leftarrow V(a)_{old} + \alpha\left(R(a) - V(a)_{old}\right) \tag{2}$$

where $V(a)$ is the expected value of the action, α is the learning rate, and $R(a)$ is the reward received for taking action a. α is a number on the range [0,1] which impacts the learning speed of the agent. Increasing the α parameter may increase the value that an agent puts on more recent information, while decreasing α lowers learning speed but increases the chance of long-term convergence to an optimal policy. Agents develop policies in order to better use their knowledge about the value of specific actions to select their next action. Due to the fact that we cannot evaluate an arbitrary design we constrain our exploration in a different way; we force action selection to match the products we have and then reward accordingly. By forcing the agents to learn by example, we are using a process called *supervised learning*.

In developing the policy of the agent, the reward mechanism used to adjust the policy can be crucial to both convergence properties and the quality of the overall policy developed. Difference rewards have been used in a wide variety of large multiagent systems including the air traffic control, rover coordination, and network routing to promote coordination among agents [7, 10, 11]. This reward works by comparing the evaluation of the system to a system in which an agent is not there, or replaced by some counterfactual which refers to an alternative solution that the agent may have taken [7]. Mathematically this is expressed as:

$$D_i(z) = G(z) - G(z_{-i} + c) \tag{3}$$

where $G(z)$ is the full-system reward and $G(z_{-i} + c)$ is the reward for a hypothetical system in which agent i was removed and its impact was replaced by a counterfactual c. In this work we use difference rewards to compare products with particular creative components to products in which the creative components are replaced by a 'standard' component.

Framing Product Design as a Multiagent Problem

To create a product, a designer must compose a set of components to fulfill a set of engineering constraints set by the customer. Functional requirements must be fulfilled by the components selected. Essentially, design may be broken down at a rudimentary level to function-satisfaction by selection of components; a process where an agent must select components to fulfill properties desired by the engineering requirements of the design. Our approach to product design is structured using three primary factors: products, functions, and components.

1. **Products:** Products represent a set of actions taken by a designer. Products feature a set of functions which they must perform, and a set of components which have been selected to satisfy these functions. We also assume that products have an associated innovation score. The creation of a product where the design process is modeled as a multiagent system represents the *joint action*, or set of all actions taken by agents, in a multiagent system.
2. **Functions:** The purpose of a product is to perform a set of functions. Because a product must perform a prescribed set of functions, this defines the requirements for the component-selection of the designer. In a multiagent system, the task of ensuring satisfying these functions in the creation of the product is given to the *agents* within the system.
3. **Components:** Components within a design represent design decisions, and can be used to perform one or multiple functions within a design. In a multiagent framework, the set of components represent actions available to agents trying to fill functions.

In a step toward an autonomous design strategy, we assign our agents the task of *component selection*, which is to select components to satisfy a particular design requirement (function). One agent is assigned to select a component to fulfill each type of function (i.e., to import electricity) and has an action space which includes all components which could possibly accomplish this task (i.e., a battery or a power cord).

We use two datasets that include real designs from the Design Repository to train our agents, using the function-component matrices of these designs to guide an agent to take an action while designing a product. We then evaluate the designs using the product-level rewards given by these datasets, and reward the actions using two separate methods, which will be explained in more detail in later sections. Agents then performed a value update (Eq. 1) with a learning rate of 0.5 using this reward. Policy values were initialized to the novelty score at the beginning, as we assume novelty does have some impact on the innovative impact of a component.

Product-Level Innovation Score Decomposition

Novelty scores have been used in previous work along with a hand-tuned weighting in order to assess the innovation score of different components. From this fact, we postulate that $G(d) = f(S_N)$, where $G(d)$ is the total innovation score of a device, and $f(S_N)$ is some function of the novelty of all of the components within the device. As a starting point, we assume that the innovative impact of a component is proportional to its novelty score. This allows us to decompose the product-level innovation score down into component impacts I_i:

$$G(d) = \left(\sum\nolimits_{i=1}^{n} \frac{S_N}{S_{N_{all}}}\right) G(d) = \sum\nolimits_{i=1}^{n} I_i \tag{4}$$

where d is a design with n different functions, I_i is the impact of component i on the system score of the design, $G(d)$ is the full score for that design (recall that this is given by our data), $S_{N_{all}}$ is the sum of all novelties of the components, and S_{N_i} is the novelty score for component i.

We assume that we know the global score. Therefore this decomposition does not add information, but instead allows us to derive a difference reward which represents what would happen if one component was taken out and replaced by another component using these impacts I_i. The difference reward for this system can be derived by using the $G(z)$ formulation in Eq. 4 in Eq. 3, with some replacement component represented by the counterfactual term c. Making this substitution and simplifying yield a derivation of the difference reward:

$$D_i(d) = \left(\frac{S_{N_i} - S_{N_c}}{S_{N_{all}}}\right) G(d) \tag{5}$$

where S_{N_c} refers to the novelty score of the component which has hypothetically replaced the original component in the design. In difference rewards, the counterfactual term c can be defined in a variety of ways. In our domain, we employed two counterfactual reward formulations.

The first counterfactual used the most common component solution (i.e., the lowest novelty score), and we refer to the difference reward formulated using this method as $C_i^{pair}(d)$. This was applied to reward the agents for all component solutions based on the design. This is given by the equation:

$$C_i^{pair}(d) = \left(\frac{S_{N_i} - S_{N_{least}}}{S_{N_{all}}}\right) G(d) \tag{6}$$

where $S_{N_{least}}$ refers to the novelty of the most commonly-used component that fulfills the agent's assigned function.

The second counterfactual used all other component solutions as comparison. This meant that the set reward, $C_i^{set}(d)$ was recalculated $k-1$ times, where k refers to the number of different components that an agent had in its action space. The agent performed a value update $k-1$ times with each difference reward. The equation for this innovation evaluation was therefore given by:

$$C_i^{set}(d) = \left(\frac{S_{N_i} - S_{N_{alt}}}{S_{N_{all}}}\right) G(d) \tag{7}$$

where $S_{N_{alt}}$ refers to the novelty of an alternative component seen in the set of products. This is applied over all alternatives.

The table of values resulting from these calculations represent the innovation score estimation of the function-component pairs found in the set. These values are the agent's estimation of the creative impact of the function-component pair on the product-level design score, and are used to present our findings in the two datasets in later sections.

Training Data

The innovation score cannot at this time, from the data available in the Design Repository, be objectively calculated for a product. Though we cannot objectively calculate innovation, it is a quality that *humans* can readily assess. There are three different ways that we gather data on the innovation of different devices: expert innovation identifications found in consumer magazines, a survey of several college students, and a latent variable analysis using data from design engineering students. These are explained here in further detail.

Expert Data: Binary identifications of *innovative* products (innovative score = 1) were taken from *Popular Science*, the IDSA IDEA Award, and *Time Magazine*'s 50 Best, and contrasted with hand-selected *standard* products (innovative score = 0).

Survey Data: To make up for the binary nature and small size of the expert dataset, a survey was conducted involving ten participants rated a series of 50 designs on innovativeness, with five levels of innovative scores. The average of this data was then taken, and this was used as a product-level score for the designs.

Latent Variable Data: We performed a latent variable analysis across eight products. The entire dataset of 156 responses consisted of undergraduate engineering students who were taking a mechanical engineering design class. Each student responded to a set of questions which were targeted at identifying the impacts of three latent variables: innovation, product usability, and company profile.

Results

We divide the results generated with our multiagent system using the Design Repository Data by the external evaluation method used. The evaluation methods vary in products evaluated, type of innovation information, and demographic. Using the two difference reward formulations (D^{least} and $D^{average}$) and three datasets (expert, survey, and latent variable), we perform six different experiments exploring these parameters. Reward order impacts the results of reinforcement learning, so we perform experiments using a randomized order and take the average across 30 runs.

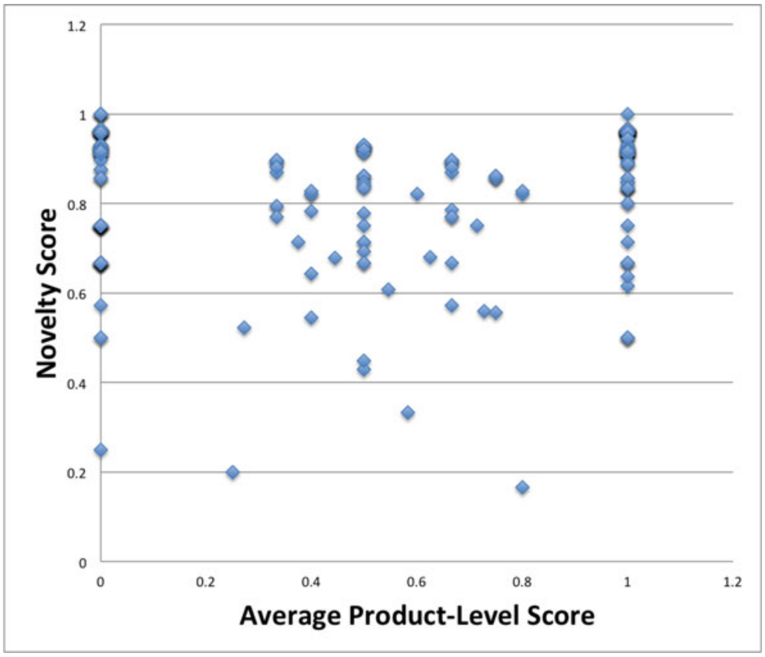


Fig. 1 Novelty for function-component pairs in the expert evaluation dataset

Expert Evaluation Dataset

As a baseline for our evaluations we analyze the novelty. As shown in Fig. 1, we plot the novelty of function-component pairs versus the average scores of products in which the function-component pair is found. Plotting the novelty against this average score solidifies a key hypothesis in the beginning of this work; *the frequency with which the component appears does not solely reflect the creative impact it has*. Novelty scores tend to be higher at the extremes of the average product-level score, with slightly lower values toward the center of the spectrum, but this correlation was weak in this data.

Figure 2a shows the results from our multiagent system method of evaluating the innovation at the component level. The data show an upward trend with two major outliers. The outlier shown at (0.750, 0.035) is the {channel material, reservoir} function-component pair, while the outlier shown at (1.000, 0.051) is the {control magnitude material, screw} function-component pair. Both of these show particularly high estimations of contribution to the innovation score. Innovation scores increase with average product-level scores, but also increase in spread with the product-level average.

The interesting part of this data is not necessarily in the trends, which will tend to increase with the average product-level score because of the learning mechanism.

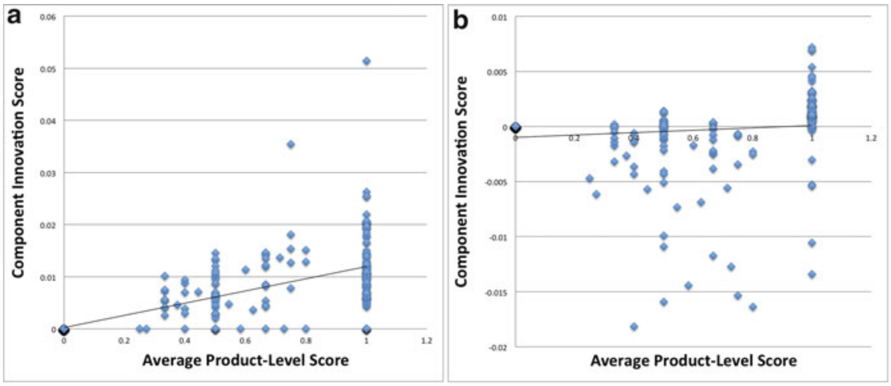


Fig. 2 Agent-estimated values using expert evaluation with (**a**) D^{least} and (**b**) $D^{average}$ reward

The outliers give a better indication of innovate component usage. In Fig. 2a there are two major outliers: the data points corresponding to {channel material, reservoir}, and {control magnitude material, screw}. While these components do not appear innovative in and of themselves, they may serve a particular function within their design in an innovative way.

Figure 2b shows much less correlation with the average product-level score than Fig. 2a. This is likely due to the fact that the $D^{average}$ difference reward has an average-novelty counterfactual, and therefore will tend to have an equal number of points which are positive and negative. Most of the positive data are collected at the higher end of the average product-level scores. The data show a slight upward trend with have a wide variation in values.

Survey Average Dataset

The novelty scores for the survey average dataset (Fig. 3) have high novelty scores at the extreme values of average product-level score, accompanied by lower levels of novelty at the lower product-level scores. The data show the diversity of the scores found in this dataset, as there is much less striation in the results as compared to the expert evaluation dataset. The novelty shows a general trend toward being high, but shows a dip on the range [0.05, 0.45].

The distribution of the data suggests that there may be *more* data in the middle of the distribution, but it remains that, apart from a small number of outliers, the novelty of components both in highly-innovative and fairly plain products tend to be higher. Conversely, there seems to be a loose trend that the novelty of components at the lower-middle range of the average product-level score tend to be *lower*, indicating that products made from frequently-seen components may have a lower product-level score on average.

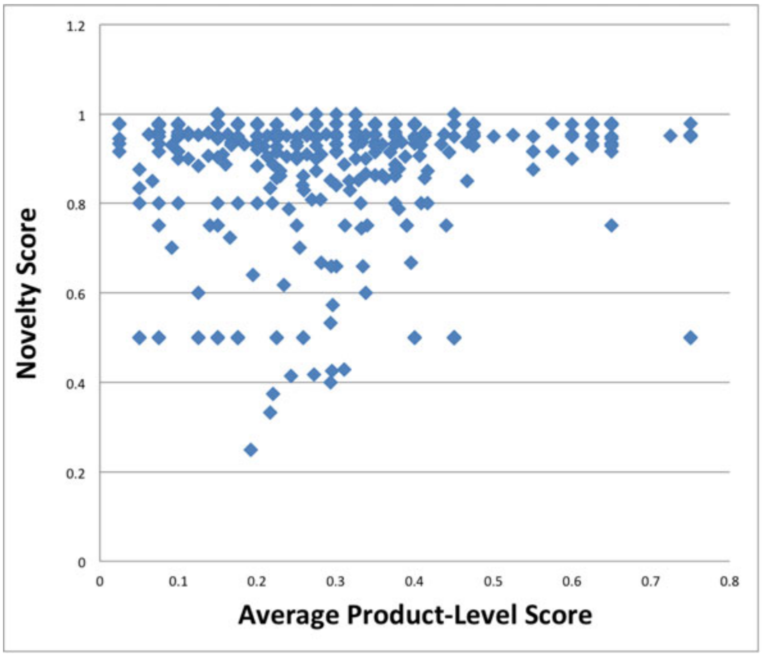


Fig. 3 Novelty scores for function-component pairs found in the survey dataset

The innovation scores shown in Fig. 4a show a clear trend; as product-level score increases, the component-level innovation both tends to increase and polarize. One particularly highly-estimated function-component pair at (0.600, 0.024) corresponds to {convert energy, handle}. At the lower levels of average product-level innovation, there tends to be a tight fit to the data. The variance in the data appears to increase with the average product-level score. At an average product-level score of 0.4, we see a dataset that disperses almost completely.

The innovation scores shown in Fig. 4b once again demonstrate the fact that roughly half of the function-component scores will score negatively. A trend that correlates somewhat with the novelty scores for this dataset (Fig. 3) is shown in the data; function-component pairs which appear at the mid-level of average product-level innovation tend to *detract* from the design of a component.

Survey Average Dataset

The latent variable dataset provides us a unique chance to test the multiagent system's response to negative product-level values. Scores in both our expert evaluation dataset as well as our survey dataset provide a training set which is bounded on the interval [0,1].

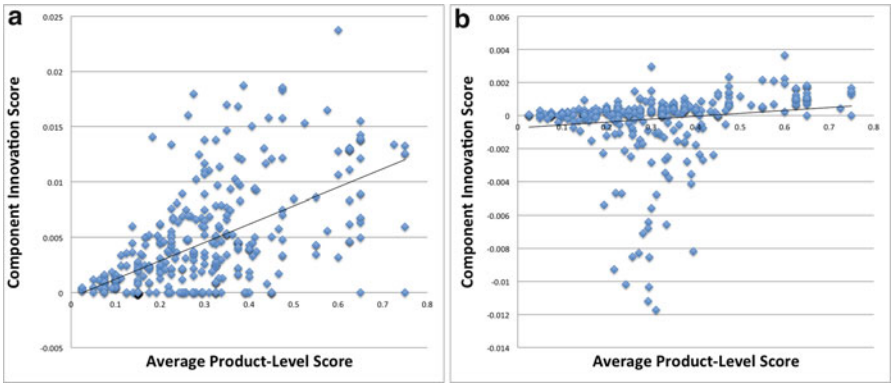


Fig. 4 Agent-estimated values using survey scores with (**a**) D^{least} and (**b**) $D^{average}$ reward

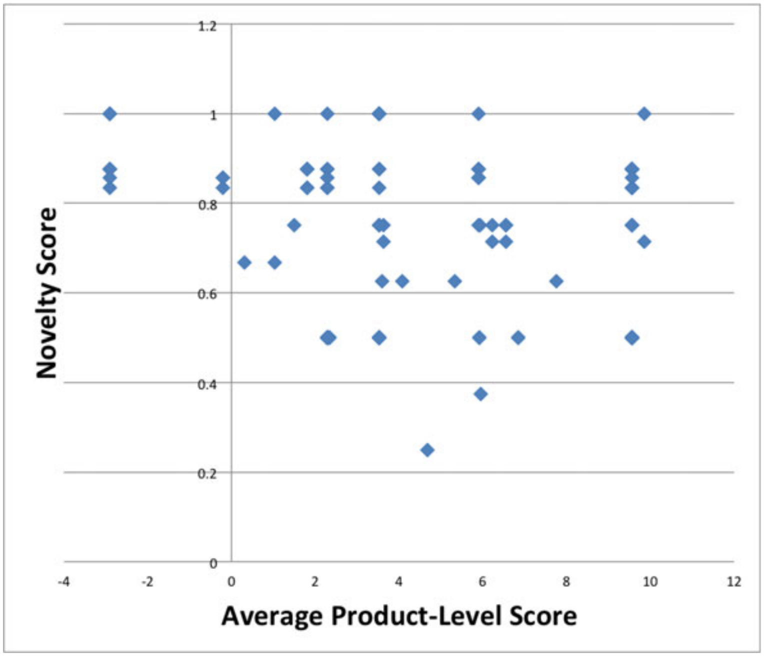


Fig. 5 Novelty scores for function-component pairs in the latent variable dataset

As shown in Fig. 5, the novelty within this dataset has the most pronounced trend in having higher novelties at the extrema and lower novelty scores toward the center. This data was almost triangular in shape, and further supports the trend across all datasets that novelty correlates to extreme high or low average product-level score. This is particularly pronounced because all negative average product-level scores have a relatively high novelty. Figure 5 also shows a certain amount of

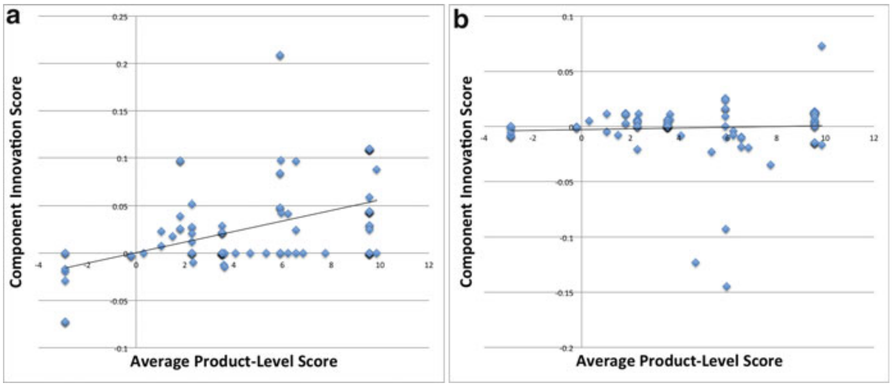


Fig. 6 Agent-estimated values using latent variable scores with (**a**) D^{least} and (**b**) $D^{average}$ reward

discretization which also exists in the expert data. The data are not striated so much as simply sparse, and therefore these discretizations are likely less due to the scoring mechanism and more due to the fact that only eight products are included in the analysis.

Figure 6a shows that as the average product-level score increases, the innovation score assigned to the components tends to increase as well. There are two outliers in this data located at (5.9, 0.209) which represent the function-component pairs {channel energy, em sensor} and {channel energy, hydraulic pump}. These data are somewhat different from the data found in Fig. 6a in that the innovation score peaks prematurely in relation to the product-level score. The tendency for the data to spread out as product-level score increases, but this trend is not pronounced.

The results shown in Fig. 6b trends toward a zero average. This data also shows a trend which has been observed in the other results, which is that it tends to spread out as the average product-level innovation increases. The data in Fig. 6b also show several outliers which are worth mentioning. The point with the highest estimated innovation score is located on Fig. 6b at (9.86, 0.073) and corresponds to {signal signal, circuit board}. What this is saying is not that products within this dataset tend to be more innovative if they *have* a circuit board, but that they tend to be more innovative if they *transmit a signal* using a circuit board.

So what do negative outliers mean? There are a group of three negative outliers obvious on Fig. 6b. Interestingly, two of them also involve a circuit board, but it is used in a different manner. The points at (5.94, −0.145) and (5.91, −0.093) correspond to {channel energy, circuit board} and {channel signal, circuit board} respectively. The fact that these have a different effect not just based on the component involved but also how it is used suggests that innovation may come from a creative application of a component rather than necessarily identifying a component as creative. Channeling energy and signals are an integral function of a circuit board, and therefore will appear wherever there is a circuit board in the design set. They perform a functional role but do not contribute to the innovation of a device. A circuit board that transmits a signal indicates that some kind of display was found on the product analyzed. A designer attempting to leverage this suggestion might attempt to incorporate a graphical display on a new product design.

Analysis of the Results

Though there are differences in the source of the innovation scores as well as the method by which they are gathered, there are trends across all datasets which reflect some of our key findings in this work. We first identify these trends, and subsequently demonstrate how to use this information to perform a innovative redesign of a well-known product.

Trends in Innovation Scores for Components

Results across all datasets show a dip in novelty scores as well as a spreading out of the innovation scores as the product-level score increases. Novelty is too dispersed throughout the dataset to draw definite conclusions, but it appears that higher novelties in all cases tend to appear at the extreme values of average product-level score.

The innovation scores for function-component pairs obtained using D^{least} across all datasets showed an upward trend, generally learning that components in scores with a higher average product-level score had more innovation value than the those found in products with lower scores. Conversely, the scores found by $D^{average}$ do not necessarily show an upward trend with the increase in average innovation score, but instead show the same dispersal of the data. This indicates that the difference reward is able to identify what, in products with high scores, is both contributing and not contributing to those high scores.

Outliers in this data can give us some insight into components that are particularly influential in promoting innovation. Through an analysis of the outliers in the Latent Variable dataset using the $D^{average}$ scoring, we were able to identify that the presence of a component does not necessarily correlate with a better novelty score – how it is *used* contributes in large part to this score as well. Looking at the outliers, the results may have been influenced in part by the audience analyzing the dataset. We found that in the expert evaluation dataset we had two outliers which were the pairs {channel material, reservoir} and {control magnitude material, screw}. These do not seem like particularly innovative components, but the innovation metrics in this case were derived from consumer magazines, which likely had a level of functionality and market influence. The function-component pairs found to be particularly innovative also are highly *useful* in a final design, and this usefulness may have played a part in their identification as 'innovative'.

The outlier identified in the survey data using $D^{average}$ {convert energy, handle} was also underwhelming. It was identified as having the highest novelty score, but this may say more about the population surveyed than its true innovation contribution. Again usefulness comes into play here; ‘innovation’ and ‘usefulness’ are hard to distinguish from one another in many cases, and the survey-takers were not design engineering students. When they were shown the designs, it was as a picture with a brief explanation, and not a full breakdown of how the product works. A handle is an easily-seen component which would show up in a picture, and would add to the usefulness of a design.

The latent variable data outliers, which *were* gathered from design engineering students, show more interesting function-component pairs identified as innovative. These were the pairs {channel energy, em sensor} and {channel energy, hydraulic pump}. These identify components that have a higher technological complexity than the outliers of the other surveys, and therefore are potentially more interesting to design engineering students. It is likely that the design engineering students were able to better understand the functionality of a product, and therefore find it more innovative than someone who had only a functional or aesthetic knowledge of a product.

The influence of demographics on training data may also be a parameter that can be leveraged. Innovation is a subjective measure, and companies cannot design a product with all demographics in mind. Products need a market. If the demographics of this market can be used to bias the training data for our multiagent system, we may obtain better evaluations of component innovation may be obtained for the target market.

Implication for Generating New Designs

Though we can demonstrate patterns in the data using our techniques, an example in an actual design application provides a more intuitive look at what the different techniques actually offer. For this reason, we present a redesign of the product which had the lowest innovation ranking according to our survey: the Dustbuster. We selected five functions in the Dustbuster with the lowest product-level average score ratings and calculated suggested replacements for the components according to the highest-ranked component solutions as discovered by our different techniques on the survey dataset. The results are shown in Table 1, which can be compared with the original design given in the ‘Repository’ column.

The assessment of the innovation of the different replacements for the design is difficult to perform objectively. Additionally, as practicality is separated from the innovation, not all suggestions are necessarily optimal or even possible to implement. Nonetheless, all suggestions offer a designer a different perspective on how to modify a vacuum. The novelty evaluation suggests using a hydraulic pump, which is rarely if ever present in vacuums and may offer an interesting perspective on how to creatively channel material sucked into the vacuum. The D^{least} evaluation suggests that a speaker replace the electric switch to turn on the vacuum, which indicates a redesign featuring a voice-activated vacuum cleaner. The $D^{average}$ evaluation suggests that guiders may be used to replace the electric switch,

Table 1 Dustbuster original design components and redesign suggestions

Function	Repository	Novelty	D^{least}	$D^{average}$
Convert energy to signal	Electric switch	Light source	Speaker	Guiders
Control magnitude energy	Electric cord	Washer	Abrasive	Coupler
Channel energy	Electric cord	Abrasive	Pulley	Coupler
Channel material	Guiders	Hydraulic pump	Friction enhancer	Shaft
Convert energy	Electric motor	Cover	Magnitude controller	Handle

suggesting that the vacuum might be designed to mechanically show when it is full. The material coming into the vacuum might brush past the guiders and put pressure on them which would activate another part of the design to detect fullness of the vacuum cleaner.

Though the two approaches have somewhat different results in their evaluation of innovativeness, they are equally valid approaches. The counterfactual term used in the difference reward for D^{least} will provide consistently positive evaluations for the innovativeness of devices, whereas $D^{average}$ provides approximately half negative evaluations. This may represent two possible ways of looking at the data; either the presence of a standard component does not help the innovation score, or it may be perceived to *bring down* the innovation score because most of the other options might contribute more meaningfully to the product's innovativeness.

Ultimately the usefulness of the design suggestions is still in the hands of the engineer, but the suggestions based on D^{least} and $D^{average}$ appear to offer more interesting solutions overall to the given functions. They do not appear to be *practical*, necessarily, as many of the suggestions do not present feasible design decisions. This draws attention to the fact that (1) we are using the most abstract language in the design repository, which may mean that components designed to handle certain specific tasks may not be applicable for all functionalities and (2) we need some sort of method of identifying which components are *possible* to perform these functions. This makes MEMIC [2] the ideal complement to this assessment technique, as its target is to identify component suggestions from designs which have had similar functional flows, and therefore this increases the chance that components may have similar functionality.

Conclusions

We have developed a method for using difference rewards under a multiagent framework to propagate product-level innovation scores down to the component level. By framing the design process as a multiagent component-selection process with functional requirements modeled as agents, we are able to have our agents learn scores for component solutions after seeing several example designs. From the data gathered in this work, we can draw three conclusions; (i) we identify trends in the novelty relating to our average product-level innovation score, validating our approach; (ii) we can use this method to identify components that both add to and detract from the innovation score of the device; (iii) we can use our evaluations to perform design alterations and give an indication of which components have historically increased innovation for a particular function.

Our first conclusion is validated by the tendency for the novelty scores to polarize, and typically decrease for mid-level average product-level scores. Novelty *does* relate to the innovation score of the device, but it is not directly proportional to innovation score. The trend toward a dip in novelty scores also uncovers a major shortcoming in our attempt to identify innovative components; the more common components may still receive a mid-level innovation score, but this depends on the *configuration* of components rather than the components themselves. We also demonstrate that in the high levels of average product-level innovation scores, more novel components are more frequently seen. We also see novel components in the lower levels of innovation – this is most likely due to devices which perform only one task uniquely.

Our second conclusion best highlights the intent of developing our difference methods for innovation identification: in innovative products, there tend to be components that add more to the innovation of the device than other components. This comes with the parallel observation that some components actually *detract* from the innovation of a product at higher product-level innovation. The fact that we can identify components within innovative products that have a large positive or negative effect on the innovation of a design suggests two interesting findings: one, that the innovation of a product may be carried primarily by only a couple of components within the product, and two, that highly innovative products must rely on tried-and-true methods of performing functions in order to have a fully operational product. Because there are so many components that have negative scores under our D^{least} evaluation in the higher levels of average product-level score, this indicates that perhaps the most innovative products do *one* thing innovatively, and they perform other functions in a traditional and therefore reliable manner. This is consistent with the previous research done using functional subtraction [1], which identified functions existing in innovative components which did not exist in the common components.

Our third conclusion was validated by our redesign of the Dustbuster. By using the data output by the multiagent system, we were able to demonstrate the fact that the learned data provides an accessible way of obtaining design suggestions and their associated innovation scores. We also identified the fact that bias in our datasets plays a large role in what components will be considered the most innovative; both the products and the people evaluating them were different across the datasets, which introduced bias not only from the backgrounds of people, but by the different novelty scores for components in a different dataset.

There is currently no standard for how close our innovation calculations are to the true innovativeness of the components. The *usefulness* of our measurements may be measured in the same way that the usefulness of concept generation techniques have been measured and validated. We have developed a tool which is intended to inspire creativity in engineers. Though this technique is theoretically sound from a multiagent perspective, the actual usefulness in introducing innovation into the design process must be decided by the engineers who use it.

Acknowledgments The authors would like to thank the National Science Foundation for their support under Grant Nos. CMMI-0928076 and 0927745 from the Engineering Design and Innovation Program and the NSF REU supplement grant CMMI-1033407. Any opinions or findings of this work are the responsibility of the authors, and do not necessarily reflect the views of the sponsors or collaborators.

References

1. Oman S, Gilchrist B, Rebhuhn C, Tumer IY, Nix A, Stone R (2012) Towards a repository of innovative products to enhance engineering creativity education. In: ASME 2012 International design engineering technical conferences and computers and information in engineering conference American Society of Mechanical Engineers, pp 207–218
2. Arnold C, Stone R, McAdams D (2008) MEMIC: an interactive morphological matrix tool for automated concept generation, In: Proceedings of the 2008 industrial engineering conference. Vancouver, BC
3. Shah JJ, Smith SM, Vargas-Hernandez N (2003) Metrics for measuring ideation effectiveness. Des Stud 24(2):111–134
4. Bohm MR, Vucovich JP, Stone RB (2008) Using a design repository to drive concept generation. J Comput Inf Sci Eng 8(1):014502
5. Hirtz J, Stone RB, McAdams DA, Szykman S, Wood KL (2002) A functional basis for engineering design: reconciling and evolving previous efforts. Res Eng Des, Springer 13 (2):65–82
6. Brown DC (2008) Guiding computational design creativity research. In: Proceedings of the international workshop on studying design creativity, University of Provence, Aix-en-Provence
7. Agogino A, Tumer K (2008) Analyzing and visualizing multiagent rewards in dynamic and stochastic domains. Auton Agent Multi-Agent Syst 17(2):320–338
8. Barto AG, Dietterich TG (2004) Reinforcement learning and its relationship to supervised learning. In: Handbook of learning and approximate dynamic programming. Wiley, Hoboken
9. Claus C, Boutilier C (1998) The dynamics of reinforcement learning in cooperative multiagent systems. In: AAAI/IAAI, pp 746–752
10. Agogino AK, Tumer K (2012) A multiagent approach to managing air traffic flow. Auton Agent Multi-Agent Syst 24(1):1–25
11. Wolpert DH, Tumer K (2002) Collective intelligence, data routing and Braess' paradox. J Artif Intell Res 16:359–387

A Process Model for Crowdsourcing Design: A Case Study in Citizen Science

Kazjon Grace, Mary Lou Maher, Jennifer Preece, Tom Yeh, Abigale Stangle, and Carol Boston

Abstract Crowdsourcing design has been applied in various areas of graphic design, software design, and product design. This paper draws on those experiences and research in diversity, creativity and motivation to present a process model for crowdsourcing experience design. Crowdsourcing experience design for volunteer online communities serves two purposes: to increase the motivation of participants by making them stakeholders in the success of the project, and to increase the creativity of the design by increasing the diversity of expertise beyond experts in experience design. Our process model for crowdsourcing design extends the meta-design architecture, where for online communities is designed to be iteratively re-designed by its users. We describe how our model has been deployed and adapted to a citizen science project where nature preserve visitors can participate in the design of a system called NatureNet. The major contribution of this paper is a model for crowdsourcing experience design and a case study of how we have deployed it for the design and development of NatureNet.

Introduction

Crowdsourcing is a way of increasing diversity and creativity. Individuals that participate in crowdsourcing endeavors typically lack formal credentials in the domain of interest, yet bring diversity of knowledge and expertise to projects and challenges. Crowdsourcing creates new challenges for the design of effective social structures and interactive technologies [1]. This research is motivated by the principle that the diversity of crowds can be of benefit in distributed creative design tasks. According to Page [2], a major challenge in effective crowdsourcing is to

K. Grace (✉) • M.L. Maher
University of North Carolina at Charlotte, Charlotte, NC, USA
e-mail: K.Grace@uncc.edu

J. Preece • C. Boston
University of Maryland, College Park, MD, USA

T. Yeh • A. Stangle
University of Colorado, Boulder, CO, USA

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_14

structure a conjunctive task (for example, design) into a set of disjunctive tasks (for example, ideation and evaluation) so that individuals can provide discrete contributions. This is particularly problematic because creative design tasks are “wicked” problems [3] and thus often not easily decomposable. While some crowdsourcing approaches (e.g.,: Innocentive.com, Threadless.com) have been successful with highly motivated volunteers performing well-defined, discretized design tasks, we lack an understanding of how to distribute dynamic and irreducible tasks to a crowd. In this paper we describe a process model for crowdsourced experience design and its implementation for use by a citizen science community.

The principle of the wisdom of the crowds states that the result of asking a large number of people to respond to a task or challenge is better than the result of asking an individual or group of experts. Hong and Page [2, 4] show that diversity, characterized by the difference in the perspectives and heuristics of each individual in the group, leads to better solutions that any individual could find. Diversity trumps formally-recognized ability in a computational model of group decision making, and a diverse group of agents outperformed a group of high ability agents at problem solving.

Crowd work is the completion of tasks through crowdsourcing platforms (usually large, easily-decomposable tasks such as data cleaning and transcription). Ethical problems arise when requesters pay very small amounts of money to obtain output from a vast workforce, or when competitive work is posted for which only a single winner will be paid [5]. To address this criticism an emerging field of research focuses on collaborative, motivating, and creative crowd work [6]. Creative crowd work will require new models of collaborating to be successful [7], making it of interest to the design research community. One design process for which significant crowd work research exists is online critique, with studies finding that that scaffolding critique with a set of design principles elicited more structured feedback [8], and that users preferred such feedback [9].

Fischer and Giaccardi’s meta-design architecture [10] describes systems that are re-designable by their users. Meta-design produces systems are “underdesigned” [11] – they contain only the assumptions and restrictions that are necessary to let the end users take over and refine them to their own needs. Meta-designed systems begin with a “seed” design and then progress through an “evolutionary growth” process, occasionally requiring a significant “reseeding” process. We propose a process model, Crowdsourced Experience Design (CSED) that describes the iterative crowd-inclusive design process that occurs during the evolutionary growth phase of a meta-designed crowdsourcing platform.

Crowdsourcing Design

Crowdsourced experience design (CSED) is based on the principle that while the highly motivated users of a system are not experts in design processes, they are experienced in the use of the system. CSED aims to leverage that existing motivation through scaffolding user interactions with meta-design, allowing non-expert designers to constructively contribute to a distributed design crowd. The system will benefit from iterative, collaborative user-driven improvement, and the users will benefit from the sense of ownership and empowerment derived from being granted collective creative control over their community platform.

Using crowdsourced design (CD) as an approach to increase participation is informed by a rich background of research on why people participate in various types of online communities: open source software [12–14], crowdsourcing applications [15–19] and open innovation communities [20–22]. These studies recognize that different user types can have vastly different reasons for participating [21, 23–26] and they may change over time [27]. By focusing on design, we appeal to three of the major reasons that individuals claim to participate in online communities: fun, challenge, and social [28]. We hypothesize that enabling users to actively and collectively design the platform used by their community will increase their motivation for and contributions to that community. Our process model for crowdsourcing design is informed by three existing design models: participatory design, co-design, and meta-design.

Participatory design (PD) [29] is a set of theories and models that relate to the participation of users and key stakeholders in the design process. Central to PD is the notion that the users' rights, knowledge and voice be not just heard by designers but expressed directly in the design. The underlying philosophy of participatory approaches is the democratization of design, a principle that has carried through to later extensions of the field, including meta-design and CD. Distributed participatory design [30] has been proposed as a way to extend the participatory design method to crowdsourcing and online communities, an approach adopted in our CD models. In CD, unlike most PD, the users do not interact with the designers through scheduled and structured activities, instead contributing to the design by volunteering ideas and critique online.

Co-design, also referred to as co-creation [31, 32], is a participatory design approach that focuses on bringing users into the earliest phases of design: the ideation or "pre-design" processes. Co-design's focus is on breaking down the barriers of objectivity between researchers and their subjects, and instead treating user research as an embedded, integrative practice. Crowdsourcing design extends this practice by placing ideation – by both stakeholders and designers – in the context of an online platform, where design decisions are not only user-transparent but also user-driven.

Meta-design [33, 10] is a conceptual framework for distributed participatory design that describes the structure of socio-technical environments designed to be designed. Meta-design is a response to the situatedness and embeddedness of computer systems in modern society, factors that traditional prescriptive, empirical user-centered design models do not account for. The designers of a system become its "metadesigners", producing a set of tools that allow users to take ownership of system components and iteratively improve them. Meta-design replaces the traditionally binary distinction between consumers and designers with a spectrum of roles, from "passive consumer" through "power user" to "meta-designer". Within this framework we develop CSED, a process model that extends metadesign to describe how users collectively contribute to designing.

Through this framework, professionals and non-professionals assume roles that characterize their contribution to the design process. This occurs whether they are members of a team as in collaborative design, members of the user community who are called upon by a design team to participate in the process as in participatory design, or a group of people who form a design community to solve a problem with their own resources. In typical collaborative design, roles are ascribed to individuals according to their expertise. Participatory design involves stakeholders in the design process whose roles are determined by the design team. Crowdsourcing design actively involves an open community in design, basing the design in part or in whole on their design contributions. Crowdsourced design platforms, such as Quirky.com, draw ideas for design problems and ideas for alternative solutions from the crowd and encourage people to choose and change their roles as they participate over a period of time.

Involving volunteers in the design of a crowdsourcing system applies the same principle to the design of the system as to the crowdsourced tasks already extant within that system. Designing experiences that are congruent with the interests and capabilities of a diverse population of participants is critical to successful crowdsourcing initiatives. Involving a large population of people in user experience design brings diversity of knowledge and expertise as well as more knowledge of the interests, motivations, and capabilities within the population. We hypothesize that CSED will result in a more creative collection of tasks that have a better fit with the motivations, interests, and capabilities of the crowd. Crowdsourcing design can facilitate inclusiveness by motivating a broader community to participate in design thinking.

Experience design has been conceptualized as a holistic combination of compositional structure, sensory engagement, emotional response and spatio-temporal context that together allow us to make sense of experiences [34]. While many authors have positioned experience design as being "more than delivering services" [35–37], widely-adopted design methods have yet to emerge. In order to render this complex and still-evolving discipline accessible to non-expert users, we adopt a framework for experience design based on the functions the system can fulfill.

A study of communication in the Quirky online design community (quirky.com) shows how the crowd contributes to ideation and evaluation as part of a larger design process [38]. In this study, protocol data was extracted from the discussion threads from three separate product designs, and each discussion thread was segmented and coded to characterize the CD process. The analysis of the coded protocol shows that a design process that includes crowdsourcing is similar to the ideation and evaluation processes present in individual and team design, and also includes a significant amount of social communication. Our model of crowdsourced experience design adapts the crowdsourced design model used in Quirky in the way the crowd and the design team contribute to, evaluate, select, and implement design ideas. Our process model differs from the model used in Quirky in its application: Quirky has predesigned the experience for the crowd to participate in the design of new products and our model allows the crowd to contribute to the design of the crowdsourcing experience. In order to achieve this, we adopt an iterative approach to designing versions of our crowdsourcing platform and we establish three phases in the development of a meta-design architecture for engaging the crowd in design.

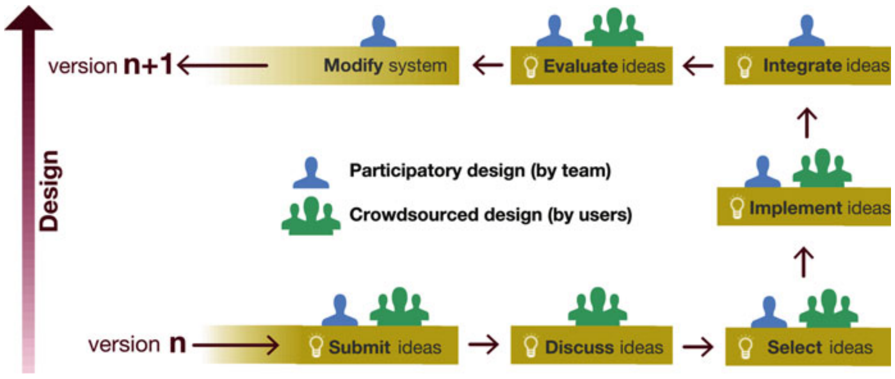


Fig. 1 The crowdsourced experience design process model

A Process Model of Crowdsourced Experience Design

CSED is a new design process model we propose for user-driven contribution to a meta-designed system. It is carried out as a collaborative effort between crowd participants (stakeholders) and a dedicated design team (experts). A design team must first create the initial "seed" (i.e., version 0). CSED enables the design to evolve iteratively in a series of cycles. Each cycle consists of seven processes: ideas submission, ideas discussion, ideas selection, ideas implementation, ideas integration, ideas evaluation, and system modification. It is through this cycle that design ideas submitted (or contributed) by individuals in the crowd lead to concrete system modifications by the design team. Figure 1 illustrates how a design evolves from version *n* to version *n* + *1* through a seven-process CSED cycle. The details of the seven processes are explained as follows:

- *Ideas submission*: the process by which user ideas for system features are entered in to the meta-design system. Individual users can submit new design ideas from any component of the system, such as making suggestions about a new activity they could perform with the system.
- *Ideas discussion:* the process by which an idea is commented on, refined and discussed by the crowd. Contributing a new idea starts a discussion thread and anyone can comment or vote on the idea, (e.g., via a "like" button), which provides crowd feedback on individual ideas.
- *Ideas selection*: the process by which an idea is nominated for inclusion into the system. Based on synthesis of crowdsourced comments and votes, strong ideas are selected by the design team (based on criteria that are visible to the crowd) and integrated into the next version.
- *Ideas implementation:* the process by which an idea is translated into software. The selected idea is implemented in the next version. Individuals in the crowd with programming skills can modify the open source software, and/or development can be performed by the project team.

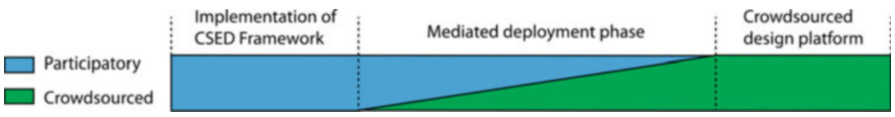

Implementation of CSED Framework
Mediated deployment phase
Crowdsourced design platform
Participatory
Crowdsourced

Fig. 2 Three-phase deployment strategy for crowdsourced experience design

- *Ideas integration*: the process by which a new idea is incorporated into the platform. Integration of new software modules and functionality is performed by the design team.
- *Ideas evaluation*: the process by which an idea, once implemented, is evaluated. New features are integrated into the platform as "prototypes", and users are encouraged to discuss and comment on them.
- *System modification:* the process by which prototype features become final. Prototypes that have been well-adopted by the community become finalized after enough time has passed.

Operationally, a CSED system must provide support for the crowd to participate in these processes. Such support can be mediated in two ways. First, it can be mediated purely through technology, such as a web-based crowdsourcing platform. Second, it can be mediated through face-to-face design sessions between the crowd and the design team. The former is crowdsourced design and the latter is participatory design. In a real deployment scenario, it is difficult to start crowdsourced design right away because a functional crowdsourcing platform does not yet exist. We have deployed a phased approach in which CSED would initially lean more heavily on participatory design and then gradually transition to crowdsourced design as the CSED platform becomes more functional. This strategy has three phases, shown in Fig. 2:

1. Implementation of the CSED framework: the initial phase where traditional participatory design approach is used to design and develop support for CSED activities such as ideas submission and discussion.
2. Mediated deployment: the intermediate phase where more stakeholders are introduced to and encouraged to participate in the meta-design process through the CSED platform. Participatory design sessions are still needed but less and less frequently. In this phase the selection of new design ideas is negotiated between the design team and stakeholders.
3. Crowdsourced design platform: the final phase where the CSED platform is sufficiently stable to support crowdsourced design. Once a CSED project reaches this phase the selection process for incorporating new ideas becomes fully crowd-driven. The design team may retain a curatorial role, but the criteria for choosing proposed design changes are determined entirely by the crowd.

Case Study: Crowdsourced Experience Design of NatureNet

NatureNet is an ongoing research project that implements the CSED framework in the domain of citizen science. Citizen science is a type of crowdsourcing in which individuals participate in scientific endeavors related to their personal interests rather than based on their formal professional credentials. Typically, citizen scientists begin participating only after the incentive mechanisms, the data collection model, the crowdsourced tasks, and the interaction design are finalized. The crowd's involvement is thus typically limited to a set of data collection tasks specified by the designers of the interactive system. Conversely, NatureNet adopts a "community field lab" approach, in which hypotheses and the experimental design to support them are developed both top-down by scientists who suggest data collection tasks and bottom-up by the community of educators, naturalists, students, and members of the public who identify topics of interest and set about collecting data. This approach, motivated by the desire to increase ecological literacy as well as perform field research, is particularly suited to crowdsourced experience design.

A major design principle for NatureNet is to embed the citizen science technology and collaboration in the natural environment, more specifically, to place the technology in a nature preserve. Our choice of embedded and immersive technology is a tabletop system that encourages people to interact with maps, data, comments, and each other by leaning over the table together [39]. We chose a tabletop platform to encourage participation by multiple simultaneous users, based on Kim and Maher's study showing that a tabletop environment results in larger, more immersive body movements [40]. Creating a more immersive experience leads to users' feeling like they are part of the action rather than passively watching others, consistent with Laurel's "Computers as Theater" metaphor [41].

NatureNet uses an incremental design process in which each stage of the design process provides increasing functionality, starting with an initial interaction design, an overall system model, and an initial data model. This approach to crowdsourcing design differs from other models for crowdsourcing design such as quirky.com and topcoder.com by seeding the community with an initial design that will encourage participants to contribute to a system that has limited functionality in order to co-create a design that satisfies their needs and desires, a principle described by [42] for encouraging initial participation. As participants shape the tasks and the citizen science environment, they will become stakeholders in the success of the citizen science project and the design of the technology to support their tasks. Hence crowdsourcing task and interaction design should result in greater participation in the resulting citizen science initiative, not just from the individuals who participated in the crowdsourcing design activities – but also from the population as a whole.

In traditional scientific data collection, scientists collect field notes of their observations, which are often attributed to a specific location and documented through sketches, photographs, descriptions, or other media. Field notes are often stored in a book or extracted to contribute to collections of data that represents a specific finding. The interaction design in NatureNet allows observation notes to be added to photographs and collections. In addition, NatureNet allows users to add design ideas to each functional component in the citizen science platform by clicking on an attached “Design Ideas” button to activate the Tabletop Designer Client for that class of component. This allows a user with an idea about how to redesign the interaction for a component to immediately contribute their suggestion.

While the design for NatureNet diverged significantly from its preliminary idea (as is appropriate for meta-design!) the initial design concept supported the following user tasks, to be deployed in three phases:

- Stage 1. Map your walk: use an app on a mobile device to take photos and observations while walking in the nature preserve. Media and notes are automatically transferred to a tabletop computer at the entrance to the preserve and located on a shared map of the preserve. Photos are shown as collections organized by user or by location. The social networking part of the system allows people to create a profile and to comment on their images, their walk, and others’ data and experiences.
- Stage 2. Know your nature preserve: in this version, increasing functionality will include a history of citizen science efforts in the nature preserve. For example, a map of the nature preserve will include links to data about previous visits to the nature preserve. As more data is collected, various layers of data can be turned on or off an overlay on the map of the nature preserve showing previous visits for an individual, all visits, data about a specific plant or animal species. Increasing functionality for the social networking part of the system will allow people to add comments about visits and species and give points to those that contribute the most or the best photos.
- Stage 3. Who is in the nature preserve: in this version, increasing functionality will include real time data about species sightings and consenting participants, overlaid on the map of the nature preserve.

At each stage the design team oversees and manages the increasing functionality of the design; either based on a participatory design approach in a control case or based on the crowdsourced design ideas in an experimental case. This fits with the cycles of “reseeding” proposed in [10].

System Architecture

CSED systems present an unusual software design problem: the system must be designed so as to permit the content, features and interactions to be continuously redesigned by its users. With the rapid turnaround between iterations of the interface expected in the crowdsourced interaction design approach, the system is highly vulnerable to the “Law of Increasing Complexity” [43]. As the system evolves it will inevitably become more complex and thus require additional effort

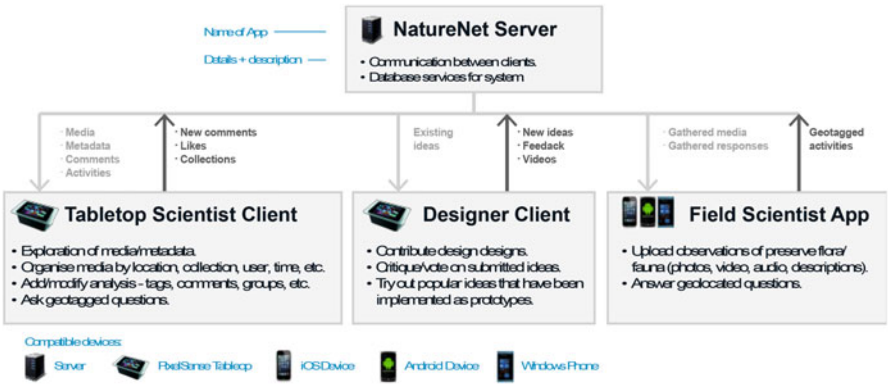

Name of App
Details + description
NatureNet Server
• Communication between clients.
• Database services for system
Media
Metadata
Comments
Activities
New comments
Likes
Collections
Existing ideas
New ideas
Feedack
Videos
Gathered media
Gathered responses
Geotagged activities
Tabletop Scientist Client
• Exploration of media/metadata.
• Organise media by location, collection, user, time, etc.
• Add/modify analysis - tags, comments, groups, etc.
• Ask geotagged questions.
Designer Client
• Contribute design designs.
• Critique/vote on submitted ideas.
• Try out popular ideas that have been implemented as prototypes.
Field Scientist App
• Upload observations of preserve flora/fauna (photos, video, audio, descriptions).
• Answer geolocated questions.
Compatible devices:
Server
PixelSense Tabletop
iOS Device
Android Device
Windows Phone

Fig. 3 The four components of the NatureNet system architecture and their roles

to maintain and upgrade. The system architecture has been designed to mitigate this and other pitfalls of rapidly iterating, distributed and participatory development.

NatureNet leverages ideas from the Open Source movement, in which rapid iteration and discussions of design ideas are common. The online communities that spring up around open source software projects create, discuss, vote on and implement design ideas in a distributed environment, similarly to this project, but all this design activity takes place outside the software being designed, and has traditionally been primarily focused on system design, rather than interaction design. By empowering users to iteratively self-design their system, NatureNet's architecture emphasizes the principles of transparency, perpetuity, interoperability and flexibility that are regarded as critical to the Open Source movement [44]. NatureNet is also stored on *git*, a distributed revision control system designed to manage the divergent development of open source projects [45]. The branching structure of revisions that allows many programmers to contribute patches to a project is well suited to a crowdsourced design environment.

The NatureNet project specifies a high-level architecture of system components each with defined roles, seen in Fig. 3. These components are expected to be constantly adapting, with existing features being changed and new ones added as a result of the crowdsourced design process. A client/server architecture is followed, with four components:

- The *NatureNet Server* hosts all of the system's data – biodiversity data, media, comments, user profiles and design ideas. Users do not interact with it directly.
- The *Field Scientist Client* is a mobile application that allows users to gather biodiversity data from within the nature preserve and transmit it to the NatureNet Server.
- The *Lab Scientist Client* allows users to interact with biodiversity data on a tablet, multi-touch tabletop or desktop computer.
- The *Designer Client* allows users to view, submit, discuss and select design ideas about how to interact with the other clients.

The Field Scientist client is available on mobile devices through an Android app, while the Lab Scientist and Design clients are available on a multi-touch tabletop and eventually via the web. The lab scientist and designer clients can be switched between simultaneously within the one application – they are seamlessly integrated components of the user experience that are only distinguished from a system design perspective.

Interaction Model

The emergent crowdsourced experience design of the system is facilitated by the separation of the interaction model into three aspects of the user experience: Observing, Reflecting/Discovering and Suggesting. These aspects describe the high-level purpose of each component of the NatureNet system abstractly enough to be unlikely to change during meta-design. The aspects frame the users' mental models that the system is designed to evoke. These aspects form an organizing framework through which NatureNet can be understood as it is being iteratively re-designed:

- *Observing* involves making phonological observations of nature in a guided or unguided way. This uses the field scientist client.
- *Reflecting/Discovering* involves reviewing observations made by users and comparing or analyzing them to create new scientific knowledge. This uses the lab scientist client.
- *Suggesting* involves contributing and discussing ideas about the nature preserve experience. This uses the designer client.

Deploying CSED for NatureNet

In this section we describe our case study in terms of the three phases shown in Fig. 2. The phase 1 participatory design took place at the University of Maryland with science students. The phase 2 mediated deployment took place at Hallam Lake in Aspen Colorado at the Aspen Center for environmental Studies (ACES) with visitors and naturalists. Phase 3 is planned to continue at ACES in 2014, and will be fully crowdsourced.

Phase 1: Participatory Design

The initial deployment phase of NatureNet took a participatory design approach. The design participants were 20 university students and instructors with diverse interests in nature, design, and technology. The participants were selected for initial feedback because they had already had experiences collaborating with each other on group research projects related to biodiversity and technology and were thus well-prepared for ideation and crowdsourcing design. Users were prompted to explore NatureNet's "seeded" design features (the map, scientist and designer clients) as the tabletop technology was novel to all users.

Two focus groups were conducted. The first round included eight participants; the second included 12 participants. Participants in both groups were given information about the research project and the co-design process prior to their tabletop sessions. Students in the first group took pictures of an outdoor study site using a camera; the pictures were then made available to them for commenting and sharing on the tabletop. Students in the second study explored the photographs from the first study.

While using the multi-touch tabletop, participants in both groups were invited to use think-aloud and discussion techniques to share their reactions as they attempted to contribute observations (Observing), annotate photos and share observations of plant and animal life (Reflecting/Discovering), provide comments on and ideas for the system (Suggesting), and pin photos to specific locations on the map. These tasks align with the "Map your walk" stage of NatureNet interaction design.

Ideas Submission and Management: By Design Team The design team collected notes as the participants made observations and suggestions about the design of NatureNet. The design team's focus was to observe how users reacted to the system, and whether the interface features would support their desire to contribute biodiversity observations, reflections and discovery, and suggestions. No design idea submissions were directly entered into the NatureNet system by participants or the design team during the first CSED phase, and all design was participatory, not crowdsourced.

Ideas Comments/Votes: By Participants The participants in study one shared many visual design and interaction design ideas, including: improving the ability to manipulate individual photos on the tabletop, the ability to merge their photos, look at them together, and pick from them to create a special collection of the best ones – in other words, to undertake the truly collaborative task of group photo sharing, and add comments to individual images. In general, participants commented on what constituted too much visual stimulation on the display – too many layered elements. Several participants suggested some sort of grid to which images could be made to snap. Interestingly, the same participants did not find the visual clutter troublesome whilst working with Microsoft's "Bing Image Search" app.

Participants in the second group responded to photos taken by those in the first group and provided feedback on the interaction design. They wanted the system to allow people to arrange collages of their images; allow the best pictures of any given plant or animal to be displayed; encourage people to construct an ecosystem; or show the same location over time or across seasons. Other suggestions involved the interaction design and social features. For example, participants requested multiple on-screen keyboards for simultaneous commenting. They further advised that comments – whether on biodiversity data or on design issues – be threaded for clarity, and that a "Like" button should be added for endorsing contributions. These responses suggest that crowdsourcing design has the potential to motivate participant involvement.

The participants also discussed their overall experience of using the system and provided use scenarios of the system. One student said that they would go to a nature center and take a look at the tabletop before they went on a trail to "see where the good places are"; others said their inclination would be to go on the trail first and then share their photos and observations when they returned – but all agreed that they would somehow have to know what was possible with the system before entering or else they might not capture any biodiversity data or make design suggestions.

Idea Selection: By Design Team The design team selected the ideas from the participants that would lead to better usability rather than increased or different functionality. These ideas included changes in the basic interaction features for collections and the map.

Idea Implementation: By Design Team The design team implemented changes while NatureNet was not in use.

Idea Integration: By Design Team The design team integrated the changes such that the new version of NatureNet looked very different, knowing that the users for Phases 1 and 2 would not significantly overlap.

System Modification: By Design Team The system was immediately modified to reflect the new version as no prototyping feature was available.

Phase 2: Mediated Deployment

Once the design team completed the modifications informed by the participatory design sessions, two rounds of testing were conducted at ACES. ACES receives over 80,000 annual visitors, who typically spend time at Hallam Lake, a 25-acre nature preserve and environmental learning center. Visitors walk nature trails, observe raptors, trout and other wildlife, attend guided nature tours and activities, attend community events, attend day-camp, apprentice in the Field School or work as a Naturalist. ACES was also motivated by the opportunity to participate in the development of a citizen science tool that could complement and enhance the crowds experience of their preserve. The Phase 2 tabletop interface is visible in Fig. 4.

The design team visited ACES twice to engage ACES stakeholders in the design process and to collect crowdsourced input into the system, resulting in two CSED cycles. During both studies, the tabletop was situated in the foyer of ACES, enabling visitors to engage with the content as they entered into the nature preserve. When visitors expressed interest in the tabletop, the design team approached them with the opportunity to use a mobile phone containing the *Field Scientist Client*.

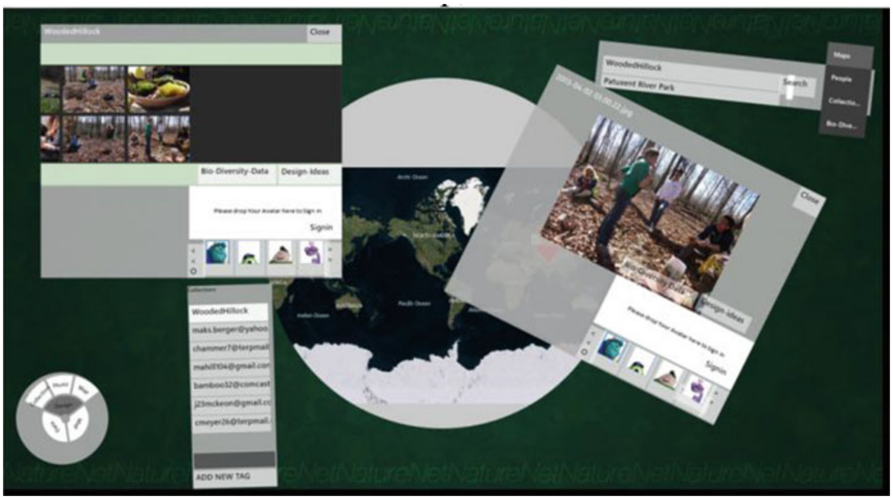


Fig. 4 Original user experience design: design ideas buttons are located within the data collection experience and user icons and profiles are attached to individual contributions

Members of the design team explained the overall intent of the system and research, shadowed visitors as they walked the preserve, took pictures and made notes. Visitors walked around the preserve with the mobile app and were encouraged to take photos and make comments on their observations. They were invited to think aloud and discuss new use and functionality ideas for the technology with other members of the crowd, ACES stakeholders, and the design team. Upon returning to the nature center, users were asked to find their pictures and comments on the tabletop, and read other users' data.

Phase 2 also involved a second visit to ACES several months after the first, during which the design team personally engaged the crowd in contributing observations and ideas into the *Field Scientist* mobile app, and onto the tabletop. In addition, the design team conducted a focus group with ten members of the ACES crowd, including staff, board members, and loyal visitors. Focus group members were issued a mobile phone and were asked to use the app to record their observations, debrief in a group discussion and complete a paper questionnaire. Design ideas generated during this session were manually added into the database. Twenty participants tested the system, including ACES naturalists, visitors, and the design team. Fifteen design ideas and 31 notes were entered into the system, 61 comments about the system were collected via face-to-face discussion, and 7 distinct design ideas were collected via questionnaire.

Ideas Submission and Management: By Design Team, ACES Stakeholders, and Crowd The design team used the *Field Scientist Client* mobile app to engage the ACES stakeholders in the design of the system since *observation* was the first task within the interaction model. Over the course of 3 days, 25 members of the crowd entered 83 field notes into the system via the app. Notes included biodiversity notes and design notes. Approximately one third of the comments were from ACES stakeholders; one third came from the design team (prompted by discussions with ACES stakeholders) and the final third of the participants were visitors. Seventy-five additional comments about the system were manually recorded during in-group discussions around the tabletop and walking the preserve. Some software robustness issues prevented additional comments from being recorded through the interface, and these were captured manually where possible. Whilst the design team was guiding the activities, during this phase of the design process, the ACES stakeholders, the design team, and the ACES visitors participated in design feedback as a collective crowd.

During the second visit to ACES many of the suggestions and comments focused on app features, including the lack of GPS locating, the desire to add questions for the naturalists to answer, and in-app search. Comments also discussed improving the graphics and keyboard, as well as how to use the tabletop as a device for collecting design suggestions.

Face to face discussion reveled that many visitors come to ACES because of the programs and informal learning opportunities ACES offers. This feedback sparked further discussions between the design team and the ACES crowd about how the app might extend the activities already occurring at the Center. Integrating existing ACES relevant activities into the NatureNet system will engage the crowd since they are connected to the original reason they visit ACES. Further, by allowing the crowd to suggest ideas for new activities extends the ability of ACES to support the interests of their visitors without establishing additional programs.

Ideas Comments/Votes: By Design Team, ACES Stakeholders, and Crowd The crowd's comments focused on the visual and interaction design components of the *Field Scientist Client,* the functionality of the tabletop interface, and reasons people visit ACES. Design comments from the crowd involved the desire for a map of the preserve in the app, integration of information about ACES, functionality of the camera feature, and the visual design of the different components. The crowd also asked for the ability to share images between users and platforms, and for increased app stability.

While using the mobile app, ACES stakeholders also provided the design team with insight about the nature preserve and their educational programs and activities. The programs and activities that ACES formally and informally organizes are the incentive for visitors to return repeatedly; in turn their patrons feel part of the ACES community and are drawn to provide feedback about their experience. Feedback revealed that ACES valued the app as a tool for making observations, and as a method to engage people in the center's educational programs.

During the second visit to ACES, the design team and the ACES stakeholders designed a series of activities for the app and the tabletop to solicit further participation in the use of the technology, as well as opportunities for the crowd to make focused suggestions for how to improve the activities and the suggestions. The design team selected specific usability ideas from this phase to produce new versions of the tabletop and mobile app.

Idea Selection: By Design Team After each day of testing the design team reviewed feedback from the crowd and choose features to add to the three clients. The design team took on the role of selecting ideas to implement in order to expedite iteration while the team was physically present. Fourteen of the 75 ideas were selected by the design team, including: a self-guided tour within the app as an incentive to use the mobile device, adjustment of terminology to match ACES's vocabulary (from observation to field note), an introductory page for the app, and ease-of use interface changes.

Idea Implementation: By Design Team The design team followed a 24 h iterative agile development cycle to rapidly integrate users' design suggestions and solicit more feedback about the system while present on site. Amendments to the app included interaction and interface design changes, including an ACES's Self-Guided Tour with a map and interpretive information, image tagging categories, and an improved Field Notes feature. The design team identified a need for enhanced robustness of the data transfer between the app and the tabletop, and a tabletop interface with the user profile options pinned down. The crowdsourced ideas collected about the tabletop were implemented over a 3-month period.

System Modification: By Design Team The crowdsourced experience design approach of letting ideas be discussed among the community for a period of time before integrating them into the system was not followed for this step. Design ideas were integrated into the system as they were implemented. While there was no explicit prototyping feature in the system during Phase 2, users and stakeholders were invited to interact with the system while it was being actively developed, with new versions being released daily. The prototyping period for changes – during which users can comment on and vote to repeal features – will be implemented for Phase 3.

Phase 3: Crowdsourced Design

Phase 3 is a fully crowdsourced design. This phase will start in 2014. Figure 5 shows the current design that will be the starting point for this phase.

Conclusion

We present a model for crowdsourcing experience design that is centered on ideation: each new version of the design is the result of a series of ideation processes achieved collectively by the crowd and mediated by a design team. In order to achieve a fully crowdsourcing design platform for experience design, we deployed the design in stages that transitions from the original seeded design developed by the design team, through a participatory design process, and then the final third stage of fully crowdsourced design. Our case study is a citizen science

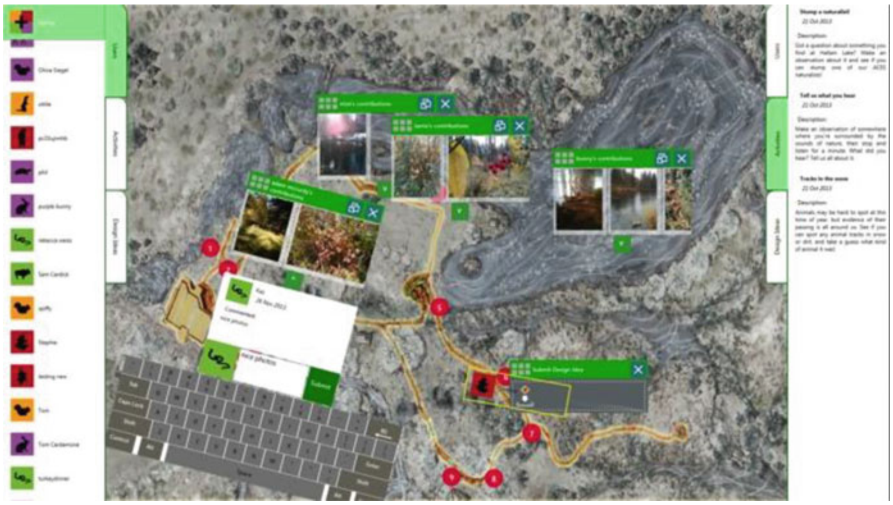


Fig. 5 Current user experience design: a user profile panel is located along the *left* and the activities panel on the *right*. Several collections are open for viewing, and a user is leaving a comment (*on left*) while an avatar is being dragged into a drop zone to enable editing (*on right*)

project, NatureNet: an application of the CSED model that aims to identify creative ways in which individuals can participate in citizen science projects.

In NatureNet, our design process is centered on the concept of ideas to improve the nature preserve experience: individuals in the crowd can contribute, comment, like, select, implement, and integrate their ideas for NatureNet. In this paper we describe how we have deployed NatureNet in a 3-phases: Team design resulted in an early version of NatureNet developed to encourage design ideas. Participatory design is the dominant strategy early in the rollout when attracting participants as stakeholders is most important. Crowdsourced design becomes dominant after the initial “seed” system is compelling and the user base has grown to the point where crowd involvement is self-sustaining. The participatory design studies offer a proof of concept regarding individuals’ interest in participating in the collaborative re-design of a shared citizen science platform, as well as validation of the ideation process model, particularly idea contribution, idea comments, and idea implementation. The mediated design studies have engaged the community that is located within the nature preserve and has lead to a user experience in which the crowd can direct the activities as well as the interaction techniques.

The crowdsourcing approach has been described as “among the most promising” new way of combining the strengths of people and computers [46], and represents an area that design researchers should seek greater engagement with. The model of crowdsourced design of interactive systems presented in this paper is a step towards characterizing the processes by.

NatureNet will continue to be deployed in Aspen Colorado in 2014, providing additional data on design ideas from a more diverse crowd. With this next deployment we can begin to collect data on the third phase of our deployment approach. We will also conduct further qualitative and quantitative assessments of the affect of CSED on the citizen science project, on user motivation, and on the creativity of ideas submitted by users.

References

1. Wiggins A, Crowston K (2011) From conservation to crowdsourcing: a typology of citizen science. In: Syst. Sci. HICSS 2011 44th Hawaii Int. Conf. On. pp 1–10
2. Page SE (2007) Making the difference: applying a logic of diversity. Acad Manag Perspect Arch 21:6–20
3. Rittel H, Webber M (1974) Wicked problems. Man Made Future. pp 272–280
4. Hong L, Page SE (2004) Groups of diverse problem solvers can outperform groups of high-ability problem solvers. Proc Natl Acad Sci U S A 101:16385–16389
5. Silberman M, Irani L, Ross J (2010) Ethics and tactics of professional crowdwork. XRDS Crossroads ACM Mag Stud 17:39–43
6. Kittur A, Nickerson JV, Bernstein M, Gerber E, Shaw A, Zimmerman J, Lease M, Horton J (2013) The future of crowd work. In: Proc. 2013 Conf. Comput. Support. Coop. Work. ACM. pp 1301–1318
7. Morris RR, Dontcheva M, Finkelstein A, Gerber E (2013) Affect and creative performance on crowdsourcing platforms. In: Affect. Comput. Intell. Interact. ACII 2013 Hum. Assoc. Conf. On. IEEE. pp 67–72
8. Wu W, Luther K, Pavel A, Hartmann B, Dow S, Agrawala M (2013) CrowdCritter: strategies for crowdsourcing visual design critique. University of California Press, Berkeley
9. Xu A, Bailey B (2012) What do you think?: a case study of benefit, expectation, and interaction in a large online critique community. In: Proc. ACM 2012 Conf. Comput. Support. Coop. Work. ACM. pp 295–304
10. Fischer G, Giaccardi E (2006) Meta-design: a framework for the future of end-user development. End User Dev. Springer, pp 427–457
11. Brand S (1995) How buildings learn: what happens after they're built. Penguin, New York
12. Lakhani K, Wolf R (2003) Why hackers do what they do: understanding motivation and effort in free/open source software projects. MIT Sloan Working Paper No. 4425-03. Available at SSRN: http://ssrn.com/abstract=443040 or http://dx.doi.org/10.2139/ssrn.443040
13. Hertel G, Niedner S, Herrmann S (2003) Motivation of software developers in open source projects: an internet-based survey of contributors to the Linux kernel. Res Policy 32:1159–1177
14. Nov O (2007) What motivates Wikipedians? Commun ACM 50:60–64
15. Brabham DC (2008) Moving the crowd at iStockphoto: the composition of the crowd and motivations for participation in a crowdsourcing application. First Monday 13
16. Chandler D, Kapelner A (2013) Breaking monotony with meaning: motivation in crowdsourcing markets. J Econ Behav Organ 90:123–133
17. Shaw AD, Horton JJ, Chen DL (2011) Designing incentives for inexpert human raters. In: Proc. ACM 2011 Conf. Comput. Support. Coop. Work. pp 275–284
18. Rogstadius J, Kostakos V, Kittur A, Smus B, Laredo J, Vukovic M (2011) An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets. In: ICWSM
19. Frey K, Haag S, Schneider V (2011) The role of interests, abilities, and motivation in online idea contests. Paper 71. Wirtschaftinformatik Proceedings 2011
20. Wasko M, Faraj S (2000) "It is what one does": why people participate and help others in electronic communities of practice. J Strateg Inf Syst 9:155–173

21. Antikainen M, Väätäjä H (2008) Innovating is fun–motivations to participate in online open innovation communities. In: Proc. First ISPIM Innov. Symp. Singap. Manag. Innov. Connect. World. pp 14–17
22. Merrick K, Niazi M, Shafi K, Gu N Motivation (2011) Cyberworlds and collective design. In: Circuit Bend. Break. Mending Proc. 16th Int. Conf. on CAADRIA, Newcastle, Australia. CAADRIA Association, Hong Kong, pp 697–706
23. Clary EG, Snyder M, Ridge RD, Copeland J, Stukas AA, Haugen J, Miene P (1998) Understanding and assessing the motivations of volunteers: a functional approach. J Pers Soc Psychol 74:1516
24. Lampe C, Wash R, Velasquez A, Ozkaya E (2010) Motivations to participate in online communities. In: Proc. SIGCHI Conf. Hum. Factors Comput. Syst. pp 1927–1936
25. Maher ML, Paulini M, Murty P (2011) Scaling up: from individual design to collaborative design to collective design. Des. Comput. Cogn. Springer, pp 581–599
26. Malone T, Laubacher R, Dellarocas C (2009) Harnessing crowds: mapping the genome of collective intelligence. MIT Sloan Research Paper No. 4732-09. Available at SSRN: http://ssrn.com/abstract=1381502 or http://dx.doi.org/10.2139/ssrn.1381502
27. Bryant SL, Forte A, Bruckman A (2005) Becoming Wikipedian: transformation of participation in a collaborative online encyclopedia. In: Proc. 2005 Int. ACM SIGGROUP Conf. Support. Group Work. pp 1–10
28. Paulini M, Maher M, Murty P (2014) Motivating participation in online innovation communities. Int J Web Based Communities Accept Publ 10:94–114
29. Muller MJ, Kuhn S (1993) Participatory design. Commun ACM 36:24–28
30. Walsh G (2011) Distributed participatory design. CHI11 Ext Abstr Hum Factors Comput Syst 46:1061–1064
31. Prahalad CK, Ramaswamy V (2004) Co-creation experiences: the next practice in value creation. J Interact Mark 18:5–14
32. Sanders EB-N, Stappers PJ (2008) Co-creation and the new landscapes of design. Codes Des 4:5–18
33. Fischer G, Scharff E (2000) Meta-design: design for designers. In: Proc. 3rd Conf. Des. Interact. Syst. Process. Pract. Methods Tech. pp 396–405
34. Wright P, McCarthy J, Meekison L (2005) Making sense of experience. Funology. Springer, pp 43–53
35. McLellan H (2000) Experience design. Cyberpsychol Behav 3:59–69
36. Shedroff N (2001) Experience design. New Riders, Indiana
37. Pine B, Gilmore JH (2011) The experience economy. Harvard Business Press, Harvard
38. Paulini M, Murty P, Maher ML (2011) Understanding collective design communication in open innovation communities. J Co Creation Des Arts 9:90–112
39. Kim MJ, Maher ML (2008) The impact of tangible user interfaces on designers' spatial cognition. Hum Comput Interact 23:101–137
40. Kim MJ, Maher ML (2008) The impact of tangible user interfaces on spatial cognition during collaborative design. Des Stud 29:222–253
41. Laurel B (1991) Computers as theatre. Pearson Education Inc., Crawfordsville, USA
42. Fischer G, Giaccardi E, Ye Y, Sutcliffe AG, Mehandjiev N (2004) Meta-design: a manifesto for end-user development. Commun ACM 47:33–37
43. Lehman MM (1980) Programs, life cycles, and laws of software evolution. Proc IEEE 68:1060–1076
44. Casson T, Ryan P (2006) Open standards, open source adoption in the public sector, and their relationship to Microsoft's market dominance. Stand Edge Unifier Divid 87
45. Loeliger J, McCullough M (2012) Version control with Git: powerful tools and techniques for collaborative software development. O'Reilly Media, Cambridge
46. Luther K (2011) Fast, accurate, and brilliant: realizing the potential of crowdsourcing and human computation. In: CHI 2011 Workshop Crowdsourcing Hum. Comput. Vanc. Can

Collective Design: Remixing and Visibility

Jeffrey V. Nickerson

Abstract Creative work can be performed by thousands of people who interact through collective design systems. The designers modify and combine each other's work in a process called remixing. Remixing can catalyze innovation by allowing designers to build on each other's ideas. For those designing collective design systems, decisions to be made about remixing are intertwined with decisions to be made about the visibility of work in the system – that is, the extent to which designers can see each other's work. Visibility can be universally shared, or can be partitioned according to teams or interest groups. Even if all designs are made visible, some designs are more likely to be seen than others, depending on the methods of display. The interactions between remixing and other features of collective design systems are described, and suggestions are made towards improving these systems.

Introduction

Collective design can occur when large numbers of people engage in design activity. It is different in nature from individual and small team design activity: design participants may only know each other through their designs [1, 2]. This sharing of designs is similar to the sharing that takes place when academics publish discoveries or inventors file patents [3]. The sharing is much faster than these more traditional processes, and so the hope is that these collective design systems will accelerate overall innovation. Because the systems provide traces of the innovation processes as they happen they allow a study of innovation at a higher degree of granularity than before: it is possible to see how one design affects another that follows it days later. This in turn may lead to insights into creative processes, which can inspire new features that encourage large-scale innovation. Such features can be implemented within collective design systems, and they can be compared with each other through user testing. Using this approach, it might be possible to discover

J.V. Nickerson (✉)
Stevens Institute of Technology, Hoboken, NJ, USA
e-mail: jnickerson@stevens.edu

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_15

casual mechanisms of innovation. And if this is possible, it might be possible to increase the rate and level of innovation – for individuals, for groups, for society.

Understanding of collective design can be informed by research in crowds, human computation and collective intelligence. Many crowd work systems ask workers to perform creative tasks, and these systems are changing rapidly as new processes are created to organize workers toward common goals [4]. Reviews of both human computation and collective intelligence mention its potential for creativity, and document the systems being developed commercially and in academic labs [5, 6]. Distinct from these previous surveys, this paper characterizes collective design systems with respect to their support for remixing, and the extent to which users can and do see each other's work. These features interact with the ability of users to generate tasks, to follow others, and to make money.

For example, Thingiverse[1] allows users to share and remix designs to be manufactured on 3D printers [7]. Scratch[2] allows users to share and remix programs written in a visual language [8]. Once a program is posted, it can be modified and reposted. While the user bases are different – Scratch appeals to youths, and Thingiverse to adults – the structure of the systems is similar: users generate their own tasks and modify each other's work. There is one crucial difference: on Scratch, users remix one design at a time, whereas on Thingiverse, users can combine multiple designs.

By contrast, other collective design systems are structured as money-making endeavors, and the locus of task generation is not with the designer. Indeed, these systems usually have three types of actors: the requesters, who want something designed, the workers, who create designs, and the platform owners, who profit from the relationship. Requesters post tasks, along with a payment scheme, and workers perform the tasks. The platform owners are paid a commission. Examples of such systems include Amazon Mechanical Turk[3] [9] and Innocentive[4] [10].

Features of Collective Design Systems

Remixing

Remixing is phrase used to describe the practice of taking ideas and modifying or recombining them [11, 12]. The term was originally used to describe a process in music, where a piece may be altered drastically by starting with all its original components, or tracks, and then altering their volume, their key, their tempo [13]. The term now serves as a metaphor for any combinatory design process.

[1] Thingiverse.com

[2] Scratch.mit.edu

[3] Mturk.com

[4] Innocentive.com

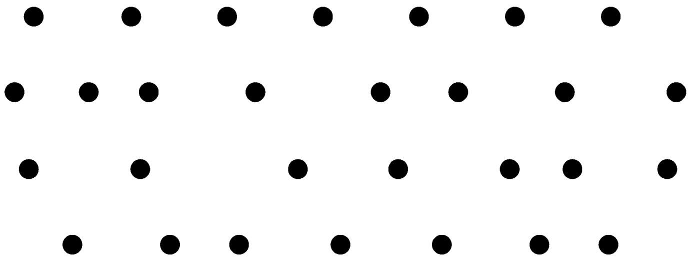


Fig. 1 Ideas unconnected to each other

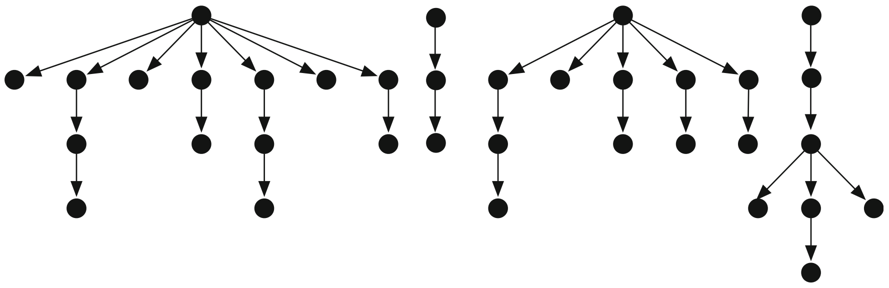


Fig. 2 Remix networks with individual inheritance

Remixing occurs in websites in which people are free to modify each other's work. Wikipedia provides an example in which ideas are constantly edited by others, to the point where a page often cannot be attributed to any one person [14]. It also occurred in a project called Polymath, in which mathematical theorems were proven by many collaborators, ranging from novices to experts [15].

Previous ideas can be considered parents, and new ideas children that inherit characteristics of their parents. Different mechanisms have been used to allow, encourage and keep track of remixing.

But some systems offer no such mechanism. For example, contests such as those provided on Innocentive expect all ideas submitted to be original, and to be blind to other submitted ideas. More generally, there are many systems that ask for contest submission, and such sites often explicitly ask submitters to attest to the originality of the idea. Ideas are generated independently from each other, as shown in Fig. 1. These sorts of systems have been described as greenfield systems because they allow ideas to be generated in the same way a building might be constructed on an empty plot [16].

By contrast, a mechanism that can be described as *single inheritance* provides a way to acknowledge one parent idea, as in Fig. 2. This might be accomplished by a hypertext link, or a textual acknowledgement. On such a system, the ideas can be expressed as tree structures in which each idea has at most one parent, but can have many children. Scratch is an example of this kind of system. Designers can inherit the characteristics of a certain project. An explicit link is created between parent

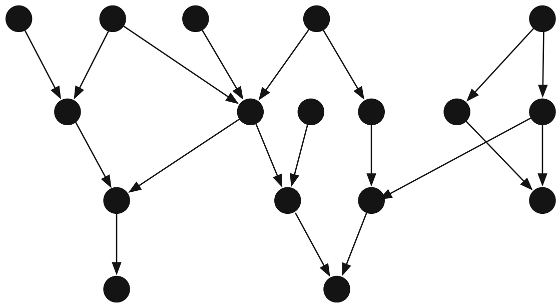


Fig. 3 Multiple inheritance

and child. In addition, a visualization of the remix tree is provided, so designers can find the original source of a chain of remixes, or explore the different branches.

Other systems provide a more complex remix mechanism, *multiple inheritance*, allowing users to derive characteristics from more than one design. This encourages the combination of ideas from other users. It produces a lattice, as shown in Fig. 3, because each design can inherit from multiple parents, and can also have multiple children.

For example, the designer of a chess set on the Thingiverse site might inherit from many different designs to create pawns, bishops, knights, rooks, kings and queens. This sort of remixing is the assembly of components. Blending is also possible: a knight might be created by combining the designs representing horse and human.

In order to further understand remixing, we also need to understand issues related to visibility.

Permitted Visibility

A collective design system can control visibility as well as remixing. For example, a system can prevent one from seeing other competitor's work, but let one see and remix one's own work, or one's team member's work. This might be useful for systems that host contests. A system may decide not to let users see competing teams' work, but might want to provide ways for team members to easily edit and modify each other's work.

With respect to contests, sometimes visibility is universal. For example, everyone can see each other's work in Matlab contests, but the winner is the person who makes the last successful change [17]. This encourages designers to build on what has been created before, but maintains a competitive motivation to seek continued improvement.

These models can coexist. The Climate CoLab[5] at MIT, an online site that encourages people to compete with ideas to address climate change, lets those

[5] Climatecolab.org

creating a contest entry to designate whether the submission is public and editable, or is private [18]. If it is public, everyone can see it: open collaborations then compete against traditional secretive teams.

Visibility, then, can be a universal aspect of the system, or can be localized to parts of the system, to particular groups or projects. Visibility might be controlled not just for reasons related to competition. If everything is visible to everybody, it might have the effect of overloading the system's user. It also might lessen diversity, in that users may focus on particular previous designs and find it difficult to shake the fixation. So, instead, participants might be provided one or two previous designs, and be asked to modify or combine them.

If this process is repeated, a human based genetic algorithm can be built [19, 20]. That is, an initial set of ideas can be created by crowd members who have no visibility of each other's ideas. Then, these ideas can be selected from, and each pair presented to a new user, who will be asked to combine ideas. The new child ideas can in turn become parents of another generation. Such a system was built and tested on the design of chairs [21]. It was also tested on clocks, and the results compared to a simultaneously running greenfield system [16]. In another experiment, users were asked to critique designs, and new users were asked to incorporate critique into a modified design [22].

In these experiments, remixing and visibility were intertwined – users were presented with none, one, or two ideas and told to come up with new ideas. It is also possible to imagine systems in which users are presented with ten ideas and asked pick one or two as starting points for new ideas. Limiting visibility should reduce cognitive load. But limiting visibility may also make modification and combination less satisfying – a single idea to be modified may not be intrinsically appealing to the designer, and a pair of ideas may not be readily combinable. These are the tradeoffs involved.

Natural Visibility

There is another aspect to visibility. In systems in which everyone can see everything, the number of designs completed will eventually eclipse the ability of any one designer to consider them all. The designs will need to be displayed in some way, and the display will have an effect on which designs are remixed, the same way that the ordering of responses to an online search will affect which results are seen. For example, on Scratch, each remix is treated as a separate object, and the remixes are visualized and traversed through a tree visualization. By contrast, on Wikipedia, stories are constantly remixed, and all changes are saved so that previous versions can be reconstituted, but, by default, only the most recent version is seen, as in Fig. 4.

By showing just the latest version, the overall load on the user is lightened. The assumption is that the latest version is the best, that the work has been improved,

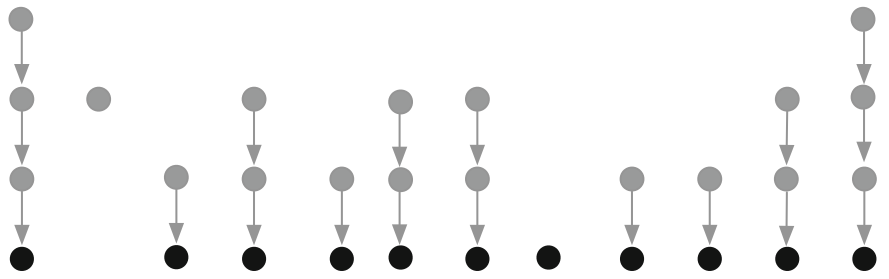


Fig. 4 Natural visibility of Wikis

and so, for the most part, there is little need to go backwards. If there is need, then extra effort is required to trace back through edits to a previous version.

Source code control systems also make this assumption – one may want to go backwards in the branching structure, to undo a mistake, or to understand why a mistake was made, but these searches will be unusual.

This is in contrast to systems such as Scratch and Thingiverse, in which each project is considered autonomous, and all are visible. But in such environments there are thousands of active projects, and it is impractical for a user to inspect them all. The system will use algorithms to order the past work, and this ordering will affect what is actually seen.

In web-based systems, this natural visibility is driven by the projects that appear on the home page of the site, and the pages that are home pages for particular interest groups, and the dynamically created home page tailored for particular users. For example, the home page of Scratch shows featured projects (the result of editorial decisions), featured studios (user-curated lists of projects), projects curated by a particular user, the projects around a particular theme decided by the editors, followed by projects that are currently being actively remixed or actively liked by other members of the community. Participants in the community actively seek to have projects appear on this page [8]. Thingiverse, likewise, shows featured projects, a result of editorial choice, as well as featured collections, projects along a theme, and a listing of projects recently printed on 3D printers. Both sites allow one designer to follow another: it is as if one has subscribed to the designs of the other, in that updates to a design are shown to all followers of the designer.

One possible way to understand natural visibility is through the theory of information foraging [23]. Users will devote time to searching for new ideas, but the longer it takes to find an idea, the lower the probability it will be encountered. Thus, the sorting of projects according to their popularity, or the interests of users, or the social networks of the inventors, will all have an effect on what designers will see. Changing these algorithms may change the evolution of designs in a system.

The evolution will also be affected by who controls the generation of tasks.

Locus of Task Generation

On systems such as Scratch, Thingiverse, and GitHub, designers can decide to contribute to an existing project or create their own. The actors in the system are the designers and the site owners, with the site owners facilitating but not directing work.

By contrast, systems such as Innocentive and Amazon Mechanical Turk work on a different model. A third type of actor, the requester, can sponsor contests or direct work. In the case of Innocentive, the worker enters contests and is compensated if the entry wins. In the case of Amazon Mechanical Turk, the worker is paid up front.

Both these sites are commercial, and both limit visibility of work products. Both stipulate that the work product becomes the property of the requester.

These are not the only models for commercial sites. Quirky[6] encourages users to submit ideas. Users critique and enhance the original idea. The users share in the profits according to their contribution.

Not all non-commercial systems are self-directed. Many citizen science sites create tasks that users work on [24–26]. For example, on eteRNA[7] users will be given RNA molecule puzzles to solve, until eventually they are addressing problems yet to be solved by the scientific community [27]. They can collaborate on solving the puzzles.

There are several issues related to task generation with respect to remixing. Communities in which users have autonomy will evolve according to the interests of users, influenced by the remixing and visibility features. The evolution is, in general, not predictable, and will depend on the sustained interest of the user base, and the overall concerns of the community. Things can be steered but not directed.

By contrast, systems that have requesters can be directed through the nature of tasks offered. Such systems can also use economic incentives to increase participation levels: monetary prizes, or piece work pay, or hourly pay. Or they can forgo economic incentives – citizen science systems are directed, but, in general, offer no cash compensation.

Mixtures of directed and undirected task generation are possible. Scratch users have chosen to create their own contests [8]. In this way, users are directing each other. By asking for tasks to be done, and then reciprocating, bonds are built [28]. In particular, systems exist in which people do chores for each other. These systems, called time banks, are focused on simple everyday tasks, but are related to systems such as Scratch in which online communities trade creative services with each other.

Wikipedia users create or edit pages according to their interests. But sometimes projects are initiated, in order to attract other users to fill out an area of content. Monetary incentives might be offered; or social incentives such as barnstars [29].

[6] Quirky.com

[7] Eterna.cmu.edu

Thus, users can do what they want, but incentives are used to accelerate work in areas that the users may not be naturally inclined to work on. This is parallel to what happens in academia – scholars often pick their own area of research, although offers of government grants can effectively attract them to different research topics, and calls for conferences and special issues of journals can shape the nature of research in a field.

In sum, there are several alternative control strategies for collective design – letting users follow their interest, or directing them, or providing a mixture of the two.

Following

Designs are created by people, and these people form networks.

Most collective design systems provide a permanent public identity, and also allow users to follow each other. In general, these are the systems in which full visibility is permitted. Following affects natural visibility – one is more likely to see the work created by someone one follows. Following is provided as a feature by Scratch, Thingiverse, and Github.

Figure 5 shows how the networks formed by the social network intertwine with the remix network. User A created a product that has been remixed by B (as well as other users). User B has created a product that user A has remixed.

From these links we can infer that user A and B share common concerns. This might be reinforced if we find that A and B follow each other. Just as the distance between designs can be measured, so can the distance between people, by counting hops in the network, or by counting the amount of time they remix each other's work.

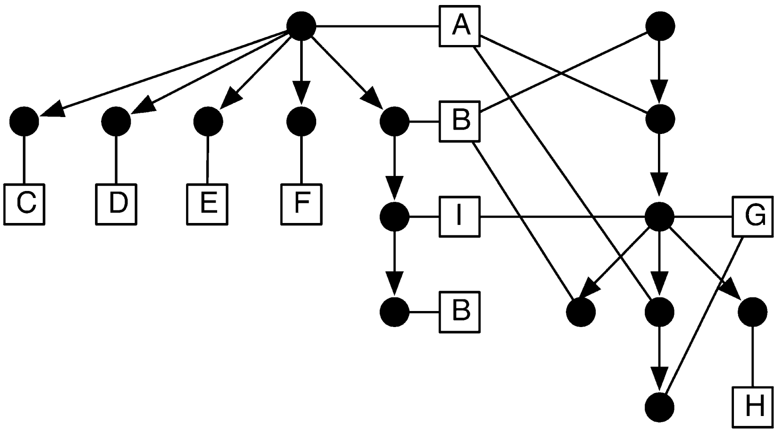


Fig. 5 Social and product networks intertwined. *Directed lines* show inheritance, and *undirected lines* show direct authorship

These distances might also inform the analysis of designs. That is, some designs may involve a large number of people each making small changes, and others might involve a small amount of people who edit each other's work.

Systems such as Github use the social information to provide reputational information – when users submit an enhancement for merger back into a particular project, the user's profile, with a history of past submissions, can be consulted as a proxy for, or another check on, the quality of the submitted work.

A crowd work site such as Mechanical Turk offers permanent but anonymous identity; users don't normally interact with each other. Requesters know users only through a cryptic ID. Reputation accumulates, but through an algorithm controlled by the platform owner.

There are other forms of following. Some systems allow following of projects. Github users can follow projects as well as developers. Wikipedia users can follow projects, but not individual editors.

Compensation for Designers

Creative work is sometimes performed for money. Many of the sites described here offer no money to the designers; several do. In general, the sites that offer money are the sites in which requesters control the tasks. Quirky is an exception: as a company, it gives designers the ability to pitch their own ideas, and then helps them develop the ideas, eventually taking control.

Compensation is related to fairness: should those contributing free content share in the profits that may flow to the systems' owners? It also is related to issues of sustainability: can a community be sustained without offering some of its members financial support?

Points in the Design Space

Table 1 shows the attributes for many of the systems discussed. It is apparent from the chart that there are two strong clusters: systems in which designers control task generation, and systems in which requesters control generation; these later systems usually provide monetary rewards.

Not shown on the chart is natural visibility, which is has many more potential states. Notable is Wikipedia: only the latest edits are naturally visible.

Sourceforge and Github both are open source communities, but Sourceforge is project-centric and Github is developer-centric; Github and Scratch have similar structures.

Table 1 Features of collective design systems

System	Remix type	Permitted visibility	Locus of task generation	Following	Compensation
Scratch	Single	All	Designer	Y	N
Thingiverse	Multiple	All	Designer	Y	N
Github	Single	All	Designer	Y	N
SourceForge	Single	All	Designer	N	N
Wikipedia	Single	Latest	Designer	N	N
Innocentive	None	Team	Requester	N	Y
Mturk	None	Individual	Requester	N	Y
Topcoder	None	Individual	Requester	N	Y
Quirky	Single	All	Designer	N	Y

Learning from Collective Design Systems

Examining the features of current systems is useful because it helps delineate the design space. But design spaces can always be expanded by defining new features. How might such new features be constructed? One way is through careful observation of design evolution in current systems.

Since most of these systems maintain histories of modifications, it is possible to see how changes in a system evolve. Patent network research provides a precedent. Patent networks have been studied in order to better understand innovation at the company and industry level [3]. Most of this analysis is based on the underlying metaphor of design as search.

The essential idea of the metaphor is that, given a particular problem, designers generate a series of alternatives [30]. The generative process involves making small or large changes to previous attempts. Small changes are called exploitation – a company makes minor changes to a well-understood product [31]. Large changes are called exploration – the company makes large changes, resulting in a product not as well understood. Larger changes are riskier. But a repeated series of small changes may not sustain interest in the marketplace.

Patents have been used as away of measuring the effects of different idea generation strategies [32]. Patents that end up being cited later are seen as being successful. The networks can be used to examine whether or not the patent directly descended from a similar patent from the same company, or whether the patent merged two disparate ideas. One important concept in this literature is technological distance [32]. Some technologies are closer to each other than others. This can be calculated in a variety of ways, for example by making use of the tree-based US patent office classification system and measuring the number of links between each pair of designs.

Then, one can compare patents that cite other patents that are very close, and patents that cite patents that are far apart. The general findings are that diversity in the patent network is better; this suggests that inventors should be encouraged to explore [3].

Patent data, however, has limitations. It takes years for patents to be granted. Citations may be driven by patent examiners, or lawyers, not by inventors. Attempts have been made to control for these difficulties [33], but there are many reasons why patents are cited, and design influence is only one of them.

Collective design system data might be examined using similar techniques. The potential advantages lie in the more timely nature of innovation networks. Citations in the form of remixes happen often in a time span of weeks rather than years. The modifications made in a remix can sometimes be measured exactly [34].

For example, changes to a program in Github or Scratch can be measured using string edit distance, the am of edits necessary to convert the parent to the child. The technological distance between edits might be measured by examining tags associated with the designs. If there is an ontology associated on the site, it could be used to perform the calculations. If not, then other ontologies, for example collaborative knowledge stores such as Wikipedia, could be used to calculate the conceptual distance between designs [35].

Just as with patents, the later citations of a design – the remixes – can be measured as an indicator of quality. In some cases, the actual uses of a design can also be measured, giving a fairly direct measure of practicality. For example, Thingiverse users who print someone else's design indicate this fact on the site. Thus, one could try to discover patterns of behavior in those that produce more cited or more used designs.

Designing New Systems

The analysis of collective design systems may provide insight into the way innovation happens, at the level of inventors, systems, and industry. This knowledge could potentially be used to improve existing systems or design new ones. One possible mechanism for exploring this space is experiment. In particular, crowd work web sites, because they limit visibility, lend themselves to experiment. For example, one recent experiment compared a competitive contest in which visibility was hidden to a contest in which visibility and prizes were shared. The study found that both systems generated good results – the competitive system generated more sustained participation, but the shared system allowed intermediate results to be shared [36].

Systems that already have full visibility might be used to perform experiments that alter natural visibility. For example, changing home pages and search algorithms may help discover ways to increase the overall amount and quality of invention. Ideally, systems would make prominent exactly those designs that are most likely to catalyze the production of a novel and practical design. They would function as a muse. To approach this ideal, recommender systems could be built and tested – systems that might, for example, suggest designs that are far enough way from existing systems to be novel, but not so far as to be irrelevant [37].

A recommender system, then, might be used to accomplish different goals at different levels. Three levels can easily be differentiated: individual, group and system. Individuals may want to increase their skills. For example, Topcoder participants may want to learn a programming language better. Groups may want to create common resources. For example, there are many individuals in the Thingiverse group that focus on building parts for robots; a group might want members to build modular components that all can share.

The systems may want to increase overall activity, and in particular may want to cross-pollinate ideas across interest areas. For example, the designers of the Scratch system may want to encourage community members who currently create animation to engage in game design, and vice versa. Or systems designers may have a different goal: to encourage users to develop and reuse modular components so that the entire community can avoid duplicating effort.

How can such recommender systems be created? They need to be implemented at the system level, in order to have access to complete data and to allow for experiments that test the effectiveness of the recommendations. In addition, it would be ideal if individuals and groups have mechanisms through which they can define the individual and collective goals that the recommender engine is focused on.

Conclusions

Collective design systems exist today in many forms. But the overall design space for such systems is large, and current examples only sparsely cover this space. New and better systems might emerge through both observation and experiment. Existing systems offer more granular picture of the social process of invention than before, and much of these data have yet to be analyzed. Observations may lead to hypotheses about how system features drive user behavior. These hypotheses can be tested through experiments on the systems. Moreover, with an experimental apparatus in place, new features can be designed and tested. In particular these experiments might focus on the design of recommender systems to guide collective design.

Building better collective design systems can potentially increase our overall capacity to innovate. They can be an inclusive way of involving people with many skills and interests in creative work. They may offer a way to explore more of a design space, more than can be explored by traditional design teams. That is, people working in parallel, following their interests, helping each other build skills, may play a role in future design, discovery, and invention. We can accelerate this process by improving the way that collective design systems are themselves designed.

Acknowledgements This material is based upon work supported by the National Science Foundation under grants 0968561 and 1211084.

References

1. Maher ML, Paulini M, Murty P (2011) Scaling up: from individual design to collaborative design to collective design. In: Design computing and cognition'10. 581–599. Springer
2. Visser W (1993) Collective design: a cognitive analysis of cooperation in practice. In: Proceedings of ICED. 93:pp 385–392
3. Fleming L, Mingo S, Chen D (2007) Collaborative brokerage, generative creativity, and creative success. Adm Sci Q 52:443–475
4. Kittur A, Nickerson JV, Bernstein MS, Gerber EM, Shaw AD, Zimmerman J, Lease M, Horton JJ (2013) The future of crowd work. In: 2013 ACM conference on computer supported collaborative work (CSCW'13)
5. Malone TW, Laubacher R, Dellarocas C (2009) Harnessing crowds: mapping the genome of collective intelligence. CCI Working Paper 2009-001, MIT
6. Quinn AJ, Bederson BB (2011) Human computation: a survey and taxonomy of a growing field. In: The 29th CHI conference on human factors in computing systems, ACM Press
7. Kyriakou H, Engelhardt S, Nickerson JV (2012) Networks of innovation in 3D printing. In: Workshop on information in networks
8. Nickerson JV, Monroy-Hernández A (2011) Appropriation and creativity: user-initiated contests in scratch. In: 44th Hawaii international conference on system sciences (HICSS'11)
9. Ipeirotis PG (2010) Analyzing the Amazon Mechanical Turk marketplace. XRDS Crossroads ACM Mag Stud 17:16–21
10. Lakhani K, Lonstein E (2012) InnoCentive. Com (B). Com (B) (August 17, 2011). Harvard Business School General Management Unit Case
11. Seneviratne O, Monroy-Hernandez A (2010) Remix culture on the web: a survey of content reuse on different user-generated content websites. In: Proceedings of WebSci10: extending the frontiers of society on-line
12. Tuite K, Smith AM, Studio EI (2012) Emergent remix culture in an anonymous collaborative art system. In: Eighth artificial intelligence and interactive digital entertainment conference
13. Cheliotis G, Yew J (2009) An analysis of the social structure of remix culture. In: Proceedings of the fourth international conference on communities and technologies, ACM
14. Kittur A, Chi E, Pendleton BA, Suh B, Mytkowicz T (2007) Power of the few vs. wisdom of the crowd: Wikipedia and the rise of the bourgeoisie. World Wide Web 1:19
15. Cranshaw J, Kittur A (2011) The polymath project: lessons from a successful online collaboration in mathematics. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM
16. Yu L, Nickerson JV (2013) An internet-scale idea generation system. ACM Trans Interact Intell Syst 3, Article 2
17. Gulley N (2001) Patterns of innovation: a web-based MATLAB programming contest. In: CHI'01 extended abstracts on Human factors in computing systems. ACM
18. Introne J, Laubacher R, Olson G, Malone TW (2011) The climate CoLab: large scale model-based collaborative planning. In: 2011 international conference on collaboration technologies and systems (CTS'11)
19. Kosorukoff AL (2001) Human based genetic algorithm. In: Systems, man, and cybernetics, 2001 I.E. international conference on. 3464–3469, IEEE
20. Kosorukoff AL, Goldberg DE (2001) Genetic algorithms for social innovation and creativity, Illigal report No 2001005. University of Illinois at Urbana-Champaign, Urbana

21. Yu L, Nickerson JV (2011) Cooks or cobblers? Crowd creativity through combination. In: The 29th CHI conference on human factors in computing systems. 1393–1402, ACM Press
22. Tanaka Y, Sakamoto Y, Kusumi T (2011) Conceptual combination versus critical combination: devising creative solutions using the sequential application of crowds. Annual Meeting of the Cognitive Science Society
23. Pirolli P, Card S (1995) Information foraging in information access environments. In: Proceedings of the SIGCHI conference on Human factors in computing systems, pp 51–58, ACM Press/Addison-Wesley Publishing Co
24. Bonney R, Cooper CB, Dickinson J, Kelling S, Phillips T, Rosenberg KV, Shirk J (2009) Citizen science: a developing tool for expanding science knowledge and scientific literacy. Bioscience 59:977–984
25. Raddick MJ, Bracey G, Gay PL, Lintott CJ, Murray P, Schawinski K, Szalay AS, Vandenberg J (2009) Galaxy zoo: exploring the motivations of citizen science volunteers. arXiv preprint arXiv:0909.2925
26. Wiggins A, Crowston K (2011) From conservation to crowdsourcing: a typology of citizen science. In: System Sciences (HICSS), 2011 44th Hawaii international conference on
27. Rowles TA (2013) Power to the people: does eterna signal the arrival of a new wave of crowd-sourced projects? BMC Biochem 14:26
28. Teodoro R, Ozturk P, Naaman M, Mason W, Lindqvist J (2014) The motivations and experiences of the on-demand mobile workforce, CSCW
29. Kriplean T, Beschastnikh I, McDonald DW (2008) Articulations of wikiwork: uncovering valued work in Wikipedia through barnstars. In: Proceedings of the 2008 ACM conference on Computer supported cooperative work. Vol. 47–56, ACM
30. Simon HA (1996) The sciences of the artificial. MIT Press, Cambridge, MA
31. March JG (1991) Exploration and exploitation in organizational learning. Organ Sci 2:71–87
32. Benner M, Waldfogel J (2008) Close to you? Bias and precision in patent-based measures of technological proximity. Res Policy 37:1556–1567
33. McNamee RC (2013) Can't see the forest for the leaves: similarity and distance measures for hierarchical taxonomies with a patent classification example. Res Policy 42:855–873
34. Nickerson JV, Corter JE, Tversky B, Rho Y-J, Zahner D, Yu L (2013) Cognitive tools shape thought: diagrams in design. Cogn Process 14:255–272
35. Genc Y, Sakamoto Y, Nickerson JV (2011) Discovering context: classifying tweets through a semantic transform based on Wikipedia. In: Foundations of augmented cognition. Directing the future of adaptive systems Springer, pp 484–492
36. Lakhani KR, Boudreau KJ, Loh P-R, Backstrom L, Baldwin C, Lonstein E, Lydon M, MacCormack A, Arnaout RA, Guinan EC (2013) Prize-based contests can provide solutions to computational biology problems. Nat Biotechnol 31:108–111
37. Ozturk P, Han Y (2014) Similar yet diverse: a recommender system. In: Collective intelligence

A Probabilistic Approach to Conceptual Design Support in Architecture

Alberto Giretti, Roberta Ansuini, and Massimo Lemma

Abstract The conceptual phase of design plays a crucial role about the overall performance of buildings. Early design choices have repercussions that are seldom controlled by designers in an effective way. More so, when quantities are involved in the decisions. This paper proposes a methodology for decision support during the conceptual design of buildings. The paper introduces a design knowledge representation framework and a computational mean for supporting the exploration of the design space through statistical inference. The proposed methodology is able to perform mixed inferences, combining symbolic and numerical computations.

Introduction

The conceptual phase of design plays a crucial role on the performance of buildings. Many fundamental features such as shape, volume, orientation, glazed surfaces, etc. are, in fact, defined and dimensioned in the early phases of design. Nevertheless, the consequences that early design choices have on the building performances are seldom controlled in an effective way. More so, when quantities are involved. This is mostly due to the complexity of the Architecture design. The extremely complex structure of the Architecture design knowledge, made of a large and intricate set of relationships, hinders the possibility of estimating quantities, and performing articulated scenario analysis in the early phases. Undeniably, taking optimal decisions during the early design phases requires the evaluation of a number of alternatives, whose complexity is very difficult to investigate in a systematic way. Designers normally use their experience, a background of knowledge and skills matured through years of practice, to ground early design decisions. Experience allows designers to estimate the consequences of their choices on a partially outlined project without using extensive calculations. Thus, designers' tacit knowledge is what mostly characterizes the early design phases, and in a certain sense represents the designers' signature in the final result.

A. Giretti (✉) • R. Ansuini • M. Lemma
Università Politecnica delle Marche, Ancona, Italy
e-mail: a.giretti@univpm.it

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_16

Structured observations of designers' behavior [3] suggest a further characterization of the cognitive processes underlying the conceptual design phases. In the early phases, designers seem to act very inefficiently in terms of time and effort spent. This is only an apparent deficiency; rather, it reflects the ability of identifying and focusing only on the relevant issues of the current problem formulation. The ability of performing this task in an effective way is among the key features of designers' expertise [4]. The concept of *problem framing* fully captures the nature of this cognitive process. Problem framing means the ability of structuring a problem space by pointing out relevant issues and their interdependencies. A fundamental role in problem framing is played by designers' experience, which allows for a rapid and effective recall of the problem structure and a rapid generation of the solution hypothesis.

Computational means supporting conceptual design phases should not introduce any cognitive gap or procedural barriers to the problem framing and to the consequent evolution of design thinking. This is a well-known issue that has been long debated in literature. Case Based Design (CBD) has been identified as one of the most promising reasoning paradigm for the representation of the cognitive processes in Architecture design [7, 8, 12]. Case Based Design is a problem solving technique that recalls previously solved problems from a database, or case base, and adapts them to the current design context. CBD resembles many aspects of designers' cognition. CBD is based on episodic knowledge. It reflects the typical arrangement of the architects' expertise, which is made of a large set of references to real buildings, clustered in many ways around a set of relevant design issues. Furthermore, the indexing-retrieve-adapt-assessment steps of the CBD inference cycle are, in fact, a model of the architect's observed design thinking. Finally, the problem framing is a direct consequence of the case base general arrangement. Case representation and indexing are problem-oriented. Therefore, case retrieval induces, as a side effect, a problem oriented qualification of the design space, which closely resembles the observed designers' problem framing activity. For these reasons, CBD has been used for many years as a reference paradigm for the development of artificial systems aimed at supporting design reasoning in Architecture, especially in the early conceptual phases.

However, practical implementations of CBD [15] have been in almost all cases limited to the index-retrieve phases. The adaptation and assessment steps have been substantially neglected. Case adaptation and assessment are critical tasks. They concern the integration of the retrieved case into the current design context. So, they are the reasoning steps that let design concretely progress. On the other hand, adaptation and assessment require a considerable amount of additional knowledge made of abstract design models. Design models are abstract representations of how properties of generic objects of the design domain, like trusses, frames, etc. combine to provide performances. Design models may be qualitative, quantitative or mixed. They can be either analytic or probabilistic, as simple as linear equations, or as complex as nonlinear differential equations. Design models are the complementary epistemic entities of design cases. Design models contain general knowledge about different cases, where design cases contain specific realizations of models. Cases provide examples of practical realizations of technical solutions, complementing the knowledge provided by models with information that is beyond the modelling effort. Tightly combined design cases and models are therefore the fertile background of any knowledge framework that would reflect the complex cognitive dynamics of conceptual design in Architecture. In a scenario where an effective combination of episodic and general knowledge is made available, case retrieval is not just a simple remind of a set of case parameter values. It involves the instantiation of the set of design models that constitute the representational ground for the subsequent adaptation and assessment phases.

Models are already widely used in the current Architecture design practice to assess design performances. The performance assessment is usually accomplished, by means of simulation, at the end of the detailed design phase, when the project is almost completely specified. Simulation is currently the only available aid for exploring the consequences of design choices in a quantitative way. This is, in fact, one of the main reasons that hinders designers in achieving a satisfying level of optimality in the final solution. There is a vast assortment of simulation software available today, but they are barely adequate for the conceptual design phase for a number of reasons [2, 5]. First, simulation software need a perfectly defined model of the artefact, rarely available in the conceptual phases. To fill in the knowledge gap, default values are used for the parameters that are not explicitly provided. A very tricky way to compensate the initial lack of knowledge. Uncontrolled decisions are thus taken about features whose relevance has not been investigated at all. It has been shown that using default values may entail significant and undesired deviations in the final performance of design [14]. Second, traditional simulation software are not adequate for figuring out new and original perspectives through quick browsing among alternative configurations. In this case, time matters, as human minds are limited in managing details that fail to appear concurrently and simultaneously. On the other hand, two digits decimal precision is not necessary in these phases. Systems capable of managing uncertainty and order of magnitude calculations are still adequate for supporting decisions. In this perspective parametric CAD software, which let designers systematically varying combinations of parameters, still fail to provide cognitively adequate support in the early phases. Finally, standard simulation software do not perform backward analysis. Proceeding backwardly from performances to technical parameters is a crucial process in the conceptual phase. In Architecture, backward analysis provides clues regarding dimensional, formal and technological solutions with respect to a certain level of performances. This is of paramount importance for a correct approach to the multi objective problem solving that is typical of any early design phase.

Summarizing, the development of a modelling framework that is computationally efficient end expressive enough to suit the requirement of conceptual design is still an open issue. The modelling framework should be able to trace the dynamic arrangement of models and cases into a coherent unified design representation. It should be able to integrate the conceptual level of the analysis, which is inherently symbolical, with the control of the design performances, which is mostly numerical. Finally, the computational frame should support a mixed variety of inferences, proceeding either forwardly (from assumptions to consequences) or backwardly (from consequences to premises), or in whatever mixed way.

This paper proposes a methodology for the design of decision support systems for the conceptual design phases in Architecture. The computational framework extends the Case Base Design by introducing the concept of Probabilistic Design Space (PDS). A PDS is a representation of the knowledge collected during the design process up to the current status. Hence, a PDS is a mean for statistically relating technological features within the horizon of the different design issues that have been formulated up to a certain point. The dynamic of the design process is captured by combining the referenced cases and the related models. The results is a well grounded representation of the current design, and a statistical model for performance assessment. PDS support forward and backward reasoning, easy browsing of design alternatives through what-if analysis, and a correct management of the uncertainty affecting the design parameters.

The PDS computation is based on Bayesian Networks (BN) [6, 9, 10]. In practice, PDS is a characterization of the Bayesian Network formalism for design support purposes. This paper introduce the PDS conceptualization and illustrates examples from the building acoustic design domain.

Probabilistic Design Models

A design domain is usually made up of a number of different sub-domains which correspond to different classes of performance. In Architecture design, for example, typical sub-domains are structural design, acoustic design, thermal design, etc. Each sub-domain is characterized by a number of relatively well structured sets of parameters that describe the technical features of the artefact to be designed. The distinction between features and performance is somewhat pragmatical and mostly depends on the design scope.

We define a *Design Model* a structured set of parameters, which describes the relationships occurring between a subset of the artefact features and a subset of the artefact performances. The structure of the design space is made by a set of relationships that bind the values of different parameters. A design decision consists in the assignment of values to one or more parameters.

A *Probabilistic Design Model* (PDM) is defined as a Design Model where parameters are represented by random variables and the relationships between parameters are represented by probabilistic conditional dependency relations. A design decision consists in the assignment of a probability distribution to some of the PDM parameters. The distribution may reflect either a smooth assignment of likelihoods to the parameter value set, or a sharp assignment of a 100 % likelihood to one of its possible values.

Bayesian networks (BNs) [11], also known as belief networks, are used as the computational mean to implement PDMs. BNs represent probabilistic models by means of directed acyclic graphs whose nodes stand for random variables. These

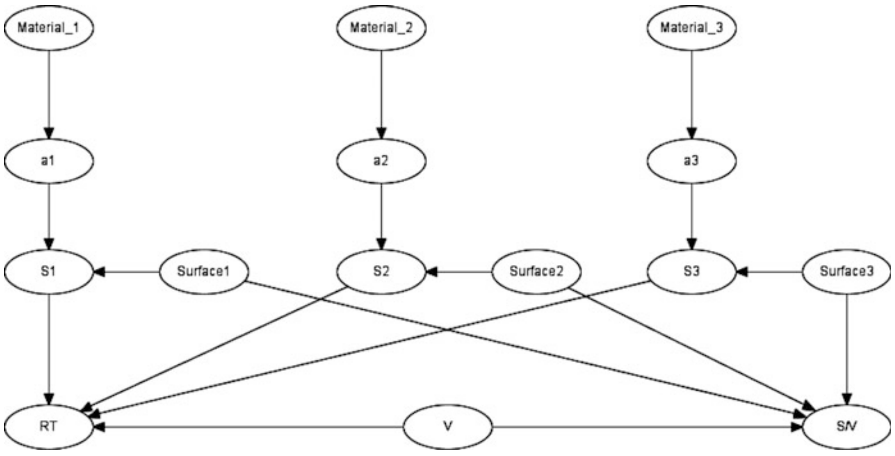


Fig. 1 A Bayesian Network implementing the Sabine equation for three different materials

can be either numeric or symbolic, while arrows among nodes denote conditional dependencies (usually interpreted as causal relationships) among variables. BN formalism is very general and can be used to represent analytic formulas, relationships implicitly present in datasets, as well as qualitative experts' opinions. The BN formalism supports a number of reasoning tasks such as deduction, abduction and a combination of both. The network in Fig. 1 is a PDM representation of the well-known Sabine Eq. 1 used to calculate the acoustic reverberation time *RT* of a room of volume *V*. According to the Sabine equation, *RT* depends on the acoustic reflection coefficient of the room finishing α_i in [m/s], on the room surfaces S_i in [m^2] and on the room volume *V* in [m^3].

$$RT = 0.16\frac{V}{\Sigma_i \propto_i S_i} \tag{1}$$

Therefore, the simple Sabine equation is indeed a design model of the acoustic design domain, since it binds one room acoustic performance, the reverberation time, with two features of the internal finishing, the acoustic reflection and the surface, and with a general feature of the room, the volume. Unfortunately, the analytical formulation (1) of the Sabine equation allows only forward calculations from V, α_i, *Si* to RT. Backward calculations are not easily allowed. Hence, the quick browsing of combinations of α_i, *Si* for different materials and for a given RT and possibly V may be rather cumbersome. On the other hand, a high precisions of the parameters is not necessary in the initial dimensioning. A simple translation of the analytic formulation of the (1) in a Bayesian Network solves these problems and, at the same time, maintains a sufficient numerical precision in the calculation. Furthermore, the resulting model can be hybridised introducing further non-numerical information, like the materials, which will increase the effectiveness in the decision support. Modelling Eq. 1 into a BN is straightforward. The first step involves the definition of the numeric domain of each variable. The discretization of the domain of real value parameters into discrete variables should be done carefully, controlling how it affects the accuracy of the overall calculation. Then BN technology provides algorithms for a direct mapping of the Sabine analytical equation into the probabilistic relations occurring among the nodes (i.e., calculating the Conditional Dependency Tables CDT), through a simple sampling of the leaf nodes (nodes with no incoming links).

The BN in Fig. 1 implements the Sabine equation for a room with three kinds of finishing. The a_i nodes represent the acoustic reflection of the different finishing, the $surface_i$ node their surface, etc. The $material_i$ nodes represent the finishing material and thus they are related to their acoustic reflection coefficient. S_i nodes are intermediate nodes introduced only for optimization purposes. *S*/*V* node is the Surface/Volume ratio of the room. The arrows represent the causal dependencies occurring among nodes, which in this case correspond to the analytical relations expressed by the Eq. 1.

Each node is a random variable, therefore it has a domain and a likelihood distribution over its values. The likelihood distribution of each node is updated once new evidences are inserted into the PDM. For example, the surfaces of *glass*, *wood* and *concrete* finishing that produces an RT value between 2 and 2.5 s for a volume of about 13,000 m^3 and an S/V ratio between 0.75 and 1, can be determined by entering the evidences (i.e., observing) of these ranges in the *RT*, *Volume* and *S*/*V* nodes. The updated likelihood distributions are then calculated by the BN evidence propagation algorithms (Fig. 2). In this simple example the BN evidence propagation algorithm proceeds in a mixed way: forwardly, from technical nodes to performance nodes, and backwardly, from performance nodes to technical nodes. By introducing further evidences in the Surface nodes, a compatible configuration is easily determined. Exploring different configurations is just a matter of some clicks. These calculations become even more interesting if the Sabine PDM is further extended introducing, for example, the cost of each material.

Learning PDM from Cases

The Sabine PDM is an example of a very general acoustic design model derived directly from first principles. One of the most powerful feature of the BN formalism is the possibility of inducing relationships among nodes from datasets. Bayesian networks in conjunction with Bayesian statistical techniques facilitate the combination of domain knowledge and data. The importance of prior or domain knowledge is of paramount importance in any real-world analysis, especially when data is scarce or expensive. Bayesian networks have a causal semantic that makes the encoding of causal prior knowledge particularly straightforward. In addition, Bayesian networks encode the strength of causal relationships with probabilities. Consequently, prior knowledge and data can be combined with well-studied

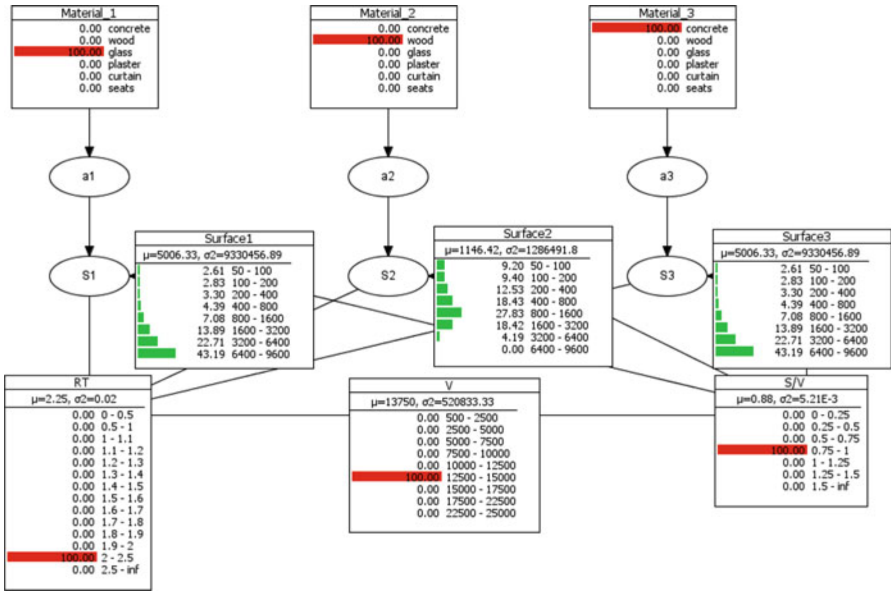


Fig. 2 The Bayesian Network implementing the Sabine equation with six nodes observed Material_1 = 'glass', Material_2 = 'wood', Material_3 = 'concrete', RT = [2.0, 2.5], V = [12500,15000], S/V = [0.75,1.0]. The probability distribution of the surfaces node related to the three finishing resulting from the propagation of the observations are displayed

techniques from Bayesian statistics. Bayesian networks have learning algorithms that can be used to induce the overall statistics of a design model from a set of design cases. Any dataset resulting from literature analysis, building simulation, on-site monitoring etc. can be used as an information source for the construction of PDMs. This is a unique and powerful approach to design knowledge shaping. Consider, for example, the problem of controlling sound and speech quality in auditoriums devoted to music and conferences. Technically speaking, the control of sound and speech quality requires a balance between the reverberation time and the amount of sound energy that reaches the listener within the first 80 ms, usually called early reflections. The parameter that represents the intensity of early reflections is called the C_{80} clarity factor and it is defined as it follows:

$$C_{80} = 10\log \frac{\displaystyle\int_0^{80ms} h^2(t)dt}{\displaystyle\int_{80ms}^{\infty} h^2(t)dt} \tag{2}$$

where *h(t)* is the impulse response of the room. Unfortunately, the value of this parameter depends on the sound path from the source to the listeners, on their distance, on the reflecting surfaces' materials encountered along the path and, in general, from the geometry of the building. Figure 3 depicts the analysis that should

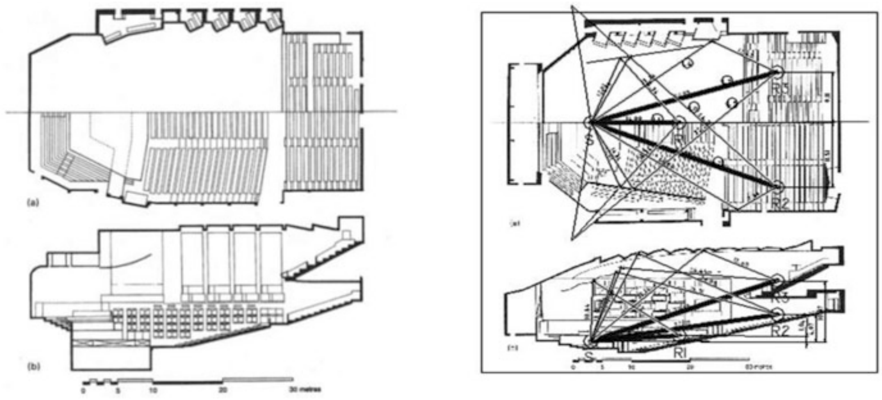


Fig. 3 Analysis of the Fairfield Hall early reflection paths

be carried out in order to estimate the C_{80} parameter for one auditorium. Thus, despite the theory of early reflections is well defined, there is no an easy way to represent the relationships among the C_{80} performance and the influencing geometrical and technological factors by means of a closed form equation. Simulation and/or on site measurements are necessary to fill in the gap. In these cases BNs learning algorithms can be used to induce the statistics of the involved parameters set from a data set. The general idea is based on the Bayesian inference theory. The uncertainty about parameters θ is encoded in a prior distribution *p(θ)*. Data *d* are used to update this distribution, and hereby obtain the posterior distribution *p(θ|d)* by using Bayes' theorem

$$p(\theta|d) = \frac{p(d|\theta)p(\theta)}{p(d)}$$

BN learning algorithms are rather complex [11] and their description is outside the scope of this paper.

From a practical standpoint, the use of BN learning algorithms is straightforward. What is needed is an adequate conceptualization of the problem and the estimation of the statistical relationships among the identified parameters. Concerning the early reflection example the literature [1] provides all the details to define a suitable conceptualization of the problem together with the data of a collection of relevant auditoriums. Once the set of auditorium cases has been analyzed and coded, the BN topology and the conditional probability tables are calculated by the BN learning algorithms. A simple star shaped network with the case identifier at its center, can be used in this example. Figure 4 shows the PDM induced from the database of cases combining measured C80 clarity factors which represents the intensity of early reflections in a number of outstanding theatres in the UK.

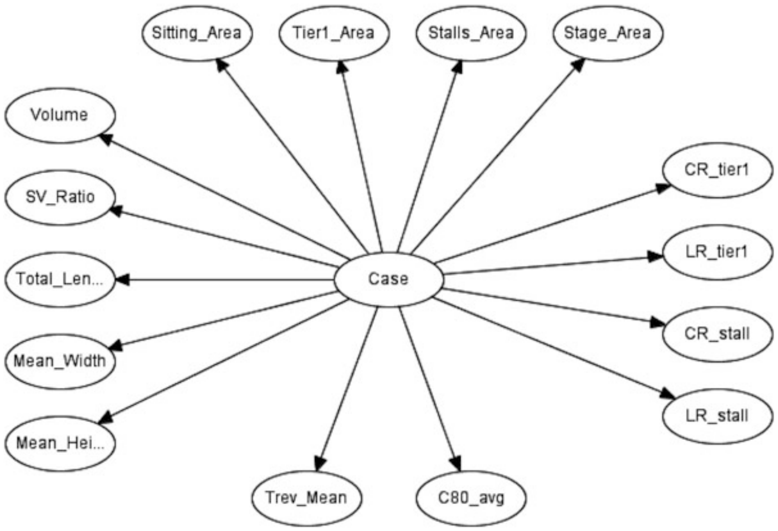


Fig. 4 A *star shaped* PDM induced from a dataset representing the relationship of the C_{80} factor and geometrical and features in a number of auditoriums in England

Probabilistic Design Spaces

We have seen so far that PDMs can be used as a very flexible mean for building domain models that combine numerical and symbolic information. We have also seen that PDMs support any sort of what-if analysis, and of forward and backward reasoning. Finally, we have seen how PDMs can be related to cases, providing model abstraction over datasets derived from case bases.

We define a *Probabilistic Design Space* (PDS) a combination of a set of reference cases with a set of related PDMs. PDMs arrange the feature set of a single case or of a set of cases according to a statistical model expressed by means of Bayesian Networks. For practical purposes, models are grouped in the general knowledge framework according to their design domain. Models are made of specific variables (e.g., acoustic reflection) and of variables shared with other models belonging to the same or to a different domain (e.g., dimensions, materials, etc.). The unique name assumption holds among PDMs' variables. Hence, shared variables allow cross model and cross domain inferences, by providing the bridge between different PDMs.

Figure 5 depicts an example of a PDS knowledge framework for Architecture design. Acoustic, thermal and structural are typical design domains in Architecture. The PDS is arranged according to two levels of abstraction. The episodic level, that contains framed representations of case features, and the model level that is made of a set PDMs shared among cases.

The main reasoning process is an extension of the CBD reasoning paradigm. The PDS can be accessed either through cases or through models. Cases are searched

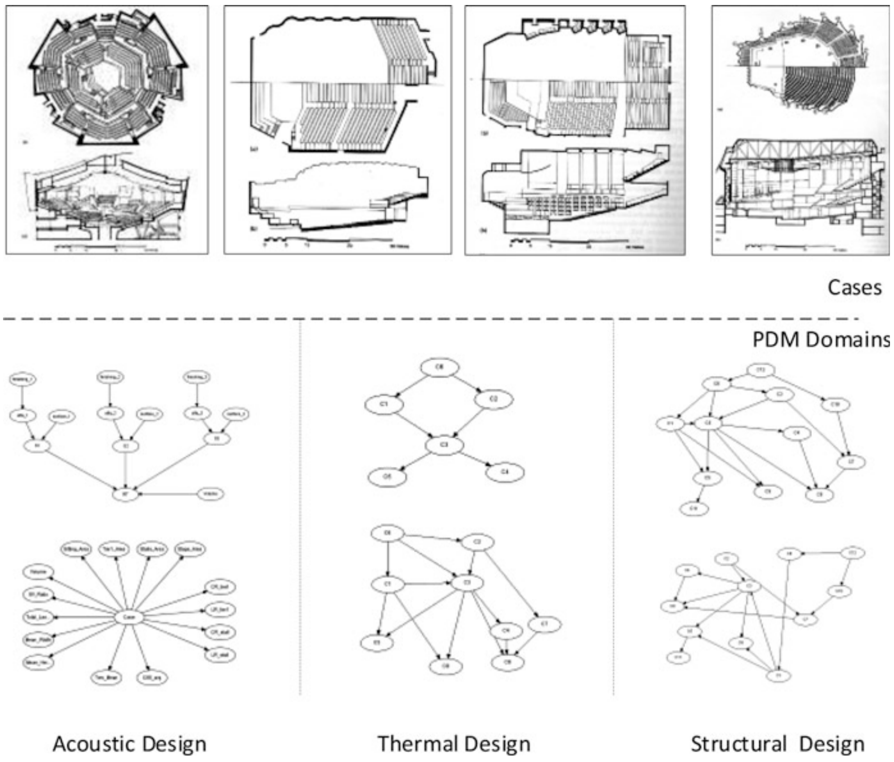


Fig. 5 An example of a PDS knowledge framework for Architecture design

and retrieved as in the usual CBD paradigm. Models can be browsed and instantiated autonomously. The retrieved element (a model or a case) plus the set of directly related PDMs and cases define the current *design focus*. Any design choice, consisting in the assignment of a model variable value, is propagated throughout the design focus. Therefore, the more the design proceeds, the more the PDS is configured as islands of partially instantiated qualitative-quantitative models. In this way, PDS allows designers to explore the design space by means of extensive what-if analysis, so that choices are made with a higher degree of awareness. The next section gives an extensive example of this dynamic.

Few more words are needed to qualify and to delimit the applicability of the proposed PDS methodology. Bayesian Networks are very powerful and very well studied means to represent knowledge. They are based on a sound probabilistic calculus and are able to perform complex statistical inferences. They have a well-defined mapping on formal logics, so they can be used to carry out deductions under uncertainty. Dynamic Bayesian Networks can implement Hidden Markov Chains, and therefore they subsume predictors like Kalman Filters. Finally BN learning algorithms are able to perform induction over data sets [13]. PDS is about the integration of the BN mean within the design process, therefore it is focused more on the cognitive adequacy of the knowledge framework resulting from the combination of case bases and Bayesian models than on the pure computational power of its components, which is already well known. The key features of the PDS approach is the seamless integration between models and cases, the integration of numerical and symbolic information, and the extreme flexibility of the resulting knowledge frame, which allows an accurate trace of real design processes. PDS is a knowledge based technique, therefore it inherits the limits of this kind of methodologies. The amount and the quality of knowledge delimits the scope and the effectiveness of any extensive implementation.

PDS in the Real Design Practice

The support of the PDS knowledge framework to conceptual design and its complementary role to CAD simulation tools are exemplified in this section. We will analyze some early steps of a typical design process of an auditorium. Environmental and economic constraints limit the building volume within 12,000 m^3. The auditorium design foresees about 1,500 seats and the hosting of both musical and theatre performances.

The architects have developed initial sketches of the volumetric arrangement of the auditorium (Fig. 6). The acoustic analysis reports a reverberation time of 0.7 s that is too short for the intended use, as well as an inadequate C_{80} value. At this design stage simulators are not able to provide any clue to correct the unsatisfying design performance. An almost blind trial and error procedure is the only support provided. On the contrary, the PDS framework is able to drive the design process towards directions that are consistent with the initial constraints.

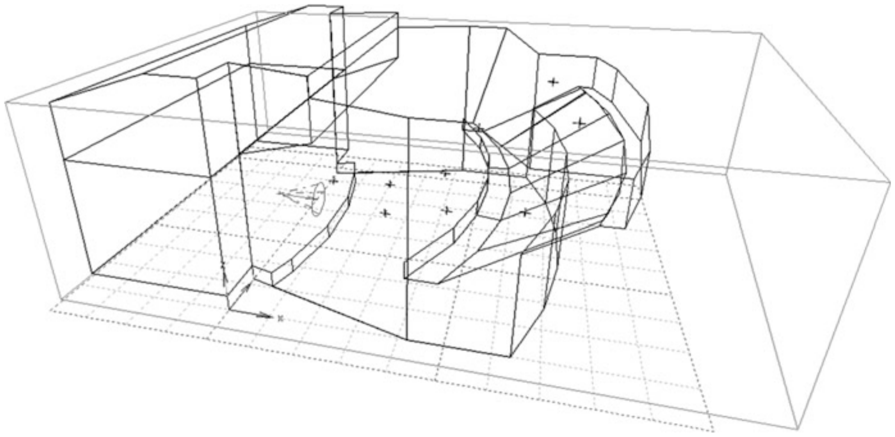


Fig. 6 Initial sketches (*dark gray*) with the volumetric constraints (*light gray*)

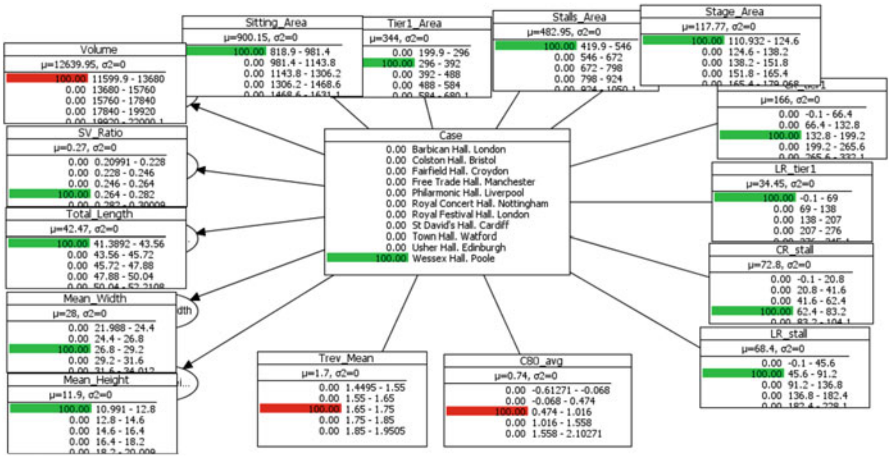


Fig. 7 Initial scenario analysis of the auditorium design by means of the C_{80} PDS

PDS operates in two ways. First the recalled case provides the parameters that are relevant for the current design issue, and secondly the related PDM set is capable of propagating the value assignment to any parameter to the others, thus providing the compatible value ranges and overviews of multiple design scenarios.

In this case, for example, the C_{80} PDM provides a link among the building dimension parameters and the acoustic performance parameters. This can be used to drive the first volumetric sketch. Figure 7 shows that assuming the volume as 12,000 m^3, and limiting the average reverberation time to an optimum value for speech and music, and selecting a good C_{80} value, set all the variable distributions of the C_{80} PDM to a very sharp form. In this way, the C_{80} PDM provides the ranges for the average volumetric dimensions (i.e., Total Length, Mean Width, etc.) and for the global sitting area, the stalls and the first tier surfaces that are compatible with the assumptions. The light gray box in Fig. 6 depicts the volumetric constraints coming from this initial acoustic analysis. As we can see the two hypotheses are not compatible. The sketched volume is considerably smaller than what is suggested by the C_{80} PDM and the sitting area as well is too small. Furthermore, the auditorium length cannot be increased too much, given that the main external accesses must be accommodated.

Therefore, the suggested dimensions must be adapted to the design context, probably limiting some of its acoustic performance. A possible strategy could be gaining volume by enlarging the surface of the stalls as much as possible, trying to increase the sitting area. Figure 8 shows the adapted design sketch.

The resulting acoustic envelope has a total surface of 3,365 m^2 and a volume of 11,996 m^3, with a SV ratio of 0.28. This solution raised the sitting area to approx 700 m^2 which can accommodate approx 1,400 seats with a 0.5 m^2 per seat. It is worth noticing that the areas of the stalls and tier1, amounting to 573 m^2 and 172 m^2 respectively, still reflect the C_{80} PDS scenario, while the total length of the building

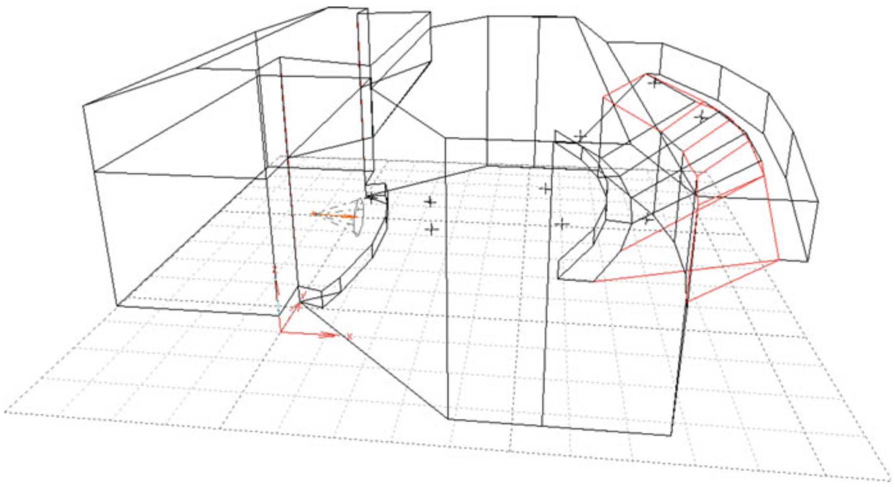


Fig. 8 The adapted design shows an increase of volume which has been obtained by enlarging the surface of the stalls

is considerably different. The adapted design, in fact, shows a reverberation time of 0.8 s at 1 KHz, as calculated by the CAD simulation tool, which is still far from the optimum reverberation time of 1.65 s.

This first design step, performed through the C_{80} PDM, leads us to an approximate solution that should be further reworked in order to correct the reverberation time. This can be done by moving the focus of the analysis and addressing the other PDMs that are related to the reverberation time. The Sabine PDM, which is based on a general functional model, is very helpful in evaluating different possible scenarios for the arrangement of the acoustic absorption of the internal surfaces for correcting the reverberation time. Reflecting the current design status on the Sabine PDM, and fixing the desired value of the reverberation time, leads to a scenario that still foresees a sitting area of about 700 m^2. This is compatible with the original design assumptions (Fig. 9). This scenario can be finalized by selecting other types of materials and their surfaces among the suggested values, providing a suitable design solution for the optimal reverberation time.

This simple design fragment shows how the PDS knowledge framework can drive the conceptual design phase by complementing the analysis performed by simulation tools. In more complex problems, the PDS guidance becomes even more fundamental. Consider for example the problem of defining the correct setup of the internal reflecting surfaces so as to obtain an adequate C_{80} value. This is a more complex problem since there are no well-defined theories about the arrangement of surfaces with respect to the listening areas and their sizing. The C_{80} PDM can provide some fundamental insight for guessing the initial sizing of the reflectors serving both the stalls and the first tier, because the C_{80} parameter is dependent only on the relative positions between the reflectors and the listening areas and on their sizes. Therefore, the C_{80} PDM is still applicable to our case even if the lengths of

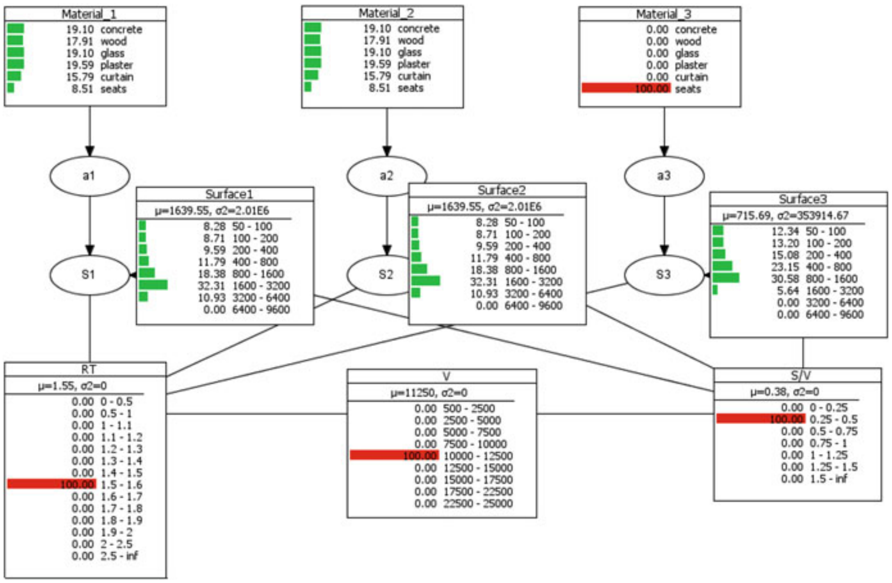


Fig. 9 The scenario analysis for the correction of the reverberation time. The reverberation time (RT node) has been forced to the desired value of 1.55. The sitting area (Surface 3 node) is calculated to 712 m^2 which is compatible with the initial design guidelines

the two volumes are different. Going back to the C_{80} PDM, we can set the status of our design and figure out the recommended sizes. Figure 10 shows that the only reflectors that are relevant for the C_{80} factor are the lateral ones and that they serve tier 1 with a total surface around 310 m^2.

The information, which determines a design strategy for the control of the C_{80} factor, is not very easy to obtain, because in general it's not clear whether the designer should intervene on the ceiling reflectors or on the lateral ones, and the optimal extent of both zones. In this case, the geometry of the reference case produced by the PDM analysis, Town Hall in Watford (see Fig. 10), drives the design directly to a plausible solution because it provides the geometric insights that are not in the scope of the PDM model. In this way, the tight arrangement of cases and PDMs in the PDS framework is able to provide a suitable knowledge background to support problem framing and solution assessment in the early phases of the Architectural design.

Conclusions

In this paper we have introduced the Probabilistic Design Space paradigm. PDS is a knowledge modeling and a computational framework supporting the early conceptual phases of design. We introduced the paradigm through extensive examples

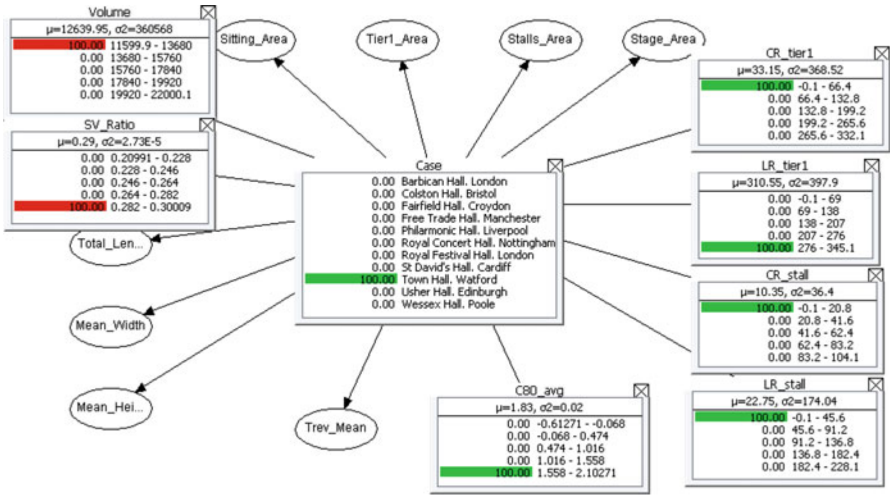


Fig. 10 C_{80} analysis performed through the C_{80} PDM. The PDM configuration suggests a reference case – Town Hall, Watford – which provides geometrical insights

avoiding the analytical complexities of the underling Bayesian Network computational formalism. We have outlined the knowledge modeling approach that are fundamental for the correct and effective implementation of the proposed paradigm. In the final chapter, we showed how the PDS paradigm can support conceptual design by providing necessary insights to drive problem framing and solution assessment in the early phases of the architectural design. Starting from this initial attempt which is focused mainly on the proposed framework's methodological aspects, future works will involve the extension of PDS to influence diagrams by including decision variables, the development of topological reasoning techniques and, of course, implementation of Probabilistic Design Spaces in other design domains.

References

1. Barron M (1993) Auditorium acoustics and architectural design. FN Spoon, London
2. Crawley D, Hand JW, Kummert M, Griffith BT (2008) Contrasting the capabilities of building energy performance simulation programs. Build Environ 43:661–673
3. Cross N (2004) Expertise in design: an overview. Des Stud 25:427–441
4. De Grassi M, Giretti A, Pinese P (1999) Knowledge structures of episodic memory in architectural design: an example of protocol analysis. In: Architectural computing from turing to 2000: In: Proceedings of the 17th ECAADE conference, Liverpool
5. EERE (2011) Building energy software tools directory. http://apps1.eere.energy.gov
6. Korb KB, Nicholson AE (2004) Bayesian artificial intelligence. Chapman & Hall, London
7. Kolodner J (1993) Case based reasoning. Morgan Kaufmann, San Matteo

8. Maher ML, Balachandran B, Zhang DM (1995) Case based reasoning in design. Lawrence Erlbaum, New Jersey
9. Matthews PC (2008) A Bayesian support tool for morphological design. Adv Eng Inform 22:236–253
10. Mittal A, Kassim A (2007) Bayesian network technologies: application and graphical models. IGI Publishing, Hershey
11. Neapolitan R (2004) Learning Bayesian networks. Prentice Hall, New Jersey
12. Oxman RE (1994) Precedents in design: a computational model for the organization of prec. knowledge. Des Stud 15(2):141–157
13. Russell S, Norvig P (2009) Artificial intelligence: a modern approach, 3rd edn. Prentice Hall, New Jersey
14. Wasilowski HA, Reinhart CF (2009) Modelling an existing building in DesignBuilder/EnergyPlus: custom versus default inputs. In: Proc. Of 11th International IBPSA Conference, Glasgow
15. Watson I (1997) Applying case-based reasoning: techniques for enterprise systems. Morgan Kaufmann, San Francisco

Part V
Design Theory

The Use and Misuse of the Concept of Affordance

Leonardo Burlamaqui and Andy Dong

Abstract Given the lack of agreement on the phenomenological elements of affordance, it is difficult to conduct empirical research to test systematic observations across contexts (e.g., industrial design and interaction design). To address this problem, this paper aims to establish a new understanding of the concept of affordance and its key concepts. Through a critical review of influential articles about affordance, the article identifies some uses and misuses of the concept. Then, a definition of affordance is provided, which delineates its foundational elements. Based on the definition, the article proposes a framework to explain how artefacts acquire affordances through the intentional behaviour of designers, certain material features, and contextual constructions. As a result, this research will contribute a new perspective on affordances that may help designers have predictable control over them when designing end-consumer products.

Introduction

Since the 1980s, numerous research and practice-oriented articles have been written about affordance across multiple design domains including industrial design, interaction design, and educational design, to name a few. Its meaning has been changing since Gibson [1, 2] gave birth to this term, sometimes in order to fill theoretical gaps, e.g., [3], and sometimes to shape the concept to fulfill the specific technical needs of a design discipline, e.g., [4]. Norman [5, 6] brought the concept of affordance to the design field, specifically to industrial design and human-computer interaction, and, despite his acknowledged research, focused on perceived affordances only. After that, some frameworks based on affordances have been proposed [3, 4, 7–12], but usually with the lack of pragmatism necessary for designers to design with affordance in mind [12] or with important concepts missing, such as the perceivability of the affordance and the influence of context on the recognition of an affordance.

L. Burlamaqui (✉) • A. Dong
University of Sydney, Sydney, Australia
e-mail: leoburla@estacazero.com

Regardless of efforts to better understand its meaning and implications on designing, there is still no conceptual agreement on the term affordance. Gibson's concept of affordance [1, 2] emphasised the perceiving agent as the source of affordance, and, in particular, the relation between the material properties of an object and the material properties of the perceiver. That is, there is a complementarity between the perceiving agent and the object; an affordance emerges from that complementarity. In contrast, other authors [12] have already described how many other theorists de-emphasise the perceptual aspect of affordances due to the theoretical framework that those authors bring to the concept of affordance.

In 1999, Norman [6] acknowledged the need for a clear meaning of affordance in writing that 'sloppy thinking about the concepts and tactics often leads to sloppiness in design. And sloppiness in design translates into confusion for users'. By clarifying the concept of affordance, and pointing out the common foundational elements that underlie it in a practical way, designers can improve the design of products and services, e.g., by eliminating errors, and have an understanding of the way innovation can be introduced from an affordance perspective, e.g., by introducing an affordance that opens up new end-use possibilities. Therefore, this research aims to develop a framework that will give designers a predictable level of control over affordances.

This article presents a critical, chronological literature review of influential articles on affordances. Articles were selected on the basis of their contribution to clarifying, extending, or refuting Gibson's original definition [1, 2]. Some uses and misuses of the concept are pointed out, and common foundational elements – which can be viewed as the necessary elements for a complete understanding of affordance – across those standpoints are provided and explained, one by one. Then, a definition of the concept is presented, which intends to conform as much as possible to Gibson's original definition, accompanied by some relevant considerations. Finally, a new framework is presented, which conveys the relationship between the context of an artefact and the artefact itself, and how these two dimensions influence the perception of affordances.

Origin of the Concept

Invented in 1977 by Gibson [1], a perceptual psychologist, the term *affordance* was presented as 'a specific combination of the properties of its substance and its surfaces taken with reference to an animal'. In 1979, it became part of his book 'Ecological Approach to Visual Perception' [2], which was an attempt to describe an ecological frame of reference for visual perception. Gibson stated that 'affordances of the environment are what it offers the animal, what it provides or furnishes, either for good or ill. . .it implies the complementarity of the animal and the environment'.

In his definition, Gibson points to the foundational elements affecting the perception of affordance: (1) *object*; (2) *observer*; (3) *environment*; and

(4) *complementary relations between these elements*. Based upon these elements, he postulates several properties of affordances: (1) affordances emerge in perception from the relation between these elements; they are not 'in' any of these elements per se; (2) affordances refer to action possibilities, that is, what the perceiver can do with the object; (3) affordances exist independent of the perceiver's ability to perceive it; (4) affordances exist independent of need.

In Gibson's definition of affordance, the most notable element is the relational aspect. Gibson claims that an affordance is not wholly dependent on the observer's perspective nor on the absolute physical properties of an object. By placing affordances in the realm of relations, the concept rejects any dualism of affordance as a property in either the object, observer, or environment. Rather, an affordance is predicated on the complementarity of these elements.

However, Gibson makes some claims about affordances that are not entirely in accord with his foundational elements. For example, affordances are binary, which means that there are only two existential possibilities: either they exist or they do not, without any middle ground. They simply exist – being perceived or not by the observer – *even if* they are species-specific. In this sense, affordances are functionally generic, applying differently according to the physiological tendencies of an animal. For example, research about mice and their ability to perceive a nest [13] demonstrates his assertion. As a result of the relation between the material properties of the object (nest) and the observer (mouse), the nest affords protection. To the mouse, the affordance of the nest is limited to being a mouse nest, and mice have a very limited conception of what objects could have this affordance. Clearly, for humans, the (mouse) nest affords no protection *if this were the need*. If the need were to hold sundry stationery, then a nest could afford storage. Gibson would claim that the affordance of storage was always present, waiting to be perceived. Our position, though, is that it is the role of the designer to make such affordances perceivable; else, the design is unsuccessful.

Other scholars have also noted this tension between a foundational element of affordances and Gibson's claim on the binary nature of affordances. As stated by Dohn [11], although 'it has often been used as if it were', we do not agree with this idea. To support our position, McGrenere and Ho [3] provide the following explanation: 'Recall the example of a stair being climbable or non-climbable by a particular individual. Reality obviously isn't this black and white; a gray area exists that is meaningful to the stair climber. For a particular individual one stair may be climbable with great difficulty whereas a different stair may be climbable with ease'.

Unfortunately, Gibson leaves the field with a broad explanation of the concept of affordance, which is not sufficient for its application by the design community. His definition of affordance is not immediately useful for the design community as it does not specify how to create the affordance intentionally, which is the role of a designer. Some other researchers have further confounded the definition in ways that hinder the designer from predicting the existence of an affordance. For example, McGrenere and Ho [3] state that an affordance means 'an action possibility available in the environment to an individual, independent of the individual's ability to perceive this possibility'. From a design perspective, this definition is impractical. Designers design distinct intentions into the visual form of an object to control the user's interpretation of the object [14]. If those visual forms are not perceptible, in a sense, the design(er) has failed. Thus, definitions that dissociate the individual from affordances fail to give guidance on *action possibilities to which affordances refer in relation to the user*, *how affordances relate to the user in an environment*, and *how affordances are perceived based on the user's prior knowledge*. In addition, the assertion that affordances are independent of the individual's ability to perceive them is inscrutable and may lead to misconceptions.

Considering that the aim of Gibson's book [2] is to clarify the problems of perception by taking an ecological account, it is justifiable that some of the definitions presented there lack specificity in fields of study other than psychology. As a result, the term *affordance* noticeably evokes different interpretations once inside domains such as interaction design, engineering design, and computer-supported collaborative learning (CSCL). This has led to confounding definitions and the misuse of the concept.

The Concept from Different Standpoints, Its Use and Misuse

In 1988, Norman introduced the term affordance to the design community by applying the concept to ordinary objects in his book 'The Psychology of Everyday Things' [5]. In this book, Norman writes that 'affordances result from the mental interpretations of things, based on our past knowledge and experience applied to our perception of the things about us'.

Norman deviates from Gibson's concept of affordance by (1) claiming that it relies on the actor's past knowledge and experience, highlighting their mental models and perceptual capabilities over their action capabilities, and (2) making a distinction between *real* and *perceived affordances*. As noted by McGrenere and Ho [3], 'Norman talks of both perceived and actual properties and implies that a perceived property may or may not be an actual property, but regardless, it is an affordance', where perceived affordances are those affordances that exist as a result of the actor's perception and, therefore, may not be what the object is actually for.

Because of the distinction Norman made and the way he referred to affordances throughout his book, it became unclear to which definition of affordance Norman was aiming. So later on, in 1999 [6], he clarified that his book was discussing perceived affordances only, not affordances in a general sense, since 'the designer cares more about what actions the user perceives to be possible than what is true', and reiterates that 'affordances reflect the possible relationships among actors and objects: they are properties of the world'.

Despite the ambiguity created around affordances, Norman's book [5] provided some valuable considerations about the psychological processes needed in operating and comprehending all sorts of devices. Thus, it rapidly became popular among designers, being one of the most important references in design, particularly in human-computer interaction, as it was considered to underpin some fundamental principles of an effective, error-free interface [15]. As a result, the term *affordance* was quickly acknowledged by the HCI community, and started to become adopted with various meanings, as observed by McGrenere and Ho [3]. Once inside the design field, the term *affordance* began to spread – reaching areas such as engineering design and computer-supported collaborative learning – and to acquire new meanings and categorisations, being applied in different ways.

Regarding Norman's definition of affordance [5], the HCI community was one of the first to realise that something was missing and started looking for solutions. Authors began to question the distinction between perceptible and imperceptible affordances. In 1991, Gaver [4] revisited Gibson's work [1, 2]. Influenced by the notion of complementarity, Gaver defined affordances as 'properties of the world defined with respect to people's interaction with it'. He wrote that the concept of affordance 'implies that the physical attributes of the thing to be acted upon are compatible with those of the actor, that information about those attributes is available in a form compatible with a perceptual system, and (implicitly) that these attributes and the action they make possible are relevant to a culture and a perceiver'.

By separating affordances from the information available about the physical attributes of the artefact, e.g., text and labels, Gaver was able to establish a distinction among (1) *correct rejections*, (2) *perceptible*, (3) *hidden*, and (4) *false affordances*. About this categorisation, Gaver explains that 'perceiving that a doorhandle affords pulling does not require a mediating concept because the attributes relevant to pulling are available for perception'. On the other hand, 'knowing that a key should be turned inside a lock does require mediation because the relevant attributes are not available'. Thus, the given examples correspond to perceptible and hidden affordances, respectively. An example of false affordance could be a door handle that (has information that) suggests pulling while it actually affords pushing only. A correct rejection, on the other hand, would be a door handle that neither affords pulling nor (has information that) suggests this kind of action. Therefore, while false affordances deal with misperception, correct rejections deal with the disregard of a certain action, which should not be interpreted as the absence of perception.

Although, from a design practice standpoint, it seems reasonable to distinguish affordances from the information available about the artefact, it is rather difficult to (1) define the conceptual boundaries between the *perceptual information* and the *physical attributes* of the thing to be acted upon, as they seem to be quite intertwined and overlaps between them may exist, (2) put in practice this distinction by successfully identifying affordances and perceptual information, and (3) provide an overarching definition of affordance that is useful to design practice and empirical design research.

Concerned about the issues around the different uses of the concept of affordance within the HCI community, McGrenere and Ho [3], in 2000, revisited Gibson's original definition [1, 2]. According to McGrenere and Ho, and we agree with their analysis to some extent, 'Norman collapsed two very important but different, and perhaps even independent, aspects of design: designing the utility of an object and designing the way in which that utility is conveyed to the user of the object'. Norman [5, 6] favoured the latter over the former due to his emphasis on perceived affordances.

In order to provide clarification, they reviewed the original definition of affordance [1, 2], compared it with Norman's [5, 6], and pointed out an ambiguity – which has already been addressed in this paper – in Norman's definition and use of affordances. Then, they conducted a review of the HCI literature that demonstrated that this ambiguity has led to widely varying uses of the concept of affordance. Lastly, they expand their clarification into a framework for design.

The framework proposed by McGrenere and Ho [3] establishes a relationship between two key elements in the concept of affordance: the ease with which an affordance can be undertaken and the clarity of the information that describes the existing affordance. Based on this relationship, they claimed that improvements in design could be achieved when both elements are maximised.

McGrenere and Ho's contribution [3], in terms of clarification of the original concept of affordance, is undeniable. Their work was indeed very useful, but it stopped short of providing any significant improvement in turning the concept of affordance – which for them meant 'an action possibility or an offering' that includes, but is not limited to, physical interaction – into a concept for the practice of creating design works. They continued with Gaver's line of thinking [4] by highlighting two aspects of design: designing affordances and designing the information that specifies the affordance, and relating them to usefulness and usability, respectively.

In McGrenere and Ho's work [3], one of the claims that sound awkward is when they state that affordances can be undertaken, as they were really actions or tasks, but at the same time they declare that affordances can be designed. Actions cannot be designed, as they are solely undertaken by the user, i.e., the user is the subject within that subject–object relationship. However, affordances can be designed, which leads us to think that, once more, the concept of affordance is still not clear in the design field.

In 2001, Maier and Fadel [7], started to advocate the application of a theory of affordances to engineering design, and, in 2009 [8], they proposed a new design theory based on affordances. Its purpose is made clear when they say that 'affordance based design prescribes that designers analyse the affordances of each embodiment, and attempt to remedy negative affordances during the design process' [8]. Due to their efforts, affordance was no longer viewed as just a useful concept within some technical activity domain, as they advanced it as the foundation of a design strategy.

In their research, Maier and Fadel [7, 8] claimed that affordances may exist between artefacts – which were called *artefact–artefact affordances (AAA)* – and not just between artefacts and users – known as *artefact–user affordances (AUA)* – as posited by the design research community up to that moment. Affordances between artefacts may only be accepted as true if these artefacts are capable of perceiving them, e.g., through a sensor. Otherwise, it would mean that affordances exist regardless of perception, which is implied from the authors' work, as they do not explicitly state the above condition. The idea of affordance as an *aperceptual* concept goes against its very nature [12], given that affordances are established by a complementary relationship between users and the environment through perception.

Meanwhile, in 2004, Kirschner, Martens and Strijbos [9] developed a model for the design of CSCL environments, based on three distinct types of affordances: *technological*, *educational*, and *social affordances*, to claim that affordances could have some kind of purpose or meaning other than the actual use of the artefacts they embody. In 2006, Suthers [10], in turn, proposed the study of what he called *technology affordances* for intersubjective meaning making as an integrating research agenda for the CSCL field. In this work, Suthers used the concept of affordance as defined by Norman [5, 6], i.e., perceived affordances, and explained that 'understanding the affordances technology offers for intersubjective meaning making is as foundational to CSCL as understanding learning'.

Similar to the HCI field, the CSCL community embraced the concept of affordance, and equally with various interpretations. So, as a response for the way the concept of affordance has been used in the CSCL field, in 2009, Dohn [11] was responsible for providing a renewed look at this theme by proposing a Merleau-Pontian account of affordances. In this work, she made use of the concept of *body schema*, which is complementary to the concept of affordance. She postulated that '*affordance* signifies that meaning is in the world, not in the head, and *body schema* signifies that the world is meaningful because of what we can do in it'.

Although the body is an important component in any action the agent undertakes, as well as in the way an agent perceives and experiences the environment, when it comes to perception, the mind is paramount. Compared to the mind, the body plays a small role in perceiving affordances, though, as it works merely as a vessel, as a variable whose influence on perception is quite limited. The mind, on the other hand, is where cognitive processes take place, from perception to meaning making. While the body matters when it comes to its relation to the artefact, remembering that affordance is a relational concept between the material properties of the artefact and the perceiver, the (human) mind gives us additional capacity to discover the affordances of artefacts through prior knowledge. The mind gives meaning to what the body can do to the artefact.

To sum up, despite the new interpretations of the term *affordance*, it is important to note that Gibson's original definition [1, 2] is still the most often referenced meaning, suggesting that the foundational elements of Gibson's concept of affordance are generally accepted, though what these elements 'claim' about the properties of affordances remain debated. However, some of the authors previously cited here may have provided confounding definitions of affordance, which has led to its misuse by the design community [6, 12]. Furthermore, while Norman [6] believes that the designer cares more about perceived affordances, this judgment should be made by the designers themselves. The judgment is supposed to be concerned with both aspects – whether or not affordances are considered to be separate from the information available about them – as subjectivity and objectivity are part of the cognitive process, being equally important for perceiving affordances, regardless of the cognitive load they exert. Within this context, real and perceived affordances are a matter of taxonomy.

In a design context, the important issues about affordances are the strategies designers could employ to make affordances more evident and understanding how the user's prior knowledge and the environment in which the user and artefact exist can shape affordances despite the intentions of the designer. It is to these ends that we review the common foundational elements of affordances as a base for a framework to address these issues.

The Common Foundational Elements

Based on the aforementioned definitions of affordance, here we provide what we believe to be a common set of themes across all these standpoints. Despite modifying the definition and use of affordance in discipline-specific ways, each of the standpoints described previously share the following common foundational elements, which should be regarded as the variables that affect the perception of affordances:

Artefact

For the purpose of this study, an artefact refers to an object, tangible or not, made or given shape by humans to be used or to be acted upon. In general, designers have a particular interest in objects, which depending on how they relate to each other, as well as their circumstances, may constitute a single artefact. Nevertheless, it is important to note that affordances are precisely elicited from the *properties* and the *behaviour* of an artefact. While the properties are the (physical) attributes of an artefact – such as size, volume, proportion, weight, color, and texture – the behaviour refers to the particular ways by which an artefact exists and, therefore, interacts with the environment (its immediate context).

For understanding what behaviour means, it is essential to consider that it *happens in time*. The behaviour forces us to view artefacts as dynamic entities in the environment that are subject to change. The distinction between properties and behaviour does not mean that the former is the static feature of an artefact, while the latter is its dynamic counterpart. Far from that, the properties of an artefact *exist in time*, and are subject to change. Behaviour refers to how an artefact responds to the environment; it is the result of a cause-effect relationship with the latter. For example, a swinging pendulum has *swinging* as one of its behaviours due to the way its properties, e.g., structure, material, and mass, respond to the environment. A melting ice cube, on the other hand, has *melting* as one of its behaviours, while some of its properties, e.g., volume, proportion, and mass, change considerably as a result of this behaviour. These behaviours can also influence affordances, such as melting ice affording friction reduction.

It is worth mentioning that the definition of behaviour provided here is quite similar to the concept of *actual behaviour* from Gero's *Function–Behaviour–Structure (FBS)* model [16], specifically in relation to its clarified definition [17], which was paraphrased by Vermaas and Dorst [18] as 'the artefact's actions or processes in given circumstances of the natural environment'.

Agent

An agent refers to someone or something capable of perceiving an affordance, and capable of acting upon its corresponding artefact. Thus, it can be defined as a potential perceiver and actor. Having said that, machines can be equipped and programmed in such a way that they can be treated as agents.

Agents are driven by *motivations*, which refer to the true reasons of their intended actions. Alternatively, when aimed at humans, which are the agents of greatest interest here, this definition may be interpreted as needs and desires, while it can be viewed as results and outcomes for machines and, finally, just as needs for nonhuman animals. About this subject, Gibson [1, 2] states that 'the affordance of something does not change as the need of the observer changes'. However, this need is capable of sharpening the perception of a specific affordance, standing out from the others.

In addition, agents are provided with what we call here as *knowledge*, which is the sum of what has been perceived, discovered, and learned. With regard to humans, however, this definition encompasses experience, culture, and beliefs.

Environment

Environment is the container of both artefact and agent. Due to its characteristics and its content, it is naturally active and dynamic. It is where the potential use of an artefact is made available and, therefore, the place in which the relationship between agent and artefact, i.e., interaction process, is established.

Since an artefact is available in the environment, the latter takes on a decisive role on how an agent perceives the affordances of the former, being capable of influencing perception. The result of this relationship (between an artefact and the environment) is a concept that is not only strongly related to the environment, but is also likewise dynamic in nature: the *context* within which the artefact lives.

Although it is only implied in each of the previous definitions, the context turns out to be a concept of utmost importance for the correct understanding of affordances, as it permeates all its common foundational elements. Just as the environment, the context gives off information, and this information, in turn, influences how affordances are perceived.

Perception

Perception is the sensory experience that involves the use of the five traditionally recognised senses, i.e., sight, smell, taste, touch, and hearing. Additionally, it can be viewed as the primary link between an agent and an artefact, from an intended action perspective. An intended action always requires an agent, as well as perception, while an unintended action does not. Unintended actions result from an agent's false perception or from the absence of perception. Returning to our previous example, if an agent accidentally falls inside an open manhole, we can infer that it has occurred due to misperception, e.g., the manhole appeared to be closed, or to the lack of perception, e.g., the agent was distracted. Therefore, in affordance, perception can be viewed as a moderating variable because it affects misperception of the potential action.

Potential Use

Potential use is an element that encompasses Gibson's complementarity concept [1, 2]. It refers to an action that might occur upon the artefact from an agent's perspective. This action always needs an agent, i.e., the actor, and an artefact to be acted upon. It is thus based on the relation between the physical capabilities of the agent and the material properties of the artefact.

Actions can be intended or not, and although the understanding of an intended action is clear, which is when an action actually matches what an agent previously had in mind, i.e., a purpose, an unintended action might not be. Whenever the action does not match what an agent had in mind, this can be considered an unintended action, even if the agent did not mean to act upon an artefact. For example, the action of an agent accidentally falling into an open manhole can be classified as an unintended action.

While the term *action* usually implies a subject, the term *use* implies a subject and an object, which is one of the reasons why *potential use* has been chosen to the detriment of *action possibility*. In addition, *use* means *to apply for a purpose* – that is, doing *a* for achieving *p*, where *a* means one or more actions, and *p* means purpose – which fits well into an affordance account, due to the fact that only actions that have intention, i.e., intended actions, are of interest. Thus, the concept of potential use has to be viewed as the necessary actions for achieving a purpose that, by any chance, can be assigned to an artefact by an agent. In other words, to put an artefact into use is to set the purpose one assigns to it, which we call *assigned purpose*, in motion.

According to Gero [16], ‘design is purposeful, and the activity of designing is goal oriented’. Thus, designing an artefact means creating something capable to meet specific needs and desires, i.e., purposes. Put it that way, design is a process in which an artefact is built to achieve a purpose, its purpose, which we call *designed purpose*. However, whatever the purpose of an artefact might be, designed or not, it can only be achieved when an artefact is put into use by an agent, which is a consequence of the process of assigning purposes, as previously explained. Given this, the designer has to be aware that an assigned purpose might differ from the designed one, resulting in what we call *non-designed purpose*, so the outcome can be quite different from what was originally expected. In this context, it is clear that the agent’s motivations are key, as they might promote, for example, a purpose that actually puts people’s lives at risk.

From a design perspective, the operation of an artefact can be considered correct or not, which refers to the way an artefact should be manipulated to achieve its designed purposes. In this case, a comparison between the actual operation of the artefact by an agent, which we call *actual use*, and its expected operation, i.e., the use planned by the designer, that we call *expected use*, takes place. If they match each other, one can infer that the agent correctly manipulated the artefact.

Although this is not the aim of this paper, we acknowledge that the expected use of an artefact is defined in accordance with its designed purposes and, therefore, is based upon its functions, which fits well into the FBS model and its conceptualisations around function [16, 17]. Alternatively, by taking into account Vermaas and Dorst’s considerations [18] on the FBS model, we could say that, while the designed purposes are the reason for which an artefact exists, i.e., they are the originally designated goals of an artefact, and functions are ‘physical dispositions of an artefact that contribute to the purposes for which the artefact is designed’, the expected use refers to a process in which functions are manipulated as a means for achieving the purposes of an artefact. Consequently, the expected use of an artefact may be interpreted as a *use plan*, which, according to Vermaas and Dorst [18], is ‘a plan for achieving the purpose associated with the artefact that contains at least one considered action that involves the manipulation of the artefact’.

Definition of the Concept

Now that the five common foundational elements that underlie the meaning of affordance across all the provided standpoints have been described, we are finally capable to postulate that the concept of affordance refers to *cues of the potential uses of an artefact by an agent in a given environment*.

Based on the above definition, the only uncontroversial claim about affordances is that they are about action possibilities relative to the agent. To properly understand what affordance is about, first and foremost, it is essential to consider that affordances are not the use itself, but the call for it. They are cues that *invite* an agent to act upon the artefact. Gibson [19] describes a car as providing to the driver 'a sort of field which yields a variety of perceptual cues and which invites and supports specific actions. The impressions constituting it are kinesthetic, tactual, and auditory, as well as visual, and they interact with the impressions from the terrain to produce the totality of cues on which the driving-process is based'. In other words, affordances are the means by which an artefact conveys the ways by which to be operated. Its operation, in turn, implies a purpose, which may be different from the designed ones. Affordances are, thus, strongly related not only to the designed purposes of an artefact, but also to the latent ones, i.e., assigned purposes. So, if we put the agent aside, when an artefact is used differently from its expected use, this is because of what it affords, as affordances are indicative of its latent purposes. Yet affordances alone are not the only reason why an artefact may be used differently from the way it was supposed to, since context has been disregarded, as well as the agent's knowledge and motivations. It is obvious that those elements play a role in the process of perceiving affordances, as it will be shown later.

In addition, it is important to note that every affordance needs an agent to exist. If there is no agent capable of perceiving something regarded as an affordance, its existence turns out to be a philosophical question.

Although, in the literature, it appears that artefacts have just a few affordances, if we consider that they are cues to the necessary actions for achieving purposes that can be assigned – whatever the odds might be – to an artefact by an agent, this scenario would probably change. Based on that assumption, it could be inferred that every artefact has an uncountable number of affordances, which is in fact what we and others [12] postulate. In addition, we have to be reminded that affordances are not binary, like being turned on or off. To be perceived, they are subject to a few variables, which will be addressed soon.

Considering that perception is a rather complex process, a clarification around the conditions an affordance is perceived seems to be necessary. However, before stepping in this matter, it is important to note that, although affordances do not always depend on perception to exist per se, from the design point of view, perception is paramount and, thus, has to be included for a complete understanding of affordance. If affordance is treated as an *aperceptual* concept, it would be rather difficult to evaluate its effectiveness for a given artefact, as well as the design of the artefact itself. If the designer, for example, does not understand what is happening with the affordance, given the fact that the agent, i.e., the end-user, is not able to perceive it, the designer might not know how to design the artefact, which will eventually result in failure. So, there is no point in conceptualising something that does not involve the process by which an agent detects stimuli from an artefact.

Once realising that affordances are indicative of the latent purposes of an artefact, they should be thought of not as phenomena that are attached to it, but as a dimension in which the whole artefact provides cues of how to make use of it. Therefore, affordances are perceived by taking into account the whole artefact – including all the parts that are needed for achieving its purposes and made available for perception, attached to it or not – even if its operation is limited to just a small part of it. To illustrate this point, consider the power button of a TV's remote control: alone, it affords just *pressing*, which is actually the button's behaviour from the agent standpoint, but if we think about both TV and remote control as a single artefact, it could be said that the button affords *pressing for turning the TV on/off*, which is indeed an affordance, as it is related to a purpose, in this case the designed one; to fulfill the affordance means producing an outcome that depends on the artefact as a whole, which would not occur otherwise, i.e., the remote control alone.

An affordance is correctly perceived when, to accomplish a specific task, an agent recognises the corresponding use an artefact has to offer, that is, the necessary actions for achieving its assigned purpose. To understand the feasible use of an artefact, some cognitive effort is needed, even if it involves a *direct perception* [1, 2] or an *automatic process* [12]. Regardless of the complexity of an affordance, the process of perceiving it always exerts some cognitive effort from agents. This match between perception and the potential use of an artefact is based on knowledge, or in specific cases, on primitive instincts.

Aside the most basic and primitive affordances, which we could call *instinctive affordances*, such as the artefacts that afford nesting for mice [13] – as they are successfully perceived without any previous experience, i.e., knowledge, because the meaning of such an artefact appears to exist in their brains from birth. The cognitive effort relies on memory, which is the result of knowledge. In other words, without knowledge, an agent cannot perceive affordances other than the instinctive ones.

In the example provided earlier, the affordance that exists in the power button of a TV's remote control would probably never be perceived if, before anything else, an agent does not know what a button is. In this case, the only way to overcome this obstacle is by analogy, which depends on the agent's knowledge of artefacts whose properties and behaviour may be similar to the button. Else, the affordance could be guessed at some point. It is worth mentioning that this process in which an agent tries to make use of an artefact by analogy refers to what the HCI field calls *familiarity* [15] or *guessability* [20]. Hinze-Hoare [20] defines it as the 'degree to which the user's own real world personal experience and knowledge can be drawn upon to provide an insight into the workings of the new system'.

With regard to the process in which affordances are perceived and, then, purposes are assigned, the agent's motivations are central and should not be overlooked. But, together with the agent's knowledge, these are not the only elements at play. Context, which was the missing part in this consideration, provides relevant information about an artefact and influences how affordances are perceived. Depending on the artefact, the context is responsible for strengthening and weakening certain affordances, as it will soon be shown.

Therefore, for the purposes of empirical design research, the concept of affordance should be viewed as a problem of transmission of cues from the artefact and the environment to the perceiver. It is a problem about the way the information of an artefact is made available to the agent, and the way this information is captured by this same agent.

The Framework

The purpose of this framework is to provide the necessary knowledge to assist designers in creating new artefacts, or introducing new features and functionalities, within an affordance perspective. The framework has to consider that the problem for the designer is to strengthen or weaken the perception of the affordance, that is, impose control over the perception of an affordance. By embracing the aforementioned elements, the framework should be viewed as an outcome of our considerations around the concept of affordance.

During the process of building the framework, Bernstein's notions of *classification* and *framing* [21] emerged as an interesting way for conveying the groundwork in which the elements that have been selected earlier should be built upon. For Bernstein, education is about the transmission of knowledge, which is governed by the regulation over what knowledge can be transmitted and its relation to other knowledge (classification) and how the knowledge is transmitted and acquired (framing). Therefore, based on the similarities that have been found, we decided to adopt Bernstein's terminology and framework, by applying some necessary changes and making them suit the elements that have been previously selected, so that the resulting framework would meet the purpose.

In affordance, classification refers to the degree to which the artefact is perceived as it was meant to be, i.e., its designed purposes, in relation to the context. Framing refers to the degree to which the artefact is perceived in relation to its own constraints, i.e., the artefact's properties and behaviour, to the detriment of the agent's knowledge and motivations.

On the one hand, when an artefact has a strong classification it means that, regardless of how the context presents itself, the understanding of the artefact's affordance remains intact; and, when it has a weak classification, it means that non-designed purposes may be assigned to the artefact, depending on the context. On the other hand, in a strong framing the artefact itself brings the frame, i.e., the affordance is tightly bound to the artefact, due to the strength of its properties and behaviour to the detriment of the agent's knowledge and motivations. As Gibson [2] wrote, 'the object does what it does because it is what it is'. In a weak framing the agent brings the frame, i.e., the affordance is not so tightly bound to the artefact, due to the strength of the agent's knowledge and motivations to the detriment of the artefact's properties and behaviour. Our knowledge about artefacts is not generated strictly through the act of perception; prior knowledge will shape the perception of suggested possibilities. The predictive brain hypothesis [22] suggests that even prior to our perception of the artefact, the brain has already activated analogical references of *what this artefact is most like* based upon other artefacts in our field of view.

To allow some form of measurement, the framework is represented as a Cartesian graph, Fig. 1, where the ordered pairs (x, y) relate classification, on the x-axis, to framing, on the y-axis. To facilitate its understanding and implementation, the graph is divided into four quadrants and an artefact is pointed out as an example for

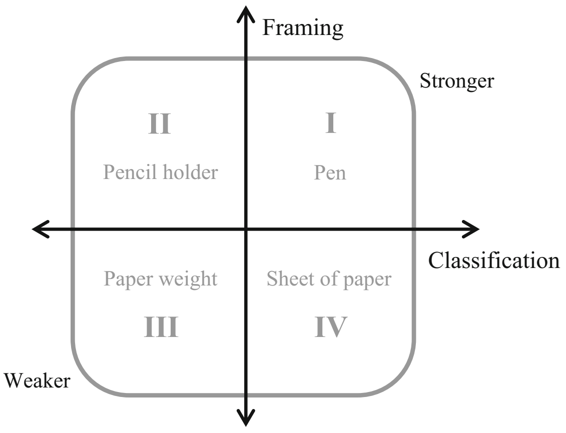


Fig. 1 Visual representation of the proposed framework on affordances

each one. Thus, *quadrant I* refers to stronger classification and framing, e.g., a pen, *quadrant II* refers to weaker classification and stronger framing, e.g., a pencil holder, *quadrant III* refers to weaker classification and framing, e.g., a paper weight, and *quadrant IV* refers to stronger classification and weaker framing, e.g., a sheet of paper.

Classification and framing are referred to as *affordance dimensions*, because they not only relate to each other, but also represent quantitative values, which are derived from the aforementioned elements. Note that classification and framing are *stronger* or *weaker* to rebut the *exists*/*does not exist* dualism of affordances in favour of attention towards the degree to which an affordance is perceived.

With regard to the examples provided and the way they relate to the affordance dimensions, an ordinary pen is supposed to be in quadrant I because, regardless of the context (stronger classification), its own constraints, to some extent, do not allow different uses other than its expected use (stronger framing), which is *writing*/*drawing on a surface*; a pencil holder is supposed to be in quadrant II because it may afford something different from its expected use, if it is found next to a sink or on a kitchen bench (weaker classification), although being constrained by its own properties and behaviour (stronger framing); a paper weight is supposed to be in quadrant III because it can be interpreted as many things, such as a door holder or a decorative object, depending on the context (weaker classification), and on the agent's knowledge and motivations (weaker framing); a sheet of paper is supposed to be in quadrant IV because its designed purpose is still clear in different contexts (stronger classification), although it is open to a plethora of different uses, e.g., origami, due to its weak intrinsic constraints (weaker framing).

Generally speaking, the resulting framework shows that there are situations in which the contexts, or the artefact itself, i.e., its properties and behaviour, constrain the agent's perception of an affordance strongly. As such, the framework suits cultural paradigms of affordance. Changing the environment of an executive chair from a boardroom to a waiting room weakens the classification by removing the cultural rule that the chair affords sitting for the senior executive. And, then, there are other situations in which the agent's knowledge and motivations constrain affordances. So, the context, or the artefact itself, intrinsically tells the agent much more about what the artefact is capable of *doing*.

Final Considerations

Although we recognise that the proposed framework has not yet been empirically tested yet, which is the next step of our research, in this paper we provided enough arguments, i.e., a dialectical approach, capable of supporting its underlying mechanisms. Although one could argue that without further proof our framework is just a guess, lacking empirical evidence, the framework generates testable hypotheses. One such hypothesis is that strengthening the framing of artefacts reduces the variety of uses to which a human agent associates to an artefact. According to the framework, depending on the situation, when the designer has considerable control over the artefact being designed, i.e., its properties and behaviour, as well as its context, then it is more likely that it will evoke fewer non-designed purposes, because of how the agent, i.e., the end-user, perceives its affordances; otherwise, we predict that the artefact is more likely to produce exactly the opposite effect.

Designing artefacts from an affordance perspective is not a matter of making certain affordances perceivable, e.g., visible, to the detriment of others, as being turned on or off; rather, it is about making them more or less obvious. As previously stated, affordances are not binary; they can be strengthened or weakened, as if they were connected to a 'slider'. Once affordance perception is treated as a problem of information transmission, it is reasonable to consider the importance played by context or the agent's knowledge and motivations; both of them act as a force in favour of, or against, the artefact's designed purpose. Therefore, the proposed framework not only provides some awareness about this subject, but may also help designers to create end-user products that are less error-prone or more open to different assigned purposes.

Given the degree to which potential non-designed purposes may be assigned to an artefact, this paper might serve as a reference for designing towards (1) strong framing, i.e., rigid uses, in which case it has to be ensured that the constraints are perceivable enough, so the agent's understanding over the artefact is not affected, whatever the context or the knowledge and motivations of the agent; or (2) weak framing, i.e., more flexible uses, where affordances are perceived in such a way that the operation and/or the purpose of an artefact are intentionally open to the agent's interpretation, which can be viewed as a process of empowerment of the end-user.

Acknowledgments Leonardo Burlamaqui gratefully acknowledges the financial support of CAPES Foundation, Ministry of Education of Brazil (Grant Process BEX 1079/13-1). Andy Dong is the recipient of an Australian Research Council Future Fellowship (Project No. FT100100376).

References

1. Gibson JJ (1977) The theory of affordances, Perceiving, acting, and knowing: toward an ecological psychology. Lawrence Erlbaum, Hillsdale
2. Gibson JJ (1979) The ecological approach to visual perception. Houghton Mifflin, Boston
3. McGrenere J, Ho W (2000) Affordances: clarifying and evolving a concept. In: Proceedings of Graphics Interface 2000, Montreal
4. Gaver WW (1991) Technology affordances. In: Proceedings of the SIGCHI conference on human factors in computing systems, New Orleans
5. Norman DA (1988) The psychology of everyday things. Basic Books, New York
6. Norman DA (1999) Affordance, conventions, and design. Interactions 6:38–43
7. Maier JRA, Fadel GM (2001) Affordance: the fundamental concept in engineering design. In: Proceedings of the ASME 2001 international design engineering technical conferences and computers and information in engineering conference, Pittsburgh
8. Maier JRA, Fadel GM (2009) Affordance based design: a relational theory for design. Res Eng Des 20:13–27
9. Kirschner PA, Martens RL, Strijbos J-W (2004) CSCL in higher education? A framework for designing multiple collaborative environments. What we know about CSCL and implementing it in higher education. Kluwer Academic, New York
10. Suthers D (2006) Technology affordances for intersubjective meaning making: a research agenda for CSCL. Int J Comput Supported Collab Learn 1:315–337
11. Dohn NB (2009) Affordances revisited: articulating a Merleau-Pontian view. Int J Comput Supported Collab Learn 4:151–170
12. Still JD, Dark VJ (2013) Cognitively describing and designing affordances. Des Stud 34:285–301
13. Lin L, Chen G, Kuang H et al (2007) Neural encoding of the concept of nest in the mouse brain. Proc Natl Acad Sci 104:6066–6071
14. Crilly N, Moultrie J, Clarkson PJ (2009) Shaping things: intended consumer response and the other determinants of product form. Des Stud 30:224–254
15. Dix A, Finlay JE, Abowd GD et al (2003) Human-computer interaction, 3rd edn. Prentice Hall, New York
16. Gero JS (1990) Design prototypes: a knowledge representation schema for design. AI Mag 11:26–36
17. Rosenman MA, Gero JS (1998) Purpose and function in design: from the socio-cultural to the techno-physical. Des Stud 19:161–186
18. Vermaas PE, Dorst K (2007) On the conceptual framework of John Gero's FBS-model and the prescriptive aims of design methodology. Des Stud 28:133–157
19. Gibson JJ, Crooks LE (1938) A theoretical field-analysis of automobile-driving. Am J Psychol 51:453–471
20. Hinze-Hoare V (2007) The review and analysis of human computer interaction (HCI) principles. The Computing Research Repository (CoRR)
21. Bernstein BB (1971) Class, codes and control. Routledge & Kegan Paul, London
22. Bar M (2009) The proactive brain: memory for predictions. Phil Trans R Soc B Biol Sci 364:1235–1243

Diagnosing Wicked Problems

Janet E. Burge and Raymond McCall

Abstract Horst Rittel defined his wicked problems theory to differentiate wicked problems, which are open-ended and controversial, from tame problems, which have a single correct solution. Rittel identified ten properties of wicked problems but did not indicate if all of these properties needed to hold before something could be deemed wicked. Failure to correctly classify a problem as wicked is likely to result in either design failure (if wickedness is not identified and problems are tamed prematurely) or design paralysis (if incorrectly perceived wickedness is used as an excuse for not buckling down to do the work of identifying and implementing solutions). Here we augment Rittel's theory by identifying ten causes of wickedness and describe how those can be used along with his ten properties to identify wicked problems in cases where not all of his properties apply.

Introduction

Horst Rittel introduced the theory of wicked problems to look at: "that class of problems which are ill-formulated, where the information is confusing, where there are many decision makers and clients with conflicting values, and where the ramifications in the whole system are confusing" [1]. In his articles on wicked problems [2, 3]. Rittel describes them as planning problems, but in his two decades of teaching he systematically described them as design problems as well.[1] In this article we attempt to aid designers in diagnosing wicked problems, both in the sense of identifying and classifying problems correctly as wicked and in the sense of identifying the causes of wicked problems.

[1] Author McCall was a PhD student of Rittel

J.E. Burge (✉)
Miami University, Oxford, OH, USA
e-mail: burgeje@muohio.edu

R. McCall
University of Colorado, Boulder, CO, USA

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_18

Rittel lists ten properties of wicked problems [2] that indicate ways in which such problems are inherently open-ended and controversial:

1. There is no definitive formulation of a wicked problem.
   Here by the term *formulation* Rittel means the set of all the information need to understand and to solve the problem. By *definitive* he means exhaustive.
2. Wicked problems have no stopping rule.
   There are no objective criteria for determining when a wicked problem has been solved. Designers can always go on trying to do a better job designing. Design projects end not because of the "logic of the problem" but because of limits of resources such as time, money, manpower. etc.
3. Solutions to wicked problems are not true-false, but good-bad.
   Determining whether a solution is good is a value judgment, not a factual judgment. There are no objective means for making value judgments; so such judgments are often disputed, even by reasonable and informed people.
4. There is no immediate and no ultimate test of a solution to a wicked problem.
   A solution to a wicked problem generates chains of consequences extending far into the future. Since unknown future consequences might outweigh known near-term consequences, the designer never really knows how good a solution is.
5. Every solution to a wicked problem is a "one-shot operation"; because there is no opportunity to learn by trial-and-error, every attempt counts significantly.
   A solution to a wicked problem has irreversible, real-world consequences; so trial-and-error is not ethically defensible.
6. Wicked problems do not have an enumerable (or exhaustively describable) set of potential solutions, nor is there a well-described set of permissible operations that may be incorporated into the plan.
   For wicked problems, it is not possible to prove that any list of solutions is complete. Designers can always try to devise new solution ideas.
7. Every wicked problem is essentially unique.
   No matter how similar a new wicked problem looks to a previous wicked problem, there is no guarantee that the new one will not have unique factors that are of overriding importance.
8. Every wicked problem can be considered a symptom of another wicked problem.
   There are two ways of solving a given wicked problem. One is by solving it directly. The other is by considering it to be a symptom of (caused by) another wicked problem and then solving that other problem.
9. The existence of a discrepancy representing a wicked problem can be explained in numerous ways. The choice of explanation determines the nature of the problem's solution.
   Different people may have different ideas about what causes a given wicked problem and, thus, how to solve it. There are no objective methods for settling such differences of opinion.

10. The designer has no right to be wrong.

    Designers are legally and morally responsible for the consequences of their design decisions, because those consequences take the form of irreversible effects on people.

Wicked Problems exist in contrast with tame problems, those problems that have an enumerable set of objectively testable solutions. A commonly given example of a tame problem would be chess—while winning is difficult, it is not controversial.

It is crucial to note that Rittel is using the term *problem* in a non-traditional way. Traditionally the term *problem* denotes a set of discrepancies between is and ought-to-be, i.e. between the current situation and an improved situation. Rittel, however, uses the term *wicked problem* to refer not only to such discrepancies but also to everything that is problematic about the design project and every type of information useful for the project, including the solution ideas, context of the solution effort, the stakeholders, the solution criteria, and so forth.

The *wickedness* of a design problem lies in the essential impossibility of doing justice to an open-ended and controversial problem with a limited amount of design resources, such as time, money, manpower, creativity, and patience. Rittel seems to believe—though he never states it explicitly—that all ten of the above-listed properties exist for each wicked problem. Inspection of these properties, however, makes it clear that various subsets of them would be sufficient to render a design problem open-ended and controversial. This can make a problem *wicked* from the point of view of the designer, even when that problem does not have all ten of the properties.

Rittel gives a number of examples of Wicked Problems in his writing: eliminating poverty, urban planning (freeway construction and other public works), reducing crime, designing curricula in the schools [2], designing management information systems, or designing product lines [3]. What Rittel does not state clearly is if all his criteria need to be applied before something can be categorized as wicked. This is important because when we look at the use of the term "wicked problem" in the design literature we see that it has been applied to many design problems that fail to have all of Rittel's properties.

While it seems possible that some problems are wicked even though they do not have all ten properties, it is also possible that the term wicked is used in cases where it is not appropriate. There are several reasons for possible over-use of the term wicked problem. Some researchers claim that all design problems are wicked because Rittel reportedly stated "most of the problems addressed by designers are wicked problems" [4]. Others fixate on the "essential uniqueness" criteria and claim that if a problem is unique it must be wicked. Still others confuse difficulty with wickedness and claim a problem is wicked as a reason for being unable to solve it.

The over-use of the term *wicked problem* in design is itself problematic—it is important to correctly identify the presence of wickedness because the methods for solving wicked problems are different from those for solving tame ones. If the term *wicked* is used too broadly, problems will be deemed to be more challenging than they actually are; while failing to recognize a problem as wicked and treating it tame can cause harm since solutions to wicked problems can have irreversible consequences. Correctly identifying a problem as wicked and determining the cause of that wickedness is the first step towards determining how or if it can be solved.

In this paper, we first discuss related work on classifying wicked problems. Then we discuss if all of Rittel's criteria need to hold before a problem can be classified as wicked, define ten cause of wickedness that can be used along with Rittel's criteria to differentiate the wicked from the tame, describe how Rittel's criteria and the cause of wickedness apply to the real-life case study of the Denver International Airport automated baggage handling system, and then conclude with a summary.

Related Research in WP Diagnosis

The term "wicked problem" was introduced to distinguish those that are wicked from those that are tame. So how is this distinction made? For social problems, such as solving poverty, the wickedness is much clearer than for technical problems. There are many examples in the literature where entire domains are identified as wicked. These include software development [5], interaction design [6, 7], communication design [8], Computer Supported Collaborative Work (CSCW) [9, 10], and requirements negotiation [11]. It is likely that many if not most systems in those domains are wicked but there are also probably examples that are tame.

There are cases where problems are diagnosed as wicked based on one criterion. DeGrace and Stahl [12] wrote a book on wicked problems in software engineering but all their examples correspond to only one of Rittel's properties. There are also examples where problems are described as wicked because the developer could not solve them (which does not necessarily mean they are not wicked but could also imply wickedness as an excuse). Researchers and practitioners view wickedness differently—the practitioner is more likely to be frightened away from a problem described as wicked (since it may not be solvable and their job is to create solutions) while a researcher would find working on wicked problems to be appealing, since any work towards a solution may be a novel research result.

While some examples of over-diagnosis seem to exist, there are compelling arguments made for applying subsets of Rittel's criteria. Wicked problems and tame ones can be viewed as opposite ends of a spectrum. Kreuter et al. [13] focused on four aspects as being key in distinguishing between the wicked and the tame. First, there is the issue of defining the problem itself. Tame problems can be clearly defined with equally clear solutions. Wicked problems do not have a single clear definition—different stakeholders may have differing opinions on what the problem actually is. Solutions can only be evaluated relative to each other rather than being evaluated as right or wrong. Second, there is the role of stakeholders. For tame problems, experts can clearly define the causes based on data, while in wicked problems the stakeholders are likely to disagree. The third characteristic is the stopping rule—for tame problems the task is complete when solved, for wicked problems there is no clear completion point and the solving process is more likely to be declared over when resources are depleted or other factors come into play. The fourth characteristic is the nature of the problem itself—tame problems are similar to other, already addressed problems while wicked problems are unique and solutions must always be tailored. Most of Kreuter's criteria can be mapped to Rittel's with the exception of Kreuter's assertion that wicked problems have a complex root issue (Rittel's causal chains are infinite).

Jeff Conklin focused on six wicked problem categories [14, 15]: Not understanding the problem until a solution has been developed, no stopping rule, solutions are not right or wrong, uniqueness of the wicked problem, solutions are 'one-shot', and no set number of alternative solutions. He also defines characteristics of tame problems [14]: well defined problem statements, stopping points, right or wrong solutions, part of a class of similar problems with similar solutions, easily evaluated solutions (which can be abandoned if they fail), and enumerable (and limited) solution alternatives.

The shorter list of properties avoids some of the more problematic and confusing properties, such as the designer not having the right to be wrong and there not being an immediate and ultimate test. Unfortunately this list also fails to highlight the impact of multiple explanations on the wickedness of a problem or the tendency for wicked problems to be symptoms of larger wicked problems.

Camillus [16] also focused on a modified sub-set of the wicked problem properties. His primary focus was on strategic planning. Unlike Conklin, who warns of the dangers of attempting to tame a wicked problem, Camillus feels that wicked problems can be tamed, although not solved. The properties that he uses to define wicked problems are multiple stakeholders with different priorities, lack of a precedent (uniqueness), attempted solutions change the problem, nothing to indicate if an answer is correct or not, and a complex root issue ("complex and tangled"). Table 1 shows how the subsets of properties used by Kreuter, Conklin, and Camillus compare.

Table 1 Subsets of properties used by others

Rittel	Conklin	Kreuter	Camillus
No definitive description	X	X	
No stopping rule	X	X	
No right/wrong solutions	X	X	
No "immediate and ultimate" test			X
"One shot operation"	X		X
No exhaustive list of solutions	X		
Essential uniqueness	X	X	X
Symptoms of larger wicked problems			
Many explanations where explanation choice determines resolution		X	X
"No right to be wrong"		Complex root issue	

Farrell and Hooker [17] take a different approach towards an abbreviated set of criteria. They look at the "cognitive dimension" of wickedness and extract three conditions that alone or together indicate wickedness: finitude (limitations in cognitive capability or resources), complexity (relationships between complex systems causing consequences at different levels), and normativity (value differences between "agents").

The fact that several researchers have opted to work with subsets of the original ten wicked problems suggests that it is worth considering if all of Rittel's properties still apply as stated and if there might be some value in classifying problems as wicked if not all of Rittel's properties are met.

Classifying Problems as Wicked

Rittel's two papers concentrated on defining what a wicked problem is and on presenting properties that distinguish the wicked from the tame. These properties can then be used as criteria for classifying a problem as wicked or not wicked. Problem classification is important because the choice of methods used in problem solving (or as might be more appropriate for wicked problems, problem management) needs to suit the problem type. Overstating the wickedness of a problem may lead to paralysis and needless debate if the stakes of attempting and retracting solutions are not high. Underestimating the wickedness of a problem is likely to result in creating additional problems if solutions are made without appropriate consideration of the consequences, since for many wicked problems consequences of failed solutions may be irreversible (as captured by Rittel's properties 5 and 10). It is not sufficient that a problem be difficult or even that it be part of what is traditionally an ill-defined domain [18] in order to be categorized as a wicked problem. It is not even sufficient for a problem to be unique. A novel problem may be more difficult than a familiar one but may still have a clear solution to be discovered.

In addition to an accurate assessment of the wickedness of a problem, it is also important to add an additional diagnostic step to assess the likely cause or causes of wickedness. Wicked problems are not merely hard—they are problems where there are circumstances that force them into the open-endedness and controversy that categorizes wickedness. Identifying the cause of wickedness is valuable for two reasons: first, a problem that is assessed as wicked where there are not factors indicating why it is wicked may be a challenging but tame problem; second, understanding the causes is a first step in deciding on appropriate strategies to use in taming it.

The question remains—do all of Rittel's properties need to hold before a problem can be deemed wicked? If not, what other criteria can be used to separate the wicked from the tame?

Properties Required

Many of the examples of wicked problems described in the computing literature are selective in using Rittel's properties as criteria to classify problems as wicked. In Rittel's writing, he refers to his properties as "distinguishing properties of planning-type problems, i.e. wicked ones, that planners had better be alert to" [2] but does not explicitly state that a problem can only be deemed wicked if it has *all* the properties. Still, this is implied. Two properties explicitly state that they apply to every wicked problem—essential uniqueness and being a symptom of another wicked problem. The former is clearly essential—if a problem is not unique and has been already solved then it is, by definition, not wicked. The latter is more problematic—it is interesting to note that in Table 1, this is one of the properties that none of the three researchers chose for their reduced set of wicked problem criteria. In his book on wicked problems, Jeff Conklin explicitly states that of the six properties he uses, it is not necessary for a problem to contain all six of the properties to be considered wicked [14].

There are problems that may possess some but not all of Rittel's properties but still require the consideration due to a wicked problem. For example, designing how to land the Mars Rover Curiosity on Mars had many of the characteristics of a wicked problem, such as uniqueness, an inability to enumerate all possible solutions, a one-shot operation, and no room for correction if the planner was wrong (aka, "no right to be wrong"), yet there were others that it lacked—there was an ultimate test for success or failure. No one would categorize landing a spacecraft on another planet to be a tame problem, suggesting that wickedness may live on a spectrum. This suggests that the true value in Rittel's properties are not merely in labeling, but in pointing out where the wickedness and accompanying pitfalls lie. This also suggests that some properties or combinations of properties may be more essential in defining wickedness than others.

Understanding the Cause

In addition to diagnosing a problem as wicked or not, it is also useful to understand what is causing the wickedness. If a problem only meets a subset of Rittel's properties, this additional analysis can help determine if the problem is indeed wicked or if it is simply challenging. Understanding the cause (or more likely, causes) of wickedness can aid in identifying strategies for resolution. There are many possible causes of wickedness (more than one of which may need to exist for a problem to be truly wicked). These are intended to explain the relationship of problems to Rittel's properties, not to replace Rittel's properties.

- *Politically wicked*—these are problems that are wicked because of the amount of conflict between stakeholders with conflicting goals and conflicting beliefs on what would constitute an acceptable solution. If stakeholders cannot agree on what would be considered a solution, grid-lock will result unless a compromise can be reached. Stakeholders also may not agree on what the real problem is, which relates to Rittel's Property 1 of wicked problem, where there is no definitive statement of the problem. It also relates to Property 3, where the value judgments of solutions can lead to debates between stakeholders, and to Property 9, where alternative explanations of causes can lead to similar debates.

- *Financially wicked*—these are problems that are wicked because resources are constrained to a point where solving the problem is not feasible. In some cases, adding resources is not possible because it would take those resources from other areas where they are needed and in some cases the required funds are simply out of the question. This relates to Property 2 of wicked problems, where it could forces an early stop to the design project. On the other hand, it also relates to Property 6, which says that designers can always try to come up with better solution ideas—for a shortage of financial resources might be dealt with by devising new solutions that make better use of those resources. The inability to compensate for financial restrictions is likely to force politically difficult trade-offs between the value judgments of stakeholders that are referred to in Property 3.
- *Temporally wicked*—these are problems where the time required to solve the problem is constrained in a way that makes the solution process more difficult than for other problems of that type. As with financial restrictions, temporal restrictions are likely to force politically difficult trade-offs between the value judgments of stakeholders that are referred to in Property 3.
- *Socially wicked*—these are problems where solving a problem for one part of the affected population may have negative consequences on others. This relates to Rittel's Property 5, where the solution to a wicked problem is a "one-shot operation," as well as Property 10, where designers have "no right to be wrong." And of course, it also relates to Property 3, which deals with stakeholder conflict over value judgments.
- *Scalability wicked*—these are problems where the major issue is that of scale yet the problem cannot be reduced in size or scope. Scalability can refer to the complexity of the design project or to the complexity of the designed artifact. This might cause designers to desperately attempt to devise some way out of this difficulty. They might try to come up with some clever new way of solving the problem, as referred to by Property 6. Alternatively, they might try re-framing the problem in the manner indicated by Property 8, where a problem might be solved by seeing it as a symptom of some other problem. Failing this, they are likely to have to deal with political difficulties with clients, users and other stakeholders—as referred to in Property 3.
- *Environmentally wicked*—these are problems where "solutions" have undesirable environmental consequences. This reason also relates to Properties 5 ("one-shot operation") and 10 ("no right to be wrong").
- *Technologically wicked*—these are problems that require technology that is either nonexistent or unproven. These problems have the potential to be at least partially tamed if innovations occur. Such difficulties could threaten to use up valuable design resources devising new solution ideas for low-tech

workarounds and/or increase the likelihood of failure to meet goals of the project due to failure of the technical effort. The former relates to Property 6 (new solutions always possible), the latter to Property 3 (stakeholder value conflicts).
- *Safety-critical wicked*—these are problems where a primary cause of difficulty is the risk of loss of life. Properties 5 and 10 are also factors here, as is Property 4 (no immediate or ultimate test of solutions).
- *Physically wicked*—these are problems where the physical environment is constraining the solution space in a way that makes this problem uniquely different from similar ones. This relates to Property 7 ("essential uniqueness").
- *Knowledge wicked*—these are problems where there is a lack of expertise and experience that adds to the uncertainty of potential solutions and where there is no way to even attempt a solution without being forced to make assumptions that may not hold true. These may not be true wicked problems—if the lack of knowledge is universal, then wickedness may exist, if the lack of knowledge is localized to the organization with the problem, then the problem can be tamed by finding the right participants. This is relevant to Property 5 ("one-shot operation"), Property 10 ("no right to be wrong"). It is also relevant to Property 4 (no immediate and no ultimate test of solutions).

We propose that if a problem is believed to be wicked but does not conform to all of Rittel's properties, looking to see if any of the above causes are present is a way to distinguish the wicked from the tame. If there is not some outside force on the problem forcing it into wickedness, then the problem may be difficult but tameable.

Case Study: Denver International Airport Baggage Handling

One example of a problem that might be classified as wicked is the unsuccessful design and implementation of an automated baggage handling system for the Denver International Airport (DIA) [19]. DIA was intended to be the largest airport in the country. One of the key goals was to decrease aircraft turnover time, a desirable feature when marketing an airport to the carriers as a potential hub [20]. A key component in this goal was creating a new, fully automated baggage handling system. The system, designed by BAE Automated Systems was intended to have 4,000 baggage carts carry 1,400 bags a minute (as compared with the 100 bags a minute the company's system handled for United Airlines in San Francisco at the time) [21]. There were a number of factors contributing towards project failure, including [19]:

- Deciding to expand a baggage handling system originally planned for only one airline to the entire airport (which should have been done earlier in the process);
- Changing requirements throughout the process (such as realizing that systems in Denver need to handle skis);

- Designing the baggage system after the building was in place (forcing designers to work within physical constraints);
- The death of the Chief Airport Engineer mid-project;
- Technical issues (power delivery, algorithm design); and
- Insufficient time for testing caused by schedule pressure and start-up delays.

The failure to implement the automated baggage handling system was considered to be the main cause of the airport's opening being delayed by 16 months. While a scaled-down version of the automated system was installed, after 10 years of use with a maintenance cost of one million dollars a month, the automated system was scrapped in favor of a standard manual procedure [19].

So was the DIA baggage handling system a wicked problem?

Applying Rittel's Properties to DIA

Most of Rittel's properties apply to the DIA baggage handling system.

Property 1: There is no single "definitive formulation" of a wicked problem.

This property is the most difficult to apply of the ten. The baggage handling requirements did change at several points during the effort when they decided to expand the United Airlines baggage handling to cover the whole airport and also when they added requirements for oversized bags and skis [19]. The property could be considered true for that reason but it is not clear if the criteria would have been true from the start of the project.

Property 2: Wicked problems have no clear end ("stopping rule") to indicate when the problem has been solved.

The question asked by this property is, how do we know when the design is completed and correct? Parts of the system could be given testable requirements but evaluating the system as a whole would require implementation and observation in progress, which would not be helpful in determining when design should be stopped. The reality of this problem was that design stopped when the cost of delay was too high to bear—which matches Rittel's statement that the solution effort on wicked problems stops not because the problem is solved but because the designers run out of resources. The airport eventually opened with a partially manual system, which it used for 10 years before that was scrapped in favor of a fully manual system [19].

Property 3: Solutions are not right or wrong, but instead are good or bad.

In a system as complex as the baggage handling system there were multiple ways to solve each sub-problem, each with tradeoffs. Solutions are most likely to be evaluated by acceptability rather than as being correct.

Property 4: There is no "immediate and ultimate" test that can be applied to a solution to a wicked problem.

The baggage handling system installed in Munich, a much smaller system than this one, required 6 months of 24/7 operation before they could declare it ready for use [23]. The much larger and more complicated system at DIA would have most likely required as much if not more testing. Six months does not count as "immediate." Even after the scaled down system was deployed it took 10 years to decide that it was not worth the maintenance costs.

Property 5: Solutions to wicked problems have irreversible consequences so solving a wicked problem is a "one shot operation."

The inability to solve the problem had irreversible consequences since each day the airport was delayed in opening was costing the city of Denver more than a million dollars [20]. The damage to vendors who had hoped to open in the airport was also significant [22]. The eventual solution, partial automation, was reversible but with the consequence of money lost.

Property 6: It is not possible to exhaustively list all potential solutions to a wicked problem.

The scale of the baggage handling problem and the number of contributing parts would make it impossible to list all possible solutions. BAE built a prototype baggage handling system that worked in its warehouse but that did not translate into a working system at DIA [19].

Property 7: Every wicked problem is "essentially unique."

There were similar systems at Munich and San Francisco but these were much smaller in scale. They also did not have the issues that the Denver system had where the building was already in place prior to design of the automated system. The unique building constraints added significant difficulty to the DIA system. Still, the designers of the DIA system did not study these similar systems; or if they did, they ignored the warnings. The simpler Munich system took the same time allowed for the more complicated DIA system, suggesting that DIA had seriously underestimated the time. Also, the Munich system required 6 months of 24/7 operation to work out bugs [23], something not accounted for in the DIA plans. This is an example where refusing to learn from similar systems really hurt the effort.

Property 8: Every wicked problem can be viewed as a symptom of another wicked problem.

The reason for having the automated baggage handling system was to solve the problem of aircraft turnover time. This is clearly a larger wicked problem—turnover time has many contributing elements including weather and ground crew operations in addition to the baggage handling problem. Turnover time could be considered a factor in trying to run a more profitable airport. Delays in plane turnover time can be viewed as a symptom of having an inefficient baggage handling system, but not vice versa. The inefficiency of the original baggage handling system was initially judged to be a symptom of the fact that it was not automated. This judgment, however, ultimately led to the catastrophic consequences resulting from the failed attempt to automate that system.

Property 9: There are many explanations for the cause of a wicked problem and the one chosen then determines how the problem can be resolved.

The reason for needing the automated baggage handling system was the importance of decreasing turnover time. Aircraft turnover time itself has many contributing factors, of which baggage handling was only one. There is no indication, however, that there was any disagreement about these causes. What Rittel is referring to in Property 9 is not agreement about multiple contributing causes but rather disagreement about alternative explanations as to the cause of a wicked problem; so this property does not appear to be applicable here. If there were disagreements that were never made public, then this property would apply.

Property 10: The planner "has no right to be wrong."

The failure with the baggage handling system delayed the opening of the airport by 16 months, adding 560 million dollars to the cost of the airport [19]. There were numerous lawsuits between the players in this project. United Airlines sued BAE for the system being inadequate; BAE sued United Airlines for failing to make payments. Denver sued BAE and settled out of court for $12,000 for each day late) [22]. Bondholders sued the city of Denver. Denver settled this suit for 4.5 Million [24]. Denver sued Continental airlines for reducing its number of flights (which Continental claimed was due to the baggage system not being ready) [25]. Several small businesses had to "call it quits" because of the delays [22].

Causes of Wickedness at DIA

The DIA baggage handling system can also be analyzed to study the causes of the wickedness:

- *Politically wicked*—yes—there were political pressures to open the airport sooner that caused the vendor to not insist on sufficient testing time as well as pressures from the various stakeholders involved to meet their individual needs.
- *Financially wicked*—there were financial pressures that translated into time pressures. The cost for each month of delay was estimated at US $33 million a month [20].
- *Temporally wicked*—time pressure caused a number of issues and it is suspected that schedule pressures kept appropriate risk management strategies from being put in place [19].
- *Socially wicked*—there is no evidence that this was a factor in this particular problem.
- *Scalability wicked*—the size of this particular baggage handling system compared to similar systems was a major source of difficulty. This system had 14 times the capacity of the San Francisco system, a massive leap in scale [21].
- *Environmentally wicked*—this did not appear to be an issue with the baggage handling although there were likely to be environmental concerns with other aspects of the airport design.
- *Technologically wicked*—there were mechanical and computer reliability issues that may have been due to the difficulty and scale of the problem. There were

also challenges in bag identification. For the DIA system there was first the need to read the destination from RFID chips (with a potential of misreads) and then the transmission of this information by radio to the baggage carts [20].
- *Safety-critical wicked*—risk of life was not an issue with this system (although safety of property was—early tests sent baggage carts and baggage flying [21]).
- *Physically wicked*—the baggage handling system had to work within the already built infrastructure of the airport. The tight turns that needed to be made were a major challenge. This was exacerbated by adding the requirement to install a maintenance track [19].
- *Knowledge wicked*— there were significant algorithmic challenges in the "cascade of queues" that needed to be managed when more than 100 lines needed to be fed into each other. Line balancing is needed so that all lines (and flights) get equally good service so that all aircraft connections are successful [20].

So is the baggage handling system a wicked problem? We would argue that even though it is not clear that all of Rittel's properties apply that yes, the baggage handling system is a wicked problem.

Summary and Conclusions

Rittel's theory of wicked problems makes a critical contribution to design theory by pointing out the types of problems where even stating the problem clearly is an issue. Correctly diagnosing these problems is critical since the implications of failed solutions can be irreversible and impact a wide range of stakeholders. For detecting wickedness, we do not believe that it is necessary that all of Rittel's properties hold for a problem to be considered wicked. Instead, the properties should be used along with an assessment of the cause of the problem's wickedness to distinguish problems that are inherently wicked and those that are technically difficult due to a lack of designer experience or knowledge.

Acknowledgements This work was supported by NSF CAREER Award CCF-0844638 (Burge). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

References

1. Churchman CW (1967) Wicked problems. Manag Sci 14(4):B-141–B-142
2. Rittel HWJ, Webber MM (1973) Dilemmas in a general theory of planning. Pol Sci 4:155–169
3. Rittel HWJ (1972) On the planning crisis: systems analysis of the 'first and second generations'. Bedrifts Økonomen 8:390–396
4. Buchanan R (1992) Wicked problems in design thinking. Des Issues 8(2):5–21
5. Matheson F (2006) Designing for a moving target, NordiCHI'06

6. Dalsgaard P, Halskov K (2012) Reflective design documentation. In: Proc. of the designing interactive systems conference
7. Wolf T, Rode J, Sussman J, Kellogg W (2006) Dispelling design as the black art of CHI, In: Proc. of the SIGCHI Conf. on human factors in computing systems
8. Mehlenbacher B (2009) Multidisciplinarity and 21st century communication design, In: Proc. if the Int. Conf. on design of communication
9. Baharin H, Muhlberger R, Loch A (2009) Mutuality: a key gap in the move to Telecare, In: Proc. of the 10^{th} Int. Conf. NZ Chapter of ACM's SIG on HCI
10. Fitzpatrick G (2003) The locales framework: understanding and designing for wicked problems. Kluwer Academic Publishers, London
11. Liang P, Avgeriou P, He K, Xu L (2012) From collective knowledge to intelligence: pre-requirements analysis of large and complex systems, In: Proc. of the 1st Workshop on Web 2.0 for Software Engineering
12. De Grace P, Stahl L (1990) Wicked problems, righteous solutions: a catalogue of modern software engineering paradigms. Yourdon Press, Englewood Cliffs
13. Kreuter MW, De Rosa C, Howze EH, Baldwin GT (2004) Understanding wicked problems: a key to advancing environmental health promotion. Health Educ Behav 31(4):441–454
14. Conklin J (2005) Dialogue mapping: building shared understanding of wicked problems. Wiley, New York
15. Conklin J (2009) Building shared understanding of wicked problems, Rotman Magazine, Winter
16. Camillus JC (2008) Strategy as a wicked problem. Harv Bus Rev 2008:99–106
17. Farrell R, Hooker C (2013) Design, science and wicked problems. Des Stud 34:681–705
18. Lynch C, Ashley K, Pinkwart N, Aleven V (2009) Concepts, structures, and goals: redefining ill-definedness. Int J Artif Intell Ed 19(3):253–266
19. Calleam Consulting (2008) Case Study—Denver International Airport Baggage Handling System – an illustration of ineffectual decision making, Calleam Consulting Ltd – Why Technology Projects Fail. http://calleam.com/WTPF/wp-content/uploads/articles/DIABaggage.pdf
20. de Neufville R (1994) The baggage system at Denver: prospects and lessons. J Air Transp Manag 1(4):229–236
21. Myerson AR (1994) Automation off course in Denver, New York Times, March 18
22. Kerzner H (2012) Project management case studies. Wiley, Hoboken
23. Schloh M (1996) The Denver International Airport automated baggage handling system. Cal Poly, Feb 16, 1996
24. Feder B (1997) Denver bond underwriters settle suits for $4.5 Million. New York Times, May 27
25. Paulson SK (1995) City of Denver Sues continental airlines, Associated Press

On Abduction in Design

Ehud Kroll and Lauri Koskela

Abstract The mechanism of design reasoning from function to form is addressed by examining the possibility of explaining it as abduction. We propose a new interpretation to some definitions of innovative abduction, to show first that the concept, idea, as the basis for solution must be present in the inference, and second, that the reasoning from function to form is best modeled as a two-step inference, both of the innovative abduction pattern. This double-abductive reasoning is shown also to be the main form of reasoning in the empirically-derived "parameter analysis" method of conceptual design. Finally, the introduction of abduction into design theory is critically assessed, and in so doing, topics for future research are suggested.

Aim

Better understanding of the reasoning mechanism from function to form is the main aim of the current investigation. In previous work [1] we studied the method of parameter analysis (PA) from the perspective of the proto-theory of design, which is based on the method of geometric analysis, as suggested by Aristotle. It was concluded that certain design "moves" could be explained as being deductive, some as regressive, but others were more difficult to cast in this logic-based framework and were characterized as being compositional or transformational/ interpretational. Regressive and transformational inferences were of particular interest as they involved heuristic reasoning and intuition, notions that are sometimes associated with the type of inference called abduction. For example, abductive reasoning has been identified with the notions of intuition [2, p. 33],

E. Kroll (✉)
Technion – Israel Institute of Technology, Haifa, Israel
e-mail: kroll@technion.ac.il

L. Koskela
University of Salford, Salford, UK

Aalto University, Espoo, Finland

creativity and subconscious activities [3]. Of course, abduction has for long been discussed in philosophy of science.

The structure of the current paper is therefore as follows. After a brief discussion of how the core of all design processes—reasoning from function to form—is treated in some of the design literature, the notion of abduction in science and design is introduced. Then we examine two papers that seem central in this area, by Roozenburg [4] and Dorst [5]. Roozenburg's ideas on innovative abduction are presented and analyzed both theoretically and through his kettle design example. Next we study Dorst's model of abduction and apply it to the same example. We propose a modification of both Roozenburg's and Dorst's models and relate it back to PA. The paper concludes with critical overall assessment of the introduction of the concept of abduction into design theory, and in so doing makes some suggestions for future research.

The paper intentionally looks at the "mechanics" of the reasoning involved in creating design solutions in terms of the relevant patterns of inference and what are the constituent entities in each inference. The important cognitive issues of what drives the inference and where ideas come from are briefly touched but not systematically investigated. In addition, the activity of evaluating proposed solutions is also left out, as its nature is very different from the so-called "synthetic" activity of generating something new.

How Does Form Follow Function?

The design principle of form follows function is widely accepted. But how is form inferred from function? Ullman [6, p. 140] suggests that this is done by a double mapping process: first from function to concept, and then from concept to form. Many interpretations of this sequence are possible. The German-school systematic design (e.g., [7]) and variations on this approach that appear in many design textbooks (including Ullman's) prescribe a comprehensive functional decomposition stage, followed by finding working principles (concepts) for the various subfunctions and combining them into an overall concept (the principal solution), and finally, detailing the form (layout) of the concept in the so-called "embodiment design" stage. The working principles usually consist of "physical effects + form", while the principal solution is defined as an idealized representation of the structure that defines those characteristics that are essential for the functioning of the artifact [4].

Suh's axiomatic design framework consists of the functional space and the physical space [8]. The former contains functional requirements (FRs) and the latter, design parameters (DPs). Mapping FRs into DPs is the core of the design process; but because there can be many mapping techniques, the design axioms provide the principles to be satisfied by the mapping to produce good designs. Suh does not expand on how solution concepts or ideas are generated; rather, he uses

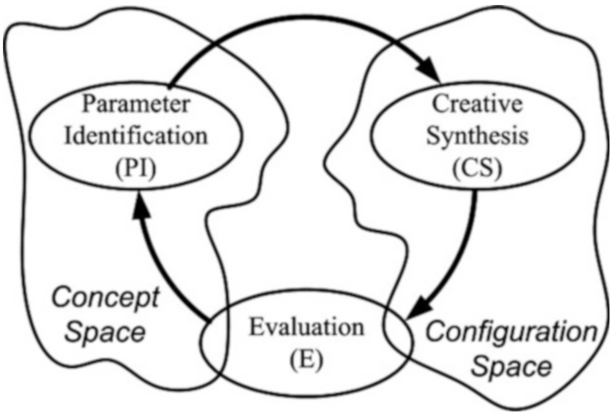


Fig. 1 The parameter analysis process consists of repeatedly moving between concept space and configuration space by applying parameter identification (*PI*), creative synthesis (*CS*) and evaluation (*E*)

many examples of design problems and their solution ideas to support the notions of FRs, DPs and the two axioms.

Another framework, function-behavior-structure or FBS [9], identifies several processes within design, such as transforming functional requirements into expected behaviors, transforming expected behaviors into solution structures, deriving actual behaviors from structures, comparing derived with expected behaviors, and responding to unsatisfactory behaviors by reformulating the design space (changing the structure, behavior or function). However, this descriptive model does not use the notion of 'concept', in the sense of underlying solution ideas, as an explicit constituent of the design space.

Parameter analysis, PA [10, 11] models the design process differently: it uses function, concept (idea) and form as entities of Ullman's double mapping but at a "smaller scale". Instead of corresponding to whole stages of the design process, PA's prescription consists of repeatedly moving between concept space and configuration space, using three distinct activities that are repeated many times (Fig. 1), and in this sense it is similar to the FBS model. The first step, *parameter identification* (PI), corresponds to finding a "parameter" (concept, idea) for resolving a functional issue with the evolving design. This concept is mapped into form by the *creative synthesis* (CS) step, and the last configuration is tested by an *evaluation* (E) step. This last step often results in new functional issues (unsatisfactory or undesirable behavior) to be resolved, so the process continues until a satisfactory solution has been reached.

PA is unique in that it places the most emphasis on the PI step. The reasoning at the conceptual level is claimed to be so important, that "parameters"—ideas, concepts, operating principles, underlying physical effects, etc.—have to be stated explicitly. The E step is considered second in importance, because it involves abstracting from a particular problem to new functional issues at the conceptual level. The actual step of giving form to the design, CS, is ranked the least important, as intermediate configurations are needed mostly to facilitate the evaluation, and any unsatisfactory characteristic of a configuration will be mended in the next cycle. So, although the outcome of the design process is certainly a configuration, the philosophy of PA is that the reasons, justifications and derivations behind the configuration are indispensable when it comes to presenting a design solution or studying the process of designing.

The following examples, although being just fragments of long conceptual design processes, demonstrate the nature of reasoning in PA:

Example 1, from [1]

Deceleration of airborne sensors, dispensed in large numbers from an aircraft for measuring air properties in the airspace over an area, was needed. Three consecutive PA cycles (rephrased for clarity and omitting calculations and sketches for brevity) were:

PI:	Produce a large drag force by using the technology of flexible parachutes
CS:	A 150-mm dia. canopy made of thin sheet material and cords to suspend the sensor
E:	The canopy may not open because the "pull" is too weak and cords may tangle
PI:	Overcome the mid-air opening and cord tangling problems by using a rigid parachute
CS:	A 150 × 150-mm square base pyramid with the sensor attached to it rigidly
E:	Compact packaging of the devices is impossible
PI:	Allow compact packaging by using an umbrella-like folding structure
CS:	Lightweight skeleton with hinges, slides and spring, and thin sheet canopy stretched over it
E:	Unreliable because of the many moving parts and costly to make

Each PI step contains a function to be satisfied and a concept, solution principle, to do that. The function in the first cycle originated from analyzing the design task, but consecutive functions emerge as a result of previous evaluations. The PI step therefore represents a reasoning step from *function* to *concept*. CS generates a description of a configuration that realizes the last concept, which is *form* of course, and E derives the behavior of the configuration and points the functional direction for the next PI.

Example 2, from [11]

The design process of a sensitive tiltmeter [12] for measuring changes of the ground angle with respect to the local gravity vector was described. A novel configuration emerged, with one regular pendulum coupled mechanically to an inverted pendulum. At this point the configuration was evaluated and the process continued as follows:

E:	Friction in the rotational joints must be reduced to ensure the required sensitivity
PI:	Minimize friction by using rolling contact instead of sliding
CS:	Flexural hinges of the appropriate design with near-zero resistance
E:	The displacement measurement should also be made without friction
PI:	Minimize friction by using a non-contact sensing technology
CS:	A capacitor-type sensor

The first E step above is taken from the middle of the design process, where a new functional issue has come up. Reasoning from this *function* to a *concept* follows in PI and from the *concept* to *form*, in CS. Evaluation of the last configuration reveals a new problem, so another design cycle begins.

A Brief Introduction to Abduction in Design

An ongoing debate exists in the philosophy of science and other areas regarding the nature of abductive reasoning. Peirce [13] is attributed with proposing that abduction is a form of "synthetic" reasoning (together with induction, but different from the "analytic" reasoning of deduction), while focusing on scientific explanation. Researchers still disagree on the exact nature of induction [14] and certainly on abduction. Schurz [15] presents a thorough classification of abduction patterns, all of which are "special patterns of inference to the best explanation". He identifies many types of abduction based on three dimensions. The main dimension is the type of hypothesis (conclusion) abduced. The other two are the type of evidence to be explained and the cognitive mechanism driving the abduction. Schurz refers to "the official Peirce abduction schema" as "factual abduction" of the following structure:

$$\begin{array}{l} \textit{Known Law}: \text{IF}\, Cx\, \text{THEN}\, Ex \\ \textit{Known Evidence}: Ea\, \text{has occurred} \\ \text{---} \\ \textit{Abduced Conjecture}:\; Ca\, \text{could be the reason} \end{array} \tag{1}$$

Investigations of abduction in relation to design have mostly been carried out by scholars in design theory and artificial intelligence. Both streams of research are briefly outlined in the following.

In design theory, March [16] seminally suggests that abduction, which he calls "productive reasoning", is the key mode of reasoning in design. He also points to the confusion and misunderstanding created by not distinguishing between scientific and design hypotheses, and between logical propositions and design proposals. Whereas the goal of science is to establish general laws, he says, design is concerned with realizing a particular outcome. The pattern of abduction proposed by March is: from certain characteristics that are sought, and on the basis of previous knowledge and models of possibilities, a design proposal is put forward.

Roozenburg [4] discusses in depth the question whether the reasoning towards a tentative description of a design follows the conventional view on abduction, or whether it should be defined differently. He argues that the commonly presented view, especially in artificial intelligence literature, deals with "explanatory abductions", which are good for diagnosis or troubleshooting, but that the core of design reasoning follows another type of abduction, for which he proposes the terms "innovative abduction" and "innoduction" [17]. In fact, says Roozenburg [4], Habermas distinguished between explanatory abduction as in (1) and innovative abduction, in which the law is not known and needs to be inferred together with the presumed reason for the evidence, and it was March who did not make that distinction.

A more recent paper by Dorst [5] proposes yet another view on design abduction. It claims that there are two types of abduction relevant to design: abduction-1 which follows a similar pattern to (1), and abduction-2 which is comparable to Roozenburg's innoduction. Furthermore, Dorst suggests chaining these two inferences into a single reasoning step, which is the core of 'design thinking'.

In artificial intelligence oriented research on design abduction, the emphasis has been on computable abduction models. To some extent this work is overlapping with and influenced by design theory research on abduction. For example, Goel [18] proposes to extend (and complicate) March's model if we wish to use it in knowledge-based systems. His argument is based on the fact that the laws (also called rules or knowledge) can have different logical natures; for example, universal or statistical, and this affects the meaning of the abduction pattern. However, the influential work led by Takeda et al. [19] on design abduction is based on original insights into design, and the connection to Peirce's seminal work on abduction in science seems looser. Abduction is defined as a process making integrated hypotheses and theories to explain given facts [20]. This definition goes beyond Schurz' classification of abduction [21]. Analogical reasoning is applied for computationally supporting abduction [22].

Alone due to space reasons, the focus of this paper is on the design theory oriented research on abduction. However, the artificial intelligence oriented abduction research is touched in the concluding section on future research needs.

Analysis of Roozenburg's Model of Innovative Abduction

Roozenburg [4] says that *explanatory abduction*, also called *presumption of fact*, which seems to him as the prevailing view on abduction, is a reversal of deduction, and <u>is not</u> the main reasoning mechanism in design. Deduction is:

$$\begin{array}{ll} p \rightarrow q & \text{(a given rule, IF } p \text{ THEN } q\text{)} \\ p & (p \text{ is a given fact, a case or cause}) \\ \hline q & (q \text{ is the conclusion, the result}) \end{array} \tag{2}$$

Reversing it, we get the following pattern:

$$\begin{array}{ll} p \rightarrow q & \text{(a given rule, IF } p \text{ THEN } q\text{)} \\ q & (q \text{ is a given fact, a result}) \\ \hline p & (p \text{ is the conclusion, the case or cause}) \end{array} \tag{3}$$

This is the definition of explanatory abduction, similar to (1). According to Roozenburg, pattern (3) is not the main reasoning form in design, where in fact the only given is a desired result, and both the rule and the cause need to be discovered. His *innovative abduction* therefore follows the pattern:

$$\begin{array}{ll} q & (q \text{ is a given fact, a desired result}) \\ \hline p \rightarrow q & (\text{a rule to be inferred first, IF } p \text{ THEN } q) \\ p & (p \text{ is the conclusion, the case, that immediately follows}) \end{array} \tag{4}$$

Pattern (4) is the real abduction in design because it represents reasoning from a desired result, a purpose or function, to *form* and *use*. Form and use are the 'principal solution', the structure of the artifact and its way of use that define its function.

Roozenburg demonstrates the above through the following example of designing a kettle. The purpose, *function*, is to boil water. The *mode of action* (defined as 'using laws of nature to produce a desired effect'), or functional behavior, is heating the bottom of the kettle and conducting the heat to the water inside. This will be facilitated by the *way of use* (also called 'actuation') of filling the kettle with water and placing it on a burner. Finally, to allow all this, the kettle must have a specific *form*: hemisphere with opening at the top and metal construction.

Now that there are <u>four</u> distinct entities involved in the reasoning (*function*, *mode of action*, *way of use*, and *form*), Roozenburg groups together *form* and *way of use* into one entity, claiming that they always go hand in hand, so he writes:

$$\textit{form} + \textit{way of use} \rightarrow \textit{mode of action} \rightarrow \textit{function} \tag{5}$$

or in other words: hemisphere and metal + fill with water and place on burner → heat bottom of kettle and conduct heat to the water inside → boil water.

Next, the intermediate result (*mode of action*) in expression (5) can be omitted, so what is left is:

$$\textit{form} + \textit{way of use} \rightarrow \textit{function} \tag{6}$$

or: hemisphere and metal + fill with water and place on burner → boil water.

The *function* (boil water) is given in design, says Roozenburg. What needs to be designed is usually considered to be the *form* (hemisphere and metal). But a description of *form* is not enough to predict the behavior which fulfills the *function*. The behavior (*mode of action*) depends on *form* but also on the *way of use*. So, the designer needs to develop ideas on *way of use* together with *form*. It follows that the "kernel of design" is the reasoning from *function* to *form* + *way of use*. This, according to Roozenburg, follows the same pattern of reasoning as Habermas' innovative abduction, expression (4), if we define p as the combined description of *form* + *way of use*:

q boil water (the only given is the *function*)

$p \rightarrow q$ IF hemisphere and metal + fill water and place on burner THEN boil water(IF *form* + *way of use* THEN *function*; the rule to be inferred first) (7)

p hemisphere and metal + fill water and place on burner (*form* + *way of use*; the second conclusion)

The meaning of the last logical derivation is that if you want to boil water, you need to 'discover' the first conclusion (hemisphere and metal *form* + filling water and placing on burner *way of use* → boil water *function*), and immediately you will get the second conclusion (hemisphere and metal *form* + filling water and placing on burner *way of use*). The second conclusion constitutes the principal solution to the design problem.

Critique of Roozenburg's Model

The question regarding Roozenburg's presentation is whether the designer who wants to boil water can generate the 'rule' in the first conclusion directly, without reasoning about the *mode of action* (heating the bottom of the kettle and conducting the heat to the water inside) first. Roozenburg's description does not include the *mode of action* explicitly, assuming perhaps that somehow the designer has gained the insight on using the specific *mode of action*, which is the main characteristic of the principal solution, and now proceeds according to pattern (7). In other words, we could modify Roozenburg's presentation of abduction to the following (where the underlined addition of the *mode of action*, the operating principle, makes it explicit):

q boil water $\underline{\text{by heating the bottom of a container and conducting the heat to the water}}$ (*function* and *mode of action*)

$p \rightarrow q$ IF hemisphere and metal + fill water and place on burner THEN boil water $\underline{\text{by heating the bottom of a container and conducting the heat to the water}}$ (the first conclusion)

p hemisphere and metal + fill water and place on burner (the second conclusion)

(8)

But this raises two new questions: (a) where did the mode of action come from in the first place, and should it not be an explicit abductive step by itself in the description of the "kernel of design"? and (b) does pattern (8) represent what really happens during design?

To answer these questions, let us try to imagine the thought process while designing a kettle. We need to design a device to boil water (but in a certain context, of having at our disposal a burner, and the boiled water will be used to make tea, as opposed for example to generating steam in a sauna). What operating principle can we use? Here is an idea: we need some sort of container that can be filled with water and placed over the burner. Then the bottom of the container will be heated, and the heat will be conducted to the water inside (note that we came up with a *mode of action* – heating the bottom of the water container and conducting the heat to the water, and *way of use* – filling the container with water and placing it on the burner). Now that we have decided on these (*mode of action* + *way of use*), we ask ourselves what *form* we should give the device to work properly (that is, a *form* that when used as intended – filled with water and placed on burner – will result in the intended *mode of action*, conducting the heat to the water). The answer now is, use a hemisphere with opening at the top and make it out of metal.

The reasoning above is clearly from *function* to *mode of action* + *way of use* first, followed by reasoning from *mode of action* + *way of use* to *form*. Roozenburg represents this process as a single innovative abduction, wherein the *mode of action* is implicit, so it gives the impression that the main idea (*mode of action*) is not part of the abduction at all. Moreover, Roozenburg combines *way of use* with *form* into a single entity, as if they are inseparable.

A more correct way to represent the above reasoning process may be by a two-step or double innovative abduction to capture the fact that two distinct inferences are carried out:

1st step:

q boil water (the *function*)

$p \rightarrow q$ IF will water and place on burner so heat is conducted to water THEN boil water (the first conclusion: *way of use+mode of action → function*) (9)

p fill water and place on burner so heat is conducted to water (the second conclusion: *way of use+mode of action*)

2nd step:

q fill water and place on burner so heat is conducted to water (the newly generated *way of use+mode of action* is now the given)

$p \rightarrow q$ IF hemisphere with opening and metal THEN fill water and place on burner so heat is conducted to water (the first conclusion: *form → way of use+mode of action*)

p hemisphere with opening and metal (the second conclusion: *form*)

(10)

To summarize, the above two-step reasoning allows inferring from *function* to an idea, concept or solution principle (showing as *way of use* + *mode of action*) first, and from that principle, to the *form*. In general we can say that each innovative abduction reasoning step of pattern (4) involves two entities, *p* and *q*, but design reasoning should involve four entities: *function*, *mode of action*, *way of use*, and *form*. And although we claim that *mode of action* and *way of use* seem to frequently show together, so they can be counted as one entity, the three remaining entities still require two inferences, not one. What Roozenburg did is actually leaving out *mode of action* and grouping *form* and *way of use* into one entity, claiming that together they are the sought solution, so he could reduce the problem to a two-entity single abduction.

Support for the insight that four entities should be involved in describing design reasoning can be found in the work of Zeng and Cheng [23], which Roozenburg claims arrived at similar conclusions to his. Zeng and Cheng argue that design reasoning involves three entities: *form*, *function* and *environment*, and that the *environment* consists of two entities: *laws of nature* and *actions of nature*. If *laws of nature* are Roozenburg's *mode of action*, and *actions of nature* are his *way of use*, then we have a one-to-one correspondence of the four entities.

Analysis of Dorst's Model of Double Abduction

Dorst [5] explains 'design thinking' as double abduction, assisted by 'frame creation', which itself is facilitated by 'theme exploration'. His presentation of abduction revolves around the following logical expression:

$$\textit{what}(\text{the artifact}) + \textit{how}(\text{the working principle}) \rightarrow \textit{value}(\text{aspired}) \tag{11}$$

in which the (aspired) *value* is always given. If the *how* is also given, we generate the *what* by a so-called *abduction-1*, which is precisely the *explanatory abduction* of pattern (3). Dorst calls this case "conventional ('closed') problem-solving that designers often do". If, however, the *how* is not given, then we have a more 'open' problem in which we need to decide on both the working principle and the artifact. This is accomplished by *abduction-2*, as in pattern (4), which is the same as Roozenburg's innovative abduction. *Abduction-2* is carried out by first developing or adopting a 'frame' (after Schön), which is a "general implication that by applying a certain working principle we will create a specific value". Interestingly, Dorst characterizes the framing activity as being "a form of induction", because it is reasoning back from consequences (this is in conflict with Peirce to whom that kind of reasoning represents abduction). With the help of framing, *abduction-2* takes place according to the following pattern:

$$\begin{array}{ll} q & (q \text{ is the given desired } \mathit{value}) \\ \hline p \rightarrow q & (\text{IF } \mathit{how} \text{ THEN } \mathit{value}, \text{ the first conclusion}) \\ p & (\mathit{how}, \text{ the second conclusion}) \end{array} \tag{12}$$

When a possible or promising frame has been proposed and the *how* is known, says Dorst, *abduction-1* can take place to design the *what*, the artifact.

Critique of Dorst's Model

Let us now apply Dorst's double-step reasoning process (*abduction-2* followed by *abduction-1*) to Roozenburg's kettle example. Surely, the *value* in expressions (11) and (12) corresponds to *function*, and the *what* in (11) corresponds to *form* (Dorst calls it the 'object' or 'thing'). The *how*, therefore, must stand for the *way of use* + *mode of action* (also to be in agreement with Zeng and Cheng [23] on having four entities involved in design reasoning). If we set *value* = "boil water" as the only known fact, *abduction-2* may yield a possible working principle, a *how*, which is the following *way of use* + *mode of action*: "fill water and place on burner so heat is conducted to water". So far this is identical to expression (9).

Now we need to design the *what*, or *form*, and Dorst suggests that this will be done by *abduction-1* because we know the *value* and *how* in expression (11). For *abduction-1* to take place according to pattern (3), however, the conclusion should appear as the premise of the given rule, and this does not seem to be the case here. The *what* is still unknown, and of course this is why this kind of *explanatory abduction* cannot be the main form of reasoning in design. The only possibility is to use *abduction-2* again, starting with the only known, the *how* found in the previous step, and using it as the given to seek a "rule" to tie together a *what* (*form*) to this *how* (working principle), and therefore inferring that *what*. The resulting inference is identical to expression (10).

The Double Innovative Abduction in Parameter Analysis

Having modified Roozenburg's and Dorst's models of reasoning from *function* to *form* to two *innovative abduction* (or *abduction-2*) inferences, as in (9) and (10), allows us to compare this model with PA. As explained and demonstrated earlier, PI is reasoning from a functional aspect to a solution principle, which is equivalent to the first innovative abduction as in (9). The solution principle (concept) consists of way of use + mode of action. The second step is CS, where the reasoning begins with the solution principle derived in PI and ends with a configuration, structure, or form, as in (10). Overall we obtain the double mapping *function* → *concept* → *form*.

The examples of PA described earlier can easily be presented as such double abductions. For instance, the first cycle of Example 1 will be:

1st step, *PI*:

q produce a large drag force (the *function*)

$p \rightarrow q$ IF the sensor is suspended by cords from a flexible parachute THEN a large drag force will be produced (the first conclusion: *way of use+mode of action* → *function*) (13)

p suspend the sensor by cords from a flexible parachute (the second conclusion: *way of use+mode of action*)

2nd step, *CS*:

q suspend the sensor by cords from a flexible parachute (the newly generated *way of use+mode of action* is now the given)

$p \rightarrow q$ IF a 150-mm dia. canopy made of thin sheet material and cords THEN the sensor will be suspended from a flexible parachute (the first conclusion: *form* → *way of use+mode of action*)

p a 150-mm dia. canopy made of thin sheet material and cords (the second conclusion: *form*)

(14)

Conversely, we can formulate the double abduction of the kettle example, expressions (9) and (10) as PA:

PI:	Boil water by filling water in a container and placing it on a burner so the heat is conducted to the water.
CS:	Hemisphere with opening and made of metal.
E:	...

Discussion

Reasoning from *function* to *form* may be productively modelled in terms of two creative leaps, each requiring an *abduction-2/innovative abduction* reasoning step. The first infers the working principle to be used to attain the desired function, and the second infers the artifact that can utilize the working principle. The pattern of abductions involved is very different from *explanatory abduction*, so having a special name for this kind of reasoning seems justified.

Working principle or concept comprises of *way of use* + *mode of action*. The *mode of action* is much more fundamental to the reasoning than the *way of use*. In fact, *way of use* may be trivial in many cases, so it may not appear in the description of the inferences. Deployment of the sensor suspended from the parachute in Example 1 and letting them both descend slowly is the obvious *way of use* in the overall setting of the design task. The *way of use* of filling water and putting the water-filled kettle over a burner is also trivial, because the initial problem statement should have involved a burner as the source of thermal energy (and not, for instance, electricity) and the purpose of boiling the water (for making tea we may want to contain the boiled water, as opposed to producing steam in a sauna).

The importance of explicitly including the *mode of action* in the inference cannot be overstated. When the designer thinks in conceptual terms about physical and working principles, the designed artifact will be based on a solid ideational foundation. Alternative working principles may be thought of, the rationale of the design will be captured better for possible use in the future, and deeper understanding of the problem domain will be gained by the designer. For example, the choice of metal construction in the form of the kettle may be modified according to the *mode of action* of heating the bottom and conducting the heat to the water inside; perhaps by looking for materials with high thermal diffusivity or combining a heat conducting material for the bottom and a heat insulating material for the sides of the kettle.

Dorst [5] specifically refers to this issue. When describing the pattern of *abduction-2* as in (11) he says: “students and other novice designers can be seen to almost randomly generate proposals for both the ‘how’ and the ‘what’, and then seek to find a matching pair that does lead to the aspired value”. In our experience, the issue is not the random trial-and-error process, but rather an attempt to reason from *function* (aspired *value*) directly to *form* (the *what*), without the intermediate step of reasoning about the *concept* (the *how*).

Having proposed a double *innovative abduction*/*abduction-2* model, we may ask whether *explanatory abduction*/*abduction-1* exists in design at all. While March and some other researchers seem to refer to only this type of abduction in the context of design, we have shown that both generating a concept (working principle) and an artifact (form) require abductive reasoning with only one fact, the *desired value*, as a given. In both cases a rule needs to be inferred first, and the premise of the rule immediately follows. The two inferences do not share the same *desired value*: when generating a working principle, the value is the *function*; when generating the form, the value is the *working principle* of the previous step. However, we can imagine situations where the working principle is taken as a given, resulting in abduction of pattern (3) occurring. These seem to be cases in which the problem situation is so familiar to the designer that the working principle is taken for granted and becomes implicit in the reasoning. For example, a structural engineer who regularly designs apartment buildings may specify an I-section (*form*) for the ceiling-support beam (implied *function* of carrying bending loads) directly, without consciously thinking of the working principle of increasing the section's second moment of area by placing most of the material away from the neutral axis.

But the above argument does not necessarily imply that *innoduction*/*abduction-2* occur only in innovative design situations. Pattern (4) of reasoning, in which the 'rule' part (be it *concept* → *function* or *form* → *concept*) is not considered a given, can in fact take place in two very different circumstances. First, in the more routine design situations, many applicable 'rules' may exist in the designer's repertoire, and the abductive step is required to select among them. For example, this may apply to the ceiling-support beam case, when the design requirements are slightly changed and the designer recalls *form* → *concept* rules concerning also C-sections and rectangular-tube sections. Magnani [24] has called this kind of inference, where one selects from a set of known rules, *selective abduction*. Second, in what may be termed "innovative design" situations, the 'rule' simply does not exist (either in the particular designer's mind, or universally) and needs to be 'discovered'. For example, if the ceiling-support beam is required to also provide an easy or aesthetic connection to glass walls, the designer may invent a new section shape that is different from 'standard' or existing shapes. Inference of a new *concept* → *function* rule seems even more innovative, as it implies discovering a new working principle to satisfy a function. Consider for example the first time houses were built out of shipping containers, or the still-futuristic concept of getting to space with an elevator.

As a conclusion, we propose here to modify the general model of design reasoning from *function* to *form* to the following two-step inference of the innovative abduction type that explicitly includes the *concept*, working principle, in it:

1st step:

$$\begin{array}{ll} q & \text{given: } \textit{function} \\ \hline p \to q & \text{first conclusion: IF } \textit{concept} \text{ THEN } \textit{function} \\ p & \text{second conclusion: } \textit{concept} \end{array} \tag{15}$$

2nd step:

$$\begin{array}{ll} q & \text{given: } \textit{concept} \\ \hline p \to q & \text{first conclusion: IF } \textit{form} \text{ THEN } \textit{concept} \\ p & \text{second conclusion: } \textit{form} \end{array} \tag{16}$$

Additionally, we showed how the *parameter identification* and *creative synthesis* reasoning steps in the conceptual design method called "parameter analysis" correspond to the above two steps.

Critical Assessment and the Way Forward

Clearly, this is not intended to be the definitive and complete treatment of abduction in design. Just as understanding of abduction in philosophy and other areas still evolves, researchers in design have to develop further understanding of this fundamental notion. In doing so, problems originating both from understanding of abduction in science, and from the adoption of abduction in design have to be overcome. In general, while especially March' and Roozenburg's treatments of abduction can be considered seminal and have stimulated further research, they leave room for several critical remarks. These are not meant to downplay the value of the early treatments but rather emphasize the generative value of them.

The central motivation for defining abduction, from Aristotle to Peirce, has been to cover for logical inferences that cannot be classified as either inductions or deductions. However, this demarcation is made challenging by the situation that still it is not at all clear what induction is, as stated by Vickers [14]: "attempting to define induction would be more difficult than rewarding". Further, Vickers contends that there is no comprehensive theory of sound induction, no set of agreed upon rules that license good or sound inductive inference, nor is there a serious prospect of such a theory. That induction is not a settled concept makes it indeed difficult to gauge what is outside induction and deduction.

However, there is more to abduction than revealed in logical analysis. Already from Peirce onwards, abduction has been connected to intuition and creativity. There has been much research on these two phenomena as such, but there seems to have been very little scholarly attention specifically on the creative and/or intuitive aspects of abduction. These connections need to be cultivated and expanded for added understanding. Indeed one question is whether we need to set criteria regarding or at least acknowledge its intuitive and creative character when defining abduction. In recent literature, Hoffman [25] seems to have moved into this direction. In this context, two further questions arise: Is all creativity in science or design channeled through abductive inferences? Is creative abduction always based on intuition?

With its origin in the scientific method, the main type of abduction has generally been identified as backwards (regressive) reasoning, essentially through guessing, from consequences to hypothetical causes (in opposition to induction and deduction). In design, regressive and deductive inferences along means-ends hierarchies are prominent forms of reasoning. However, there are also other mental moves, such as decomposition and composition, as well as transformation [26]. Can we recognize cases in these other design moves that are in essential respects similar to abduction, that is, creatively pinpoint a solution candidate or at least the direction to it? This important question is closely related to the call for classification of different types of design abduction, to be presented below.

In discussions on abduction in philosophy of science, there is a fixation to the syllogistic form of abduction, although already Peirce [13] downplayed syllogism as "the lowest and most rudimentary of all forms of reasoning". Schurz [15] cogently argues that there exist rather different kinds of abduction patterns; while some of them enjoy a broad discussion in the literature, other important patterns have been neglected. This fixation to the syllogistic form of abduction has been inherited to treatments of design abduction. The far more common way of conceptualizing design as moves along means-ends hierarchies [27] is rarely analyzed from the perspective of abduction. To the same effect, Niiniluoto [28] discusses the foundational role geometrical analysis has played as a model of reasoning in science, covering also abductive inferences in that analysis. However, the philosophical discussions on abduction rarely acknowledge this. The same complaint can be presented regarding the literature on design abduction.

The generic juxtaposition of the terms explanatory abduction and innovative abduction, as suggested by Roozenburg (under influence from Habermas), is not the best possible, as in science all abductions target explanation. The terms *selective abduction* and *creative abduction*, suggested by Magnani [24], are better in this respect, although as Magnani himself concedes through his examples, the borderline between these is fluid.

Roozenburg, and also we in this paper, use the hypothetical example of design of a metal kettle in the imagined situation that the world would be otherwise as it is now but the kettle would never have been invented. This raises the question how kettles have actually been designed? Before metal kettles, ceramic kettles were used [29], thus metal kettles probably have emerged just through switching over to metal as material. Although schematic examples are often good for purposes of presentation and demonstration, the advancement of scientific understanding on abduction requires the examination of abduction-like inferences in design as they occur in practice. Perhaps, in this way, a thorough classification, as done by Schurz [15] for scientific abductions, could be carried out for design abductions. Interestingly, already the work of Takeda et al. [22] has challenged the completeness of Schurz' classification from a design viewpoint. The attempt of Ullah et al. [30] to connect the notion of "classical abduction" as in (3) to the C–K Theory of design is another example of research endeavoring to interpret abduction from a design viewpoint. They conclude that conceiving a creative ("undecided" relative to existing knowledge) concept is more complex than abduction, being a motivation-driven process. Motivation here consists of a "compelling reason"—why a certain concept is pursued, and an "epistemic challenge"—seeking new knowledge.

Finally, as discussed above, design abduction research falls into two main fields, design theory and artificial intelligence. While these two literatures are partially overlapping, there seems to be room for greater mutual awareness as well as for better synthesis of results.

The multitude of problems related to abduction in general and specifically to design abduction may initially seem overwhelming. However, they all give direction for future research, relevant for the advancement of the field.

References

1. Kroll E, Koskela L (2012) Interpreting parameter analysis through the proto-theory of design. In: Proceedings of the International Design Conference (DESIGN 2012), Dubrovnik, 21–24 May 2012
2. Cross N (2006) Designerly ways of knowing. Springer, London
3. Dew N (2007) Abduction: a pre-condition for the intelligent design of strategy. J Bus Strategy 28(4):38–45
4. Roozenburg NFM (1993) On the pattern of reasoning in innovative design. Des Stud 14:4–18
5. Dorst K (2011) The core of 'design thinking' and its application. Des Stud 32:521–532
6. Ullman DG (1992) The mechanical design process. McGraw-Hill, New York
7. Pahl G, Beitz W, Feldhunsen J, Grote KH (2007) Engineering design: a systematic approach, 3rd edn. Springer, London
8. Suh NP (1990) The principles of design. Oxford University Press, New York
9. Gero JS, Kannengiesser U (2004) The situated function-behaviour-structure framework. Des Stud 25:373–391
10. Kroll E, Condoor SS, Jansson DG (2001) Innovative conceptual design: theory and application of parameter analysis. Cambridge University Press, Cambridge
11. Kroll E (2013) Design theory and conceptual design: contrasting functional decomposition and morphology with parameter analysis. Res Eng Des 24(2):165–183
12. Li YT (1976) Sensitive tiltmeter. US Patent 3,997,976
13. Peirce CS (1994) Collected papers of Charles Sanders Peirce. Electronic edition. Volume 7: Science and Philosophy
14. Vickers J (2013) The problem of induction. The Stanford Encyclopedia of Philosophy (Spring 2013 edition), Zalta EN (ed), http://plato.stanford.edu/archives/spr2013/entries/induction-problem/
15. Schurz G (2008) Patterns of abduction. Synthese 164:201–234
16. March L (1976) The logic of design and the question of value. In: March L (ed) The architecture of form. Cambridge University Press, Cambridge, pp 1–40
17. Roozenburg NFM, Eekels J (1995) Product design: fundamentals and methods. Wiley, Chichester, Chapter 4
18. Goel V (1988) Complicating the 'logic of design'. Des Stud 9(4):229–234
19. Takeda H, Veerkamp P, Tomiyama T, Yoshikawa H (1990) Modeling design processes. AI Mag 11(4):37–48
20. Takeda H (1994) Abduction for design. In: Gero JS, Tyugu E (eds) Formal design methods for CAD. North-Holland, Amsterdam, pp 221–244
21. Tomiyama T, Takeda H, Yoshioka M, Shimomura Y (2003) Abduction for creative design. In: Proceedings of the ASME 2003 Design Engineering Technical Conference (DETC'03), Chicago, pp 543–552, 2–6 Sept 2003
22. Takeda H, Sasaki H, Nomaguchi Y, Yoshioka M, Shimomura Y, Tomiyama T (2003) Universal abduction studio–proposal of a design support environment for creative thinking in design. In: Proceedings of the 14th International Conference on Engineering Design (ICED 03), Stockholm, 19–21 Aug 2003
23. Zeng Y, Cheng GD (1991) On the logic of design. Des Stud 12:137–141
24. Magnani L (1995) Creative processes in scientific discovery. Eur J High Ability 6(2):160–169
25. Hoffman M (1999) Problems with Peirce's concept of abduction. Found Sci 4:271–305
26. Koskela L, Codinhoto R, Tzortzopoulos P, Kagioglou M (2014) The Aristotelian proto-theory of design. In: Chakrabarti A, Blessing L (eds) An anthology of theories and models of design. Springer, London

27. Hughes J (2009) Practical reasoning and engineering. In: Meijers A (ed) Philosophy of technology and engineering sciences, A volume in handbook of the philosophy of science. Elsevier, Amsterdam, pp 375–402
28. Niiniluoto I (1999) Defending abduction. Philos Sci 66:S436–S451
29. Wu X, Zhang C, Goldberg P, Cohen D, Pan Y, Arpin T, Bar-Yosef O (2012) Early pottery at 20,000 years ago in Xianrendong Cave, China. Science 336(6089):1696–1700
30. Ullah AMMS, Rashid MM, Tamaki J (2012) On some unique features of C–K theory of design. CIRP J Manuf Sci Technol 5:55–66

Situating Needs and Requirements in a Multi-stakeholder Context

Gaetano Cascini and Francesca Montagna

Abstract This paper proposes a model aimed at representing the cognitive processes occurring in a design task while analysing needs and requirements with respect to the different stakeholders who might influence the purchase decision. The model builds on Gero's situated FBS framework, by introducing two original elements: from the one hand, needs and requirements are explicitly represented, thus differentiating from the Function, Behaviour, Structure variables. On the other hand, the external world is split into several ones, each representing the reality of a different stakeholder. Thanks to this integration, the earliest stages of the design cycle can be investigated in details by observing, representing and analysing the designer's behaviour through his elementary thinking processes. The ex-post analysis of a brainstorming session within an Italian company producing lines for bottling beverages clarifies the structure of the modelling approach and its benefits.

Introduction

The essential motive of this paper is that most design methods assume that the starting point is a correct design specification, and support "making things right", while they miss to define how to "make the right thing".

Below such superficial statement, a much denser world of issues and criticalities exists. Actually, the innovation process is considerably more complex than simply developing a product or making sure that a single buyer will buy a widget from a seller. In fact, after the purchasing decision, the product must be actually put to use (i.e., adopted) in order to deliver its benefits and the sale of the expected volume of products is not instantaneous, but typically follows a diffusion process [1]. In this other process, at any point in time, the actors who have not adopted widgets yet are usually influenced by the actors who have successfully done so, either because of

G. Cascini (✉)
Politecnico di Milano, Milan, Italy
e-mail: Gaetano.cascini@polimi.it

F. Montagna
Politecnico di Torino, Torino, Italy

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_20

direct “word of mouth”, or because of simple observation of the benefits. Moreover, products and services are seldom aimed to a single actor [2]: buyers and users are not necessarily the same person; the actor(s) that will ultimately benefit from the widget might be different from either the buyer or the user. Moreover, buyers can also be influenced by other stakeholders, such as installers or vendors, etc. configuring what is usually named the externality phenomenon [3].

Many examples can be made to clarify these concepts. Airplanes are bought by the purchasing office of the flight company, are used by the pilots and attendants, and the direct beneficiaries are the passengers. Besides, one might also include maintenance crews, or airport personnel in the analysis and possibly also the inhabitants of the urban areas around the airport. The same concept applies also to much simpler devices: for instance in a hospital, tools such as syringes, thermometers etc. are usually applied to a patient by a nurse, which received medical instructions by a doctor. A patient is surely the proper beneficiary of such a product, whose efficacy directly interests his parents, which could in turn influence the product usage process in a hospital but, in particular, the purchase process in a pharmacy. However, the capillary usage of a biomedical product is strongly determined by decisions of the corporate governance, so in turn of the hospital purchase department. Similarly, looking at the machinery tool industry, the presence of diverse potential actors is again evident. Some of them, such as machine workers, maintenance technicians, factory supervisors are involved, since directly interact with machines. Others (such as plant managers, product designers, etc.) are involved because their decisions affect the production process. This leads the machinery designer to investigate together design issues such as the machine flexibility, as well as other elements more related to the industrialization (and hence to the design) of the parts to produce. Moreover, for instance, requirements for a machine tool have obviously to consider problems related to the use and the specific needs of the operators.

In general, as shown in Fig. 1, it is possible to conceive at least three situations beyond “use”: purchase, delivery of benefits and creation of further impact or

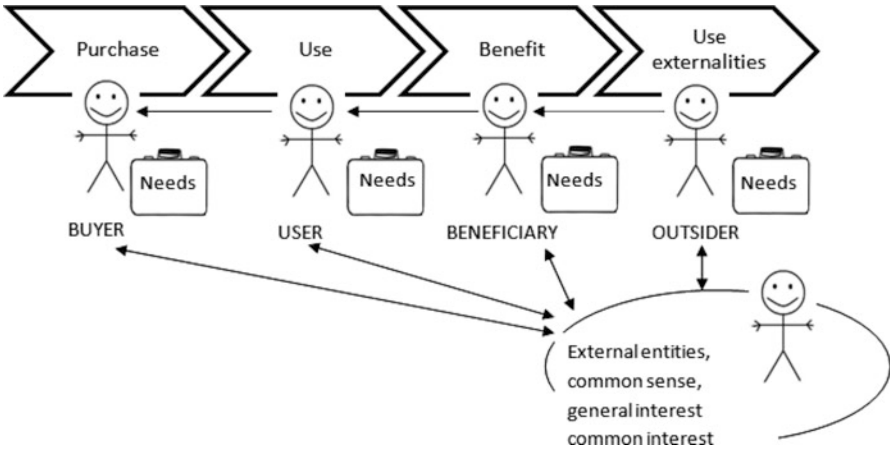


Fig. 1 The multi actor context of beyond use situations

externalities. The proposed set of situations is not exhaustive, but it represents a good balance between simplicity and representativeness. Of course, and as already mentioned, each further "beyond use" situation leads to the need of including more stakeholder roles, and namely buyers, users, beneficiaries and outsiders. For certain products, some actors may obviously coincide.

Each of the involved stakeholders operates according to a set of specific needs. These needs can be native, in the sense that derive from the actor itself, or can result by influences cast among actors and are reported [4, 5]. The influence cast by an actor on another actor can lead to a reported need, if that need would not have been considered by the actor before, as well as it can modify the importance or the perception that an actor assigns to a native need.

Some needs are well known to everyone, either because they are obvious, or elicited by external entities (e.g. regulatory institutions), or because they reflect common sense or general interest. Referring to the airplane example, it is obvious to assume that – all the rest staying the same – the management of the flight company will prefer a product that minimizes discomfort to the people living close to airports, even without receiving direct influence from them. However, the importance that management will assign to this need may be altered if citizens do cast such an influence (e.g. through their representatives in the regional administration), or if by purchasing less noisy or polluting airplanes the flight company might be able to get reduced fares by the local airport. For the design team that is defining the product specification, the ability to understand and proactively work on these influences is integral to the design of the product itself, as well as to the definition of its go-to-market strategy.

Two consequences for designers result. The former is that designers must consider a wider set of needs as the basis for the requirement definition. The latter is that designers must investigate the mutual influences among the actors and their impact on needs. The analysis of the interactions among needs was proposed in [4, 5], and it is not the focus of this paper, though being object of research at the moment. The paper, instead, aims at building a model suitable to represent, and therefore study, the reasoning behind the identification of needs and the formulation of requirements in a multi-stakeholder context. In this perspective, it proposes an extension of the situated FBS model [6], characterized by two main differences. First, it explicitly situates needs and requirements, as originally proposed in [7] and overviewed in the next section, thus suggesting further elementary processes that characterize a complete design activity. Moreover, the external world is analysed in the perspective of a multi-stakeholder context, which is the main original contribution of this paper. In turn, this allows considering the multifaceted interactions between stakeholders that characterize an adoption process; as such, it aims at better understanding the reasons behind success and failure of innovative projects.

The integrated model is presented in the third section, whereas the following one shows its application for the *a posteriori* analysis of a brainstorming session within a product planning activity in the field of aseptic filling for bottled beverages. The discussion of this case study allows highlighting the potential applications of the proposed model, as well as its advantages and limitations.

When the FBS Model Situates Also Needs and Requirements

The situated Function-Behaviour-Structure (FBS) model [6] is a reference model to describe design processes and tasks. Several papers have been written about Gero's framework since its first formulation in [8] and the scientific debate has raised diverse observations and contributions about it, as in [9]. Among them [7], proposes an extended FBS framework where Needs and Requirements are situated as distinct elements into the Gero and Kannengiesser model. This because the authors are persuaded of the theoretic value of that descriptive original framework, as well as are convinced that the requirements definition in that model appears to be too simplistic with respect to the relevance that the requirements have to make design innovative [4, 5, 10].

Marketing literature has been suggesting for some time that the definition of design requirements cannot be made without understanding customer needs (e.g. [11, 12]). Nevertheless, in addition to the marketing literature, many other fields, including engineering design (e.g., [13–16]) architecture and industrial design (since Alexander's seminal contributions in 1964 [17], up to Krippendorff [18]) provide evidence that firms should focus on customer needs for innovating. Topics such as 'co-design' or 'user-centred design' are evidence of that, and emotions, psychological issues or emotional designs have become recognised issues in design broadening the 'customer need' concept (e.g., [19, 20]).

Customer Needs are the basic motivation for purchasing products [21, 22]; they can be derived from customers' inputs, or postulated by the designer. The former category of needs involves explicit requests by the customers, while the needs postulated by the designer are more typically related to technology-driven artefacts. The talent to identify tacit needs (as made for the Geox shoes), or to envision future needs induced by new products (as made by Apple) sometimes is determinant to define the success of a novel product. Therefore, neglecting needs in design can actually kill the innovation process even in the case of products technologically advanced.

In addition to Needs, Requirements are viewed as 'structured and formalised information about a product', 'consist of a metric and a value' [23] and can be viewed as the translation of needs into 'product design specifications', which constitute the reference for the development of a not-yet-designed product.

In engineering design, these two concepts are operatively used day by day and several approaches that address Needs Identification or Requirements Definition have been proposed in the literature (for a survey see [24]) and implemented in industry. However, a proper difference between Needs and Requirements is not really recognized [9, 23] and consequently a clear distinction in the orientation between methods that address Needs Identification or those that face with Requirements does not exist too.

Starting with these considerations, the extended FBS framework [7] situates within the original Gero's model, an explicit representation of Needs and Requirements. This allows the proper means to model the entire product development

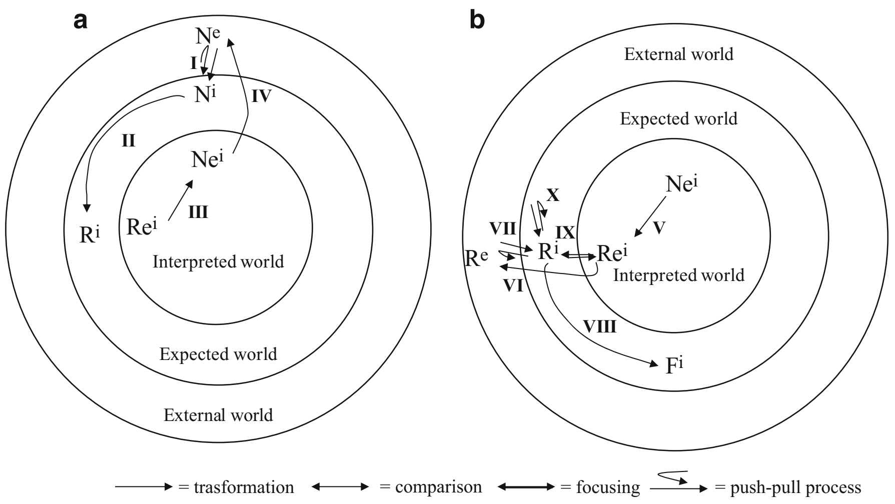


Fig. 2 Extended FBS model [7]: needs identification (**a**) and requirements definition (**b**)

process from the earliest stages and to study the occurring cognitive processes with a more comprehensive and rigorous model.

The extended FBS framework in particular, as shown in Fig. 2, adds two classes of variables (Needs, N and Requirements, R) to those proposed by Gero's model, Function (what an artefact is for), Behaviour (what it does) and Structure (what it is). By "situating" these two variables in the three worlds of the FBS framework (thus producing the followings: N^e, N^i, Ne^i, R^e, R^i and Re^i), the formulation phase is hence reviewed in those processes which definitively substitute the direct processes from the external word of R variables to the interpreted word of F^i, S^i, and B^i variables (e.g. Processes 1, 2, 3 and 18), leading to the definition of two further phases: Needs Identification and Requirements Definition.

As shown in Fig. 2a, the proposed extended model describes Needs Identification as constituted by three elementary processes:

- Process I: customer needs N^e are investigated, thus producing N^i variables (interpretation).
- Process II transforms N^i into R^i variables (transformation). These R^i are a preliminary set of requirements, useful to better categorize the gathered needs.
- Process III transforms the initial expected requirements Re^i into Ne^i variables (transformation). This step ensures that needs not provided by customers, but necessary, have the chance to be taken into consideration
- Process IV transforms Ne^i into N^e variables (transformation) to validate the expected requirements with the customers. In case of negative feedback, the emerging external needs N^e can be analysed and interpreted to reformulate the requirements (through processes I and II).

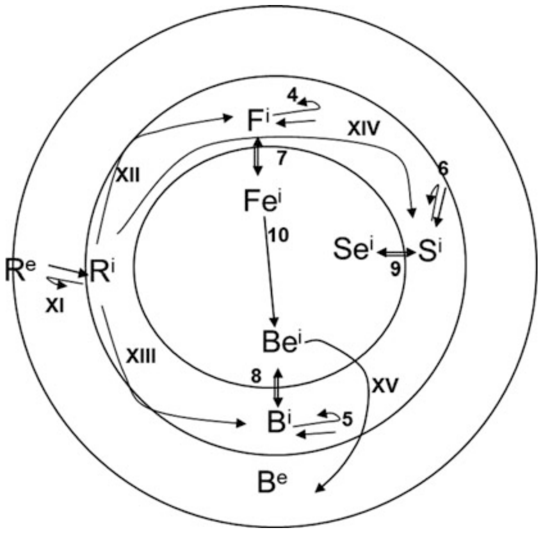


Fig. 3 Extended FBS model: the revisited formulation step

The Requirements Definition process (Fig. 2b), instead, starts from the complete list of expected needs Ne^i previously identified and it is made up of:

- Process V transforms the initial expected needs Ne^i into a first complete set of Re^i (transformation).
- Process VI expands the Re^i set into a bigger or equal number of R^e variables (transformation).
- Process VII uses R^e and results in their interpretation R^i, often through the constructive memory.
- Process VIII transforms the subset of R^i implying an active role of the product, i.e. not related to design constraints, into F^i variables (transformation).
- Process IX focuses on a subset ($Rei \subseteq Ri$) of Ri to generate an initial requirement state space (focusing).
- Process X uses constructive memory to derive further R^i.

By considering these two steps as integrated in the Gero's FBS model, the Formulation step proposed by Gero changes at least for those parts that involve Requirements. Changes, as shown in Fig. 3, consist in those processes which definitively substitute the direct processes from the external word of R variables to the interpreted word of F^i, S^i, and B^i variables:

- Process XI reuses R^e to obtain definitive R^i (interpretation). This step is oriented to the creation of definitively validated interpreted Requirements that can be correctly elaborated.
- Processes XII, XIII, XIV transform R^i into F^i, S^i and B^i variables, respectively (transformation). These F^i are not a preliminary set of functions anymore, but

constitute, together with the other interpreted variables S^i and B^i, a detailed comprehensive set of design variables.

- The other processes from the fourth to the tenth are kept as in the original model.

Actually, one could consider this proposed extension unwieldy, because of the higher number of variables and processes with respect to the original framework; however, both theoretical reflections and industrial experiences made the authors persuaded about the importance of investigating Needs Identification and Requirement Definition with proper modelling means. This because only by analysing in detail these crucial design activities and recognizing the real importance of these in generating innovation, it becomes possible to conceive, test and refine suitable design guidelines for properly interpreting and addressing users' needs. An example application of the extended FBS framework to review and improve an existing design methodology focused on user-device interaction has been published in [25].

A Multi-stakeholder Proposal of the FBS

Considering a multi-stakeholder context within a design activity means hence considering the different perspectives of the diverse actors that share a certain degree of relationship with the object of design. Each actor has own needs not necessarily overlapped with each other. Therefore, in the logic of the FBS framework, it means that different external worlds must be taken into account, each representing the domain where to situate one of the actors' needs. This is the first real change with respect to the original FBS model. As shown in Fig. 4, there will be one external world for the buyer, one for the user, etc. and the individual needs will be situated in the corresponding worlds.

Interpreted and Expected worlds instead are proper of the designer's mind. They are hence multiple if diverse designers are considered, while the various expected (or interpreted) worlds are merged together in a unique expected (or interpreted) world, if one considers a single designer case or supposes that designers working together do share their expectations and interpretations. The former case is depicted in a simplified way in Fig. 5a. This representation indeed is oversimplified because, in order to maintain the image comprehensible, multiplicity is sketched only for the interpreted world, while the expected world is considered unique, just like in the original FBS model. The latter case is illustrated in Fig. 5b, where the external and the interpreted worlds are considered unique, while multiplicity is considered only for the external worlds. This case represents the case object of this paper, where the focus is on the needs among diverse actors, rather than on the issues related to multi-designers contexts. Furthermore, with the aim of facilitating the readability of the Figure and hence of the represented processes, one can decide to "open" and split the diagram horizontally, minimizing in this way the overlaps among the diverse stakeholders' worlds.

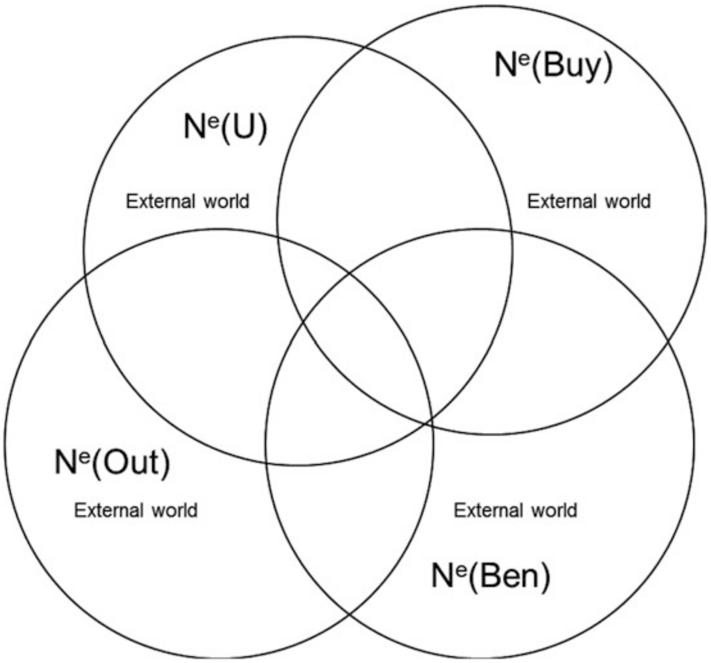


Fig. 4 Situating the external worlds in a multi-stakeholder context

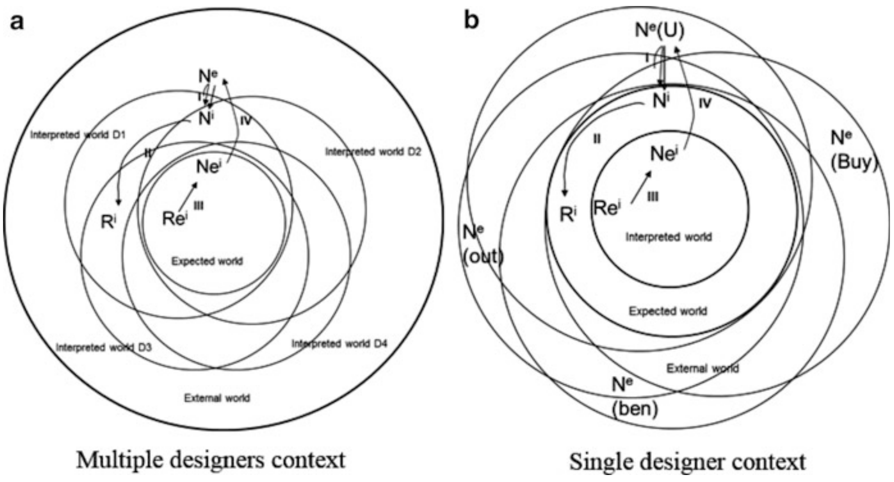


Fig. 5 Situating the need identification phase in a multi-stakeholder context

The uniqueness of the interpreted and expected world is still kept. Obviously, the intersections between the circles of the external worlds can change in relation to the way with which needs are shared among different actors. If some needs are common, they can be represented as belonging to the same intersection set.

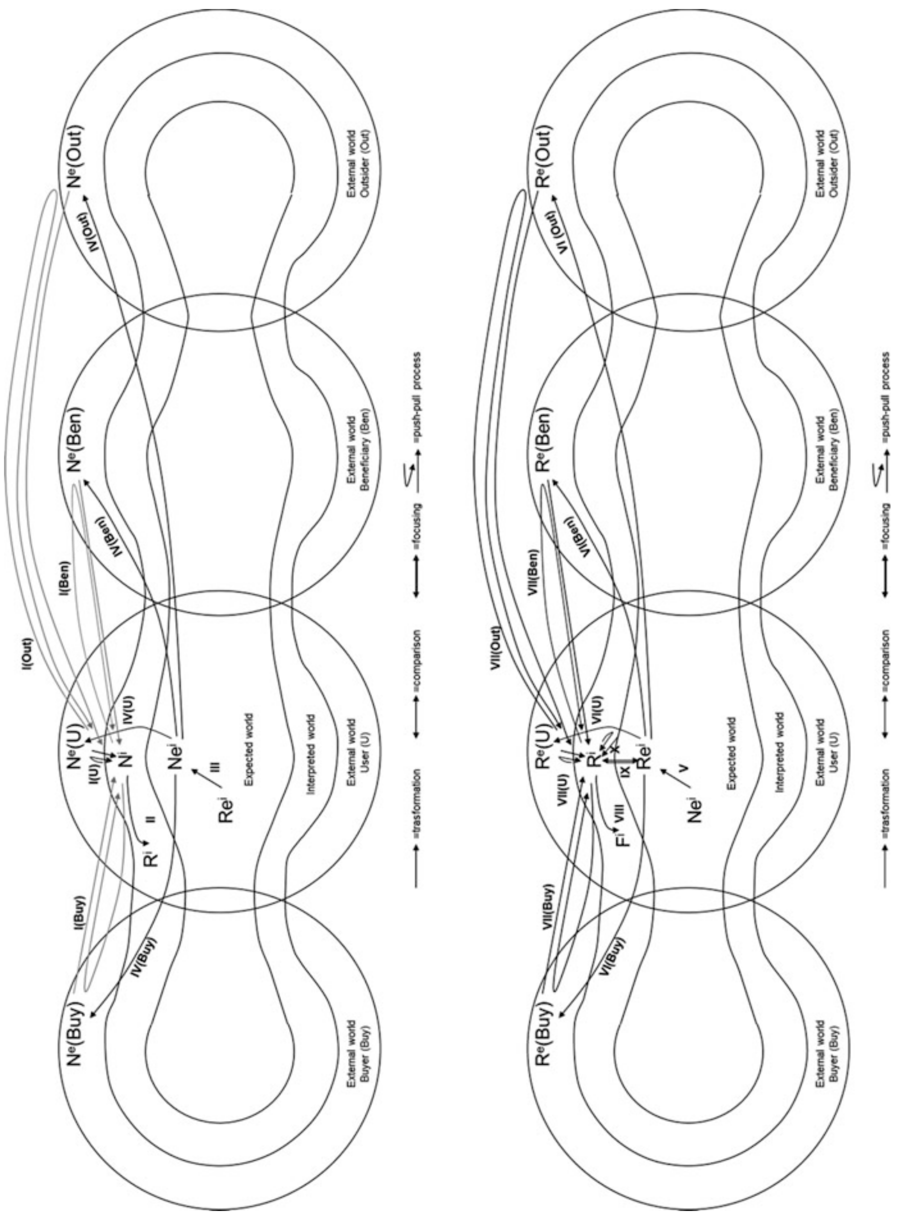


Fig. 6 Splitting need identification (*left*) and requirement definition (*right*) phases

The circle intersection modalities and the analysis of the related cognitive processes are surely worthy of a deeper investigation, but this issue will not be discussed here.

By looking at Fig. 6, which represent the Need Identification (left) and the Requirement definition phases (right) respectively, one can deeply analyse the new processes emerging when a designer considers needs by different actors.

In the Need Identification phase, in particular, this new model modifies the process I where customer needs N^e are investigated and the process IV that validates the expected requirements with the customers. This modification leads to consider one interpretation process for each actor and four further processes eventually appear:

- Process I (Buy): buyer needs N^e(Buy) are investigated, thus producing N^i variables;
- Process I (U): user needs N^e(U) are investigated, thus producing further N^i variables;
- Process I (Ben): beneficiary needs N^e(Ben) are investigated, thus producing further N^i variables;
- Process I (Out): outsider needs N^e(Out) are investigated, thus producing further N^i variables.

The same can be said for the transformation process IV:

- Process IV (Buy) transforms Ne^i into N^e(Buy) variables to validate the expected requirements with buyers.
- Process IV(U) transforms Ne^i into N^e(U) variables to validate the expected requirements with the users.
- Process IV(Ben) transforms Ne^i into N^e(Ben) variables to validate the expected requirements with the beneficiaries.
- Process IV(Out) transforms Ne^i into N^e(Out) variables to validate the expected requirements with the outsiders.

Besides, Processes II and III are situated in worlds that are unique, and hence remain the same.

Similarly, in the Requirement Definition phase, the processes V, VIII, IX, X do not withstand modifications, while the processes VI e VII are distinguished for each actor. Given j = buyer, user, beneficiary, outsider,

- Process VI expands the Re^i set into a bigger or equal number of $R^e(j)$ variables;
- Process VII uses $R^e(j)$ and results in their interpretation R^i.

An Illustrative Case Study

The following case study shows the capability of the proposed model to represent the complex dynamics of interpreting diverse stakeholders' needs. This kind of analysis allows, for instance, to investigate the completeness of the designer's discourse, the relative importance assigned to the different actors and possibly the rigour of the design process itself.

The case study refers about the ex-post analysis of a brainstorming session within an Italian company operating in the aseptic filling sector (i.e. producing lines for bottling beverages needing an aseptic container, such as aseptic carbonated drinks, isotonics, juices, milk based drinks etc.) dedicated to definition of the specification for a new production line. The session involved the R&D&I director of the company, here assumed as the reference designer whose interpretations and expectations are mapped through the extended FBS framework. Two more people attended the session: the sales director, here representing the voice of the Buyer; a senior technician responsible for the set up and the first operation of the line, here representing a User. Since in the discussion FDA (Food and Drug Administration) Standards are often mentioned, it is here assumed that FDA prescriptions represent the existing specification (Re) of further Outsiders' needs, i.e. health and safety issues for the final consumers of the beverages prescribed by a normative body. Thus, with respect to the Fig. 4 scheme, the following actors are taken into account: Buyer, User, and Outsider. Besides, it is evident that the voice of the final Beneficiaries of the design product is not represented and indeed the proposed analysis clearly highlights where this lack is meaningful.

The recording of the session has been parsed into elementary sentences, so as to check the possibility to map each of them through the proposed extended FBS model, by recognizing the nature of the dialogue exchanges in terms of the needs identification and requirements definition processes described above. More in detail, stakeholders' statements have been assumed as expressions of the corresponding external worlds; designer's verbal statements have been considered as representative of the interpreted world, while his written report on the session has been assumed as his expected world. Of course, the aim of this work is not performing a detailed analysis of the design session, even less to infer any conclusion on the design methodological approach of the company, for the evident limited statistical value of the analysed discourse. Nevertheless, the proposed model reveals to be perfectly suitable to map the interactions between the actors, as well as to highlight potentially missing ones, as described hereafter.

First, the analysis has dealt with the identification of the relevant variables associated to the stakeholders' parts of speech (Table 1). Then the variables have been mapped according to the logic followed by the dialogue, so as to recognize the corresponding processes (Table 2).

Once completed, the Tables 1 and 2 with the entire set of statements extracted from the design session, it is possible to review the identified processes, in order to systematically highlight missing links, improper interpretations and implicit assumptions. For example, the request to "Reduce the Total Cost of Ownership", which is supposed to involve both the cost of the machinery and the operating cost, has been interpreted by the designer as an invitation to reduce the operating costs. This interpretation probably involves some implicit assumptions based on the designer's experience about the relative importance of the diverse costs. By mapping this interpretation process through the proposed model, the appearance of some assumptions by the designer becomes evident, thus triggering reflections about their correctness.

Similarly, the lack of a process type II translating the interpreted need "allow machinery inspection without accessing the sterile area" into a requirement, reveals

Table 1 Example (partial) list of variables identified from the analysis of the brainstorming session involving several actors (*Buy* Buyer, *U* User, *Out* Outsider, *D* Designer), classified according to the proposed extended FBS model

Actor	Actor's statement	Associated variable
Buy	Reduce the total cost of ownership	N^e(Buy)
D	Reduce operating costs	N^i
U	Avoid the necessity to wear special protective equipment for machinery inspection	N^e(U)
D	Allow machinery inspection without accessing the sterile area	N^i
D	Electric energy consumption lower than 300 kW for a line producing 36.000 bottles/h	R^i
D	Duration of Cleaning-In-Place (CIP) and Sterilization-In-Place (SIP) processes less than 3 h	Re^i
D	Increase the availability of the line (percentage of scheduled time that the operation is available to operate)	Ne^i
Buy	Improve OEE (Overall Equipment Effectiveness)	N^e(Buy)
D	Caps removal torque (energy to open the bottle) <10 in. lbs.	Re^i
D	Minimize user's efforts to open the bottle	Ne^i
D	The rinsing system must guarantee less than 0.5 ppm peroxides residuals inside the bottle at the end of the sterilization process	Re^i
D	Avoid release of sterilant in the beverage	Ne^i
Out	FDA prescriptions on beverages quality	N^e(Out)
D	Machinery footprint 36.000 bph <1300 m^2	Re^i
Buy	Workshop minimum surface for installing the line 1,300 m^2	R^e(Buy)
D	Bottle type change over <70 min	Re^i
U	Time needed for swapping production type 70 min	R^e(U)
Out	Global Warming Potential (GWP) <0,1 Kg CO_2 eq/L	R^e(Out)
D	Bottle weight (PET) <20 g/l	R^i

that the designer has made some implicit decision about discarding this expectation by the user or the design reasoning has not been completed.

Requirements postulated by the designer such as "Duration of Cleaning-In-Place (CIP) and Sterilization-In-Place (SIP) processes less than 3 h" and "Caps removal torque (energy to open the bottle) <10 in. lbs." are supposed to be checked so as to verify their consistency with real needs. Indeed, during the session, the sales director made an explicit reference to the need of improving the OEE (Overall Equipment Effectiveness) as a confirmation for the suggested increase of the line availability. Besides, the postulated need to "minimize user's efforts to open the bottle", meant as a way to guarantee the capability to open the bottle also by elderly people, did not find any confirmation, i.e. the process IV(Ben) is missing. This consideration might sound obvious, since no representatives of the voice of the beneficiary were present during the session. Nevertheless, the need "avoid release of sterilant in the beverage", postulated through a requirement proposed by the designer, found its correspondence in the FDA standards, here classified as

Table 2 Example (partial) list of processes recognized for the variables of Table 1

Design phase	Process	Actors involved	From (example variables)	To (example variables)
Needs identification	I(Buy): $N^e(Buy)\rightarrow N^i$	Buy→D	Reduce the total cost of ownership	Reduce operating costs
	I(U): $N^e(U)\rightarrow N^i$	U→D	Avoid the necessity to wear special protective equipment for machinery inspection	Allow machinery inspection without accessing the sterile area
	I(Ben): $N^e(Ben)\rightarrow N^i$	Ben→D	Missing	
	I(Out): $N^e(Out)\rightarrow N^i$	Out→D	Missing	
	II: $N^i\rightarrow R^i$	D	Minimize operating costs	Electric energy consumption lower than 300 kW for a line producing 36.000 bottles/h
	III: $Re^i\rightarrow Ne^i$	D	Duration of Cleaning-In-Place (CIP) and Sterilization-In-Place (SIP) processes less than 3 h	Increase the availability of the line
	IV(Buy): $Ne^i\rightarrow N^e(Buy)$	D→Buy	Increase the availability of the line	Improve OEE (Overall Equipment Effectiveness)
	III: $Re^i\rightarrow Ne^i$	D	Caps removal torque (energy to open the bottle) <10 in. lbs.	Minimize user's efforts to open the bottle
	III: $Re^i\rightarrow Ne^i$	D	Less than 0.5 ppm peroxides residuals inside the bottle at the end of the sterilization process	Avoid release of sterilant in the beverage
	IV(Out): $Ne^i\rightarrow N^e(Out)$	D→Out	Avoid release of sterilant in the beverage	FDA prescriptions on beverages quality
Requirements definition	VI(Buy): $Re^i\rightarrow R^e(Buy)$	D→Buys	Machinery footprint (36.000 bph) <1,300 m^2	Workshop minimum surface for installing the line 1,300 m^2
	VI(U): $Re^i\rightarrow R^e(U)$	D→Buy	Bottle type change over <70 min	Time needed for swapping production type 70 min
	VII(Out): $R^e(Out)\rightarrow R^i$	Out→D	Global Warming Potential (GWP) <0,1 Kg CO_2 eq/L	Bottle weight (PET) <20 g/l

Outsider, despite no team members were representing the governmental body for health and sanity.

With a similar approach, it is possible to analyse the session dedicated to the requirements formulation. Tables 1 and 2 report just a few examples of the variables and processes associated to the multi-stakeholder context to illustrate the suitability of the proposed model to map also the following stages of the design process. For instance, it is interesting to notice the different way a requirement related to the size of the machinery is defined by the designer and by the buyer, as well as the time needed for switching the production from one type of bottle to another is expressed slightly differently by designer and user. Moreover, once again the detailed analysis of the occurring processes highlights some reasoning jumps by the designer (due to implicit assumptions made by experience): a limit on the Global Warming Potential is immediately translated into a maximum mass of plastic material that can be used for each bottle, while in principle a wider range of alternatives could be taken into account (e.g. changing material, reducing energy consumption, etc.). It is evident that these implicit assumptions made by the designer speed up the product development process during ordinary design activities. Besides, when the designer is involved in innovative projects, this kind of unaware assumptions can determine psychological inertia and ultimately hinder the generation of innovative solutions.

Conclusions

A key objective of this research is to highlight the importance of dedicating proper consideration to the complementary, possibly conflicting, needs of the diverse stakeholders who can exert an impact on the purchase decision of a product. As briefly discussed in the introduction of this paper, current design methodologies do not take into account the mechanisms behind the adoption of innovative products and services, despite the growing attention to the earliest stages of the product development cycle. Actually, notwithstanding the numerous studies dedicated to the so-called Voice of the Customer, there are no theoretical models suitable to represent the motivations behind the purchase decision and the analyses and choices made by a designer when proposing something new.

In this context, the paper proposes an extension of the situated FBS model aimed at representing the reasoning processes occurring in the earliest stages of product development, when the designer has to properly identify and interpret the requisites and wishes of users and customers, and transform them into a product specification. Beyond the explicit representation of variables describing Needs and Requirements, as already proposed in [7], the extended model suggests the identification of several external worlds, each representing the context of a different stakeholder. These are schematically classified into four main categories, namely buyer, user, beneficiary and outsider. By distinguishing the need and requirement variables related to these external worlds and the processes they are involved in, it is possible to study a design activity with a detailed representation of the critical factors of the product planning phase. Specifically, the model is suitable to represent tasks such as the interpretation of stakeholders' needs, their translation into formal requirements, the proposition of trade-offs between conflicting requests and all the decisions occurring in the formulation stage of design.

The illustrative example, related to the analysis of a brainstorming session in a company producing lines for bottled beverages, illustrates a practical application of the proposed model, here dedicated to the analysis of the definition of a new line specification. The debate of the session, post-processed as for a typical design protocol study, dealt with technical, economic and environmental issues emerging while considering the different expectations of the purchase department, the operations, and the maintenance crew of the customer, as well as of the final customers of the beverages.

The proposed model reveals to be suitable for mapping each statement with respect to the different actors involved and the object of the discussion. The motivation behind the positions of the diverse actors, i.e. the needs they try to satisfy, and their formal representation in the product specification, i.e. the requirements, are thoroughly represented within the entire process that goes from needs identification to the requirement formulation. Still following the same classification scheme of the original situated FBS framework, thus distinguishing the external world, the interpreted world and the expected world, the design discourse can be decomposed into elementary processes that span over a three-dimensional space type of variable-actor -world. Such analysis permits highlighting gaps due for example to the lack of information (e.g., when a stakeholder's opinion is attributed to another), or to logical jumps (e.g., as revealed by the missing processes in Table 2).

More in general, the proposed model is expected to be fruitfully applicable with several purposes, such as the analysis of the activity of an innovation team, the verification of the impact of a training on product planning, and the quality of a design method. From this perspective, the main limitation seems to be the time needed to carefully classify a design protocol, given the increased number of variable types, worlds and actors to be considered. Besides, according to the authors' knowledge, no other models currently exist to conduct detailed investigations on the earliest phases of product development. Given the importance of the latters, especially within innovation activities, the model presented in this paper and its application can be considered worthy of further investigation and improvement.

References

1. Mahajan V, Muller E, Wind J (2000) New product diffusion models. Kluwer Academic, New York
2. Cantamessa M (2011) Design … but of what? The future of design methodology. Springer, London, pp 229–237
3. Laffont JJ (1988) Fundamentals of public economics. MIT Press, Cambridge, MA, 287 pp
4. Cantamessa M, Cascini G, Montagna F (2012) Design for innovation. In: Proceedings of the international design conference – design 2012, Dubrovnik, Croatia, May 21–24, 2012, pp 747–756

5. Cantamessa M, Montagna F (2013) Multi-stakeholder analysis of requirements to design real innovations. In: Proceedings of the international conference on engineering design (ICED13), Seul, 19–22 Aug 2013
6. Gero JS, Kannengiesser U (2004) The situated function-behaviour-structure framework. Des Stud 25(4):373–391
7. Cascini G, Fantoni G, Montagna F (2013) Situating needs and requirements in the FBS framework. Des Stud 34:636–662
8. Gero JS (1990) Design prototypes: a knowledge representation schema for design. AI Mag 11 (4):26–36
9. Vermaas PE, Dorst K (2007) On the conceptual framework of John Gero's FBS-model and the prescriptive aims of design methodology. Des Stud 28:133–157
10. Weber M (2008) Developing what customers really need: involving customers in innovations, In: Proceeding of the 4th IEEE international conference on the management of innovation and technology (IEEE ICMIT08), Bangkok, 21–24 Sept 2008
11. Griffin A, Hauser JR (1993) The voice of the customer. Mark Sci 12(1):1–27
12. Sheth JN, Mittal B (2003) Customer behavior: a managerial perspective, Cengage learning; 2nd ed., 544 pp
13. Balachandra R, Friar K (1997) Factors for success in R&D projects and new product innovation: a contextual framework. IEEE Trans Eng Manag 44(3):276–287
14. Calantone RJ, Di Benedetto CA, Divine R (1993) Organisational, technical and marketing antecedents for successful new product development. R&D Manag 23:337–351
15. Cooper RC, Kleinschmidt EJ (1990) New products: the key factors in success. American Marketing Association, Chicago
16. Ulrich KT, Eppinger SD (2008) Product design and development. McGraw-Hill, New York
17. Alexander C (1964) Notes on the synthesis of form. Harvard University Press, Cambridge, MA
18. Krippendorff K (2006) The semantic turn. A new foundation for design. Taylor & Francis, Boca Raton
19. Kim KO, Hwang H (2011) Exploring consumer needs with Lewin's life space perspective. In: Proceedings of the 18th international conference on engineering design (ICED11), Copenhagen, Denmark, Vol 7, 214–223
20. Pucillo F, Cascini G (in press) The affordance turn: products as facilitators for user experience, Des Stud
21. Beatty SE, Kahle LR, Homer P, Misra S (1985) Alternative measurement approaches to consumer values: the list of values and the rokeach value survey. Psychol Mark 2(3):181–200
22. Maslow AH (1987) Motivation and personality, 3rd edn. Harper & Row, New York
23. Ericson A, Muller P, Larsson T, Stark R (2009) Product-service systems: from customer needs to requirements in early development phases. In: Proceedings of the 1st CIRP industrial product-service systems (IPS2) conference, Cranfield University, England, 1–2 Apr 2009, pp 62–68
24. Darlington MJ, Culley SJ (2002) Current research in the engineering design requirement. Proc Inst Mech Eng Part B Eng Manuf 216(3):375–388
25. Filippi S, Barattin D, Cascini G (2013) Analyzing the cognitive processes of an interaction design method using the FBS framework. In: Proceedings of the 19th international conference on engineering design (ICED13), Seoul, 19–22 Aug 2013

Part VI
Design Grammars

Analyzing Generative Design Grammars

Corinna Königseder and Kristina Shea

Abstract In the last decades, research on computational design synthesis using generative design grammars has achieved great improvements due to advanced search and optimization strategies. Even though meaningful grammars are inevitable to generate valid and optimized designs, only little attention has been paid to improving the development of grammar rules. The research presented in this paper focuses on supporting human designers in developing better grammars. The presented Grammar Rule Analysis Method (GRAM) supports a more systematic development process for grammar rules. It enables the rule designer to gain detailed knowledge of the performance of grammar rules, their relations to objectives, constraints and characteristics, and their interaction. The method's goal is to have a major impact on the quality of the generated designs by improving the quality of the rules. The case study uses two different grammars for automated gearbox synthesis to validate GRAM and show its potential.

Introduction

In many application areas, including architecture, art and engineering, generative grammars for design synthesis have been successful. Recent examples are in the synthesis of hybrid power trains [1] or the synthesis of gear trains [2, 3]. Although grammars are often developed to formalize and structure the design of products and processes, the process of grammar development is often rather unsystematic. Knight stated that "it is the designing of a grammar that resembles what a designer does. The development of rules for designs requires the same kind of intelligence, imagination, and guesswork as the development of designs in a conventional way" [4]. Although various methods have been developed for the conventional design process, rule development has only been given little attention so far. The lack of

C. Königseder (✉) • K. Shea
Swiss Federal Institute of Technology, Zurich, Switzerland
e-mail: ck@ethz.ch

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_21

support for grammar design was discussed more than a decade ago [5, 6] and the issue of systematically developing and testing generative grammars is addressed in a few publications [7–10]. However, this lack of support is still one of the major drawbacks of grammatical design approaches [9].

The goal of this paper is to systematically assist the rule development process by providing the Grammar Rule Analysis Method (GRAM) for Computational Design Synthesis (CDS). It supports designers of grammars by giving feedback on the performance of their developed rules, i.e. how they change design characteristics and objectives, through a set of visualizations. The designer can interpret these visualizations and adjust the grammar rules accordingly. Testing the rules during or after the development process and before they are embedded in a more complicated design synthesis process enables designers to obtain an increased understanding of the rules' performance and to validate them. In 1956, Chomsky stated that a grammar "gives a certain insight into the use and understanding of a language" [11]. GRAM's goal is to enable these insights and allow the human engineer to design better grammar rules.

The paper is organized as follows. The background section reviews different approaches on grammar development and analysis and motivates the need for a more systematic way of grammar rule development and analysis. Next, the grammar rule analysis method (GRAM) is presented followed by the description of two different graph grammars for automated gearbox design that are used as a case study in this paper. These grammars are analyzed and compared using GRAM. The results are presented and discussed along with general issues of GRAM and an outlook on future directions.

Background and Related Work

This paper uses the terminology for the CDS process as defined in Cagan et al. [12]. In the first step the designer formalizes a design problem at the required level of detail to allow for the synthesis of meaningful designs. After the representation is formalized, the CDS process consists of three repeated phases: generate, evaluate and guide. First, a grammar rule is selected and applied to the current design transforming it into a new design alternative. This design is then evaluated considering defined objectives and constraints. A decision is made in the search on how to proceed in the synthesis process, either to accept or reject the new alternative. The synthesis process is continued until either no further rule applications are possible or it is stopped by a stopping criterion in the search method.

In grammatical approaches to CDS, designers develop a grammar to represent a desired design language. It consists of a vocabulary, usually describing design shapes, components or subsystems, as well as a set of grammar rules. These rules describe design transformations, LHS→RHS, that are defined by a left-hand-side (LHS), i.e. where the rule can be applied in a design, and a right-hand-side (RHS), defining the design transformation. Common formalisms for engineering design grammars are, e.g., spatial and graph grammars. In spatial grammars, the rules are based on the shape of a design, i.e. its geometry and subshapes [12]. In graph grammars, rules apply to graph elements, which can be single nodes, arcs and subgraphs [13].

Several approaches in CDS using grammars focus on easing the process for the human designer. Examples are relieving the designer from tuning search algorithms through machine learning methods [14, 15], prescriptive methods to build a knowledge model representing expert knowledge in rules [16] or intelligent reduction of the number of design concepts that are presented to a human designer [17]. These methods have shown success in improving the CDS process in general, however they lack support for the early phase of rule development. Much effort is spent on deciding how to apply rules to generate beneficial designs rather than re-thinking the implemented rules. Although most publications on CDS methods using grammars describe the grammar rules, they give little to no hints on how these rules were developed. Recent approaches to support the development of rules either generate grammar rules automatically [18, 19], or give advice to a human designer on how to develop [4, 20, 21] and manually test a grammar [22]. For the first, extensive research is also done in other fields, e.g., grammar induction and improvement [23] for natural language processing. However, no research is known to the authors that supports rule development through systematic and automated rule analysis.

The research presented in this paper focuses on supporting the rule development process. The authors expect that "better" grammars can be developed through a systematic analysis of the rules during the rule design. Those are then the basis for a more successful synthesis process. The in-depth understanding gained in the analysis using GRAM can also deliver important insights about the search space that can be considered in tuning advanced search methods. In contrast to the work by Vale and Shea [14] where statistics are collected and rule sequences are defined during the CDS process, GRAM enables not only to reuse insights gained before the CDS process but also to analyze the grammar itself and to improve it based on the analysis results.

Method

GRAM is presented in Fig. 1. Dark grey boxes show steps that are carried out automatically in the current implementation, light grey boxes show steps that will be automated in the future.

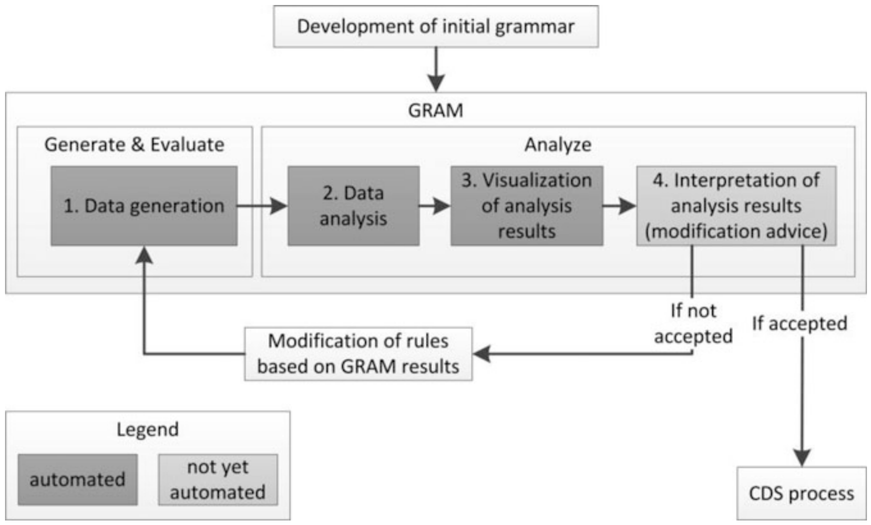

Development of initial grammar
GRAM
Generate & Evaluate
1. Data generation
Analyze
2. Data analysis
3. Visualization of analysis results
4. Interpretation of analysis results (modification advice)
If not accepted
If accepted
Modification of rules based on GRAM results
Legend
automated
not yet automated
CDS process

Fig. 1 Grammar Rule Analysis Method (GRAM) to analyze grammar rules for CDS based on the extended SG process shown in [5]

GRAM analyzes a developed grammar in a systematic way to give feedback on the rules' performance. Individual rule performances (Q1) as well as the performance of the whole rule set (Q2–Q6) are assessed such that the rule designer is able to answer the following questions when interpreting the results:

Q1. What impact does the rule have on which objective?
Q2. How probable are the applications of each rule?
Q3. What solution space do the rules define?
Q4. Does the rule set favor certain designs?
Q5. How many valid designs are generated?
Q6. How many different designs are generated?

GRAM has a defined way to generate and analyze data. Information from the data analysis is visualized and interpreted by the designer to gain a better understanding of the grammar itself. The different steps in GRAM are described in more detail in the following and a schematic representation of the GRAM steps 1–3 is shown in Fig. 2 to accompany these descriptions. GRAM is best illustrated using an example, so a grammar for gearbox synthesis is used here that will be further introduced in the case study. Note that the validity and diversity ratio are exaggerated in this schematic overview but exact ratios are given in the result section. GRAM allows analysis for any number of objectives, design characteristics and rules but for the sake of clarity it is shown here for this reduced example.

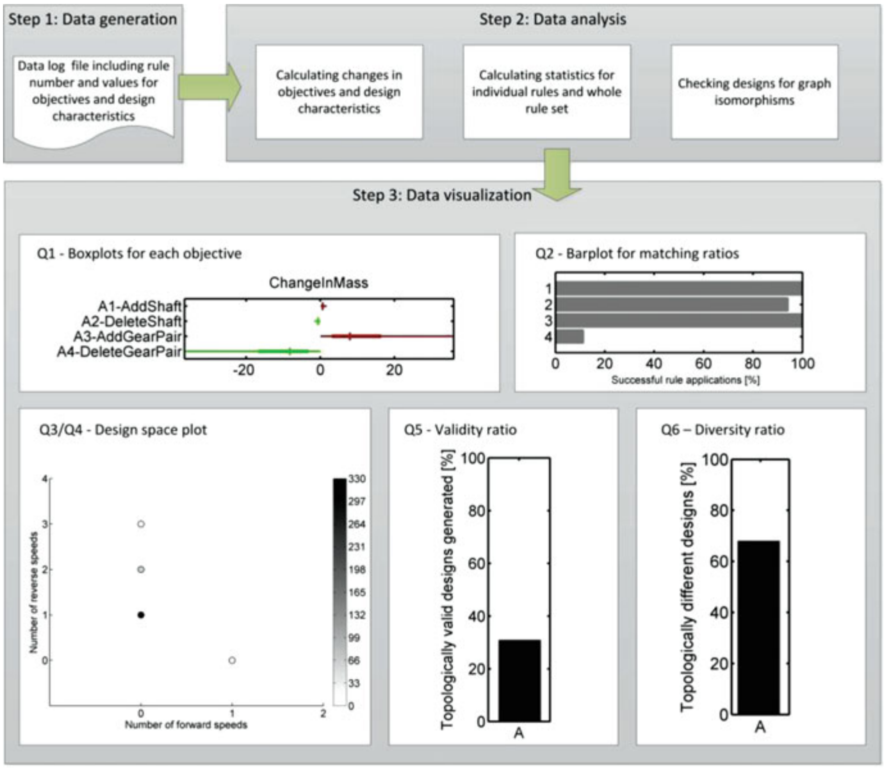

Step 1: Data generation
Data log file including rule number and values for objectives and design characteristics
Step 2: Data analysis
Calculating changes in objectives and design characteristics
Calculating statistics for individual rules and whole rule set
Checking designs for graph isomorphisms
Step 3: Data visualization
Q1 - Boxplots for each objective
ChangeInMass
A1-AddShaft
A2-DeleteShaft
A3-AddGearPair
A4-DeleteGearPair
Q2 - Barplot for matching ratios
Successful rule applications [%]
Q3/Q4 - Design space plot
Number of reverse speeds
Number of forward speeds
Q5 - Validity ratio
Topologically valid designs generated [%]
Q6 – Diversity ratio
Topologically different designs [%]

Fig. 2 Schematic overview of the GRAM steps 1–3

Data Generation

To analyze grammar rules, a variety of data, i.e. objectives and design characteristics, is acquired. In most engineering applications there are multiple objectives and it is recommended to store the metric for each objective individually. Here, constraints formulated as soft constraints, i.e. penalty functions, are included. Design characteristics can be individual variables in the rules but are more commonly system characteristics, e.g. number of components or component types. The data is generated using a simple generate-and-test process. It starts with an initial design. A rule is selected randomly from all implemented rules. It is applied, the generated design is evaluated and the data is stored. The generated design resulting from this rule application is taken as the basis for the next iteration. It is not a generate-and-test search process as the design resulting from the rule application is always used as the starting point for the next rule application regardless of the impact on design objectives.

Data Analysis

For all data, the change in the objectives is calculated to analyze the performance of each individual rule. The generated designs are analyzed to identify topologically equivalent designs to be able to represent the design space and to identify if the rules favor certain topologies, i.e. generate them multiple times. Additionally, some basic statistical models are built to prepare the visualization and support the interpretation.

Visualization and Interpretation of Analysis Results

Five different diagrams are presented in Fig. 2 to visualize the data obtained in the analysis. For a rule set of n_r rules and an analysis of n_o objectives using n_d design characteristics, the following diagrams are generated: Q1) n_o boxplots with n_r boxes each, Q2) one barplot with n_r bars, Q3/Q4) one n_d-dimensional design space plot, additional boxplots if required, Q4) one ratio for valid designs and Q5) one ratio for different designs. The diagrams are explained below and important issues for their interpretation are given.

Q1 – General performance analysis using boxplots for each objective. For each objective a diagram (Fig. 3) is generated showing how it is influenced by each rule (given on the y-axis). The user defines the desired direction of change derived from the problem formulation and a color coding gives a quick overview if the rule changed the objective in the desired direction or not. The red (dark grey) color indicates a change against the desired direction and the green (medium grey) color a change in the desired direction. Yellow (light grey) boxes show that changes in both directions are possible and black (black) is used to represent rules that have no influence on an objective. The whiskers, defined by the thin line, represent the maximum and minimum value of the dataset excluding outliers, i.e. data points more than three halves away from the lower or upper quartile. The box spans from

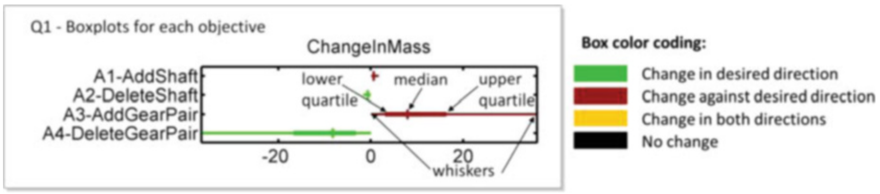


Fig. 3 Q1 – boxplot for the change in mass for the example rule set

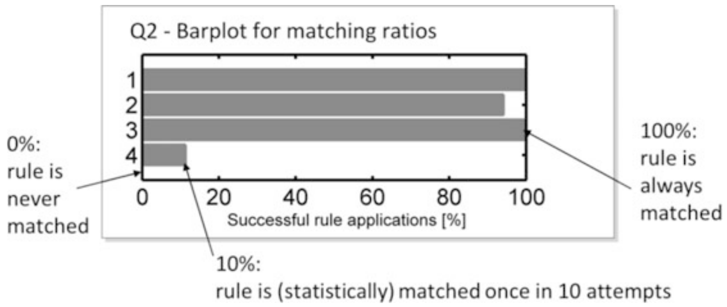


Fig. 4 Q2 – barplot for matching ratios for the example rule set

the lower quartile to the upper quartile showing also the median. Using these diagrams, the engineer can visualize the performance of each rule considering each objective separately. Interpreting the diagrams, the designer has to consider that changes against the desired direction, e.g. increasing an objective rather than decreasing it, are often valuable for design synthesis. This means that changes against the desired direction do not automatically identify inferior rules that should be removed. In contrast, it encourages the rule designer to think also about the sequences in which rules can be applied and to consider combining these sequences to create more specific rules to facilitate the generation of meaningful designs.

Q2 – Bar plots to represent matching ratios for each rule. For more detailed information on a rule's applicability, matching ratios are calculated and visualized (Fig. 4). Throughout this paper, the matching ratio of a rule is defined as the number of LHS matches of a rule divided by the number of attempts to apply this rule. This ratio defines how likely it is for a rule to be applied with a matching ratio of 100 % meaning a rule can always be applied while a matching ratio of 0 % represents a not reachable rule. From the matching ratios the rule designer can reason about the LHSs of the rules. This often helps to explain the design space that is generated with the rule set. Rules that have a very low application probability are only rarely applied. In grammars with unbalanced rule application probabilities, i.e. some rules are applied very often and others very rarely, the rule designer can, for example, consider formulating the LHS of a rarely applied rule differently to allow its application more often or decide to remove such a rule from the rule set to have a more compact grammar. Additionally the use of guidance strategies or predefined sequences for the CDS process can be helpful to improve the rule's application. The interpretation of matching ratios is dependent on the rule design as well as the search and optimization algorithm used later for design synthesis. When using intelligent search methods, it may not be required to ensure higher matching ratios for all rules, whereas when using simple generate-and-test type algorithms, this may be more helpful to explore the design space.

Q3/Q4 – Visualization of the design space. To show the size of the design space, a matrix with the dimensions of the design characteristics 1 and 2, is presented (Fig. 5). Each point in the space indicates that a design with, e.g.,

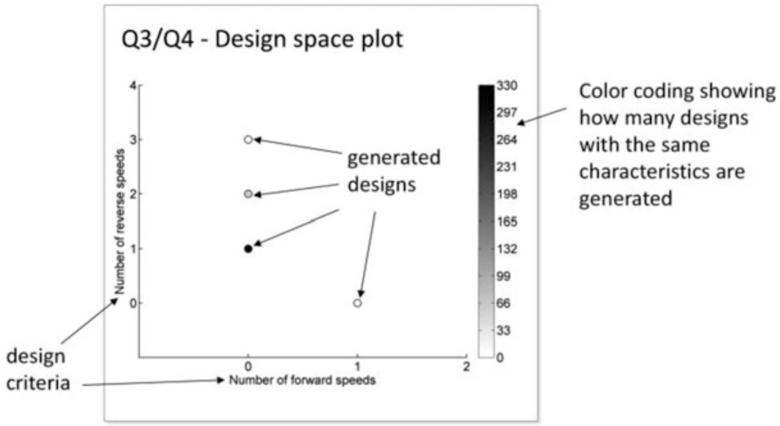


Fig. 5 Q3/Q4 – design space plot for the example rule set

x elements of design characteristic 1 and *y* elements of design characteristic 2 exists. The color indicates how often a design with the respective characteristics is generated. This plot gives an indication about the design space the rules generate. The color can be used to identify solutions in the design space that are favored by the rules ("hot spots"). When continuous design characteristics are required or when the user is interested in additional information on objectives, the *x*- and *y*-axes can be made continuous or boxplots for each objective and design characteristic can be made in addition to the design space representation.

The space is generated using random generation without feedback. The rule engineer has to consider this when interpreting the results. It can happen that the space is larger than intended, e.g. when the designer allows invalid designs and plans to use penalty functions and an optimization algorithm for the CDS process and these penalized designs are not removed from the design space yet. On the other hand, it can also happen that the generated design space is small because certain rules undo what previous rules did. To derive useful measures to improve a rule set, the rule designer has to consider not only the space explored and the favored designs during the data generation process in GRAM, but also the search and optimization process that will be used.

Q5 – Validity ratio. The validity ratio is defined here as the number of valid designs divided by the number of total designs generated. The validity of a design is defined by the designer and can be, for example, the necessity to have a connection between two components, e.g. a connection between the input and the output shaft in the gearbox example. The validity ratio gives feedback on the probability that the analyzed grammar generates valid designs with simple generate-and-test type algorithms. The lower the validity ratio is, the more intelligent the guidance has to be to lead the grammar rule application to produce feasible designs. On the other hand, a low validity ratio does not mean that a grammar necessarily produces inferior results compared to a grammar with a validity ratio of 1, i.e. generating only valid designs. In some cases it is required to generate invalid intermediate designs to be able to eventually transform an invalid design into a valid one. It is the designer's choice to decide whether or not invalid designs should be allowed during design generation for a specific problem formulation and to interpret the validity ratio accordingly.

Q6 – Diversity ratio. The diversity ratio is defined as the number of valid and topologically different designs generated during the data generation phase divided by the number of all valid designs generated during the data generation phase. A high diversity ratio means that the grammar generates topologically different designs with a high likelihood, i.e. the design space is more easily explored than when having a lower diversity ratio and generating the same designs repeatedly. If required, a visualization similar to the design space representation (see Q3) is possible. In this case the user can analyze how often each individual topology is generated, i.e. if rules favor certain topologies. Note that in the presented case study, designs with the same design characteristics can have different topologies. The rule designer has to be aware of the fact that the diversity ratio reflects only the design space explored during the data generation process. In most cases the entire design space is unknown and considering parametric rules can be infinite.

Using these diagrams and their interpretations, the designer can check if the grammar represents the intended design language and interpret the relative ease of generating known, intended designs or they can further improve the grammar considering the analysis.

Case Study: Automated Gearbox Synthesis

To show the applicability of the proposed method, GRAM is applied to two different grammars (A and B) for automated gearbox synthesis. Two additional grammars are analyzed and compared in [24]. Gearbox design using generative grammars is an established CDS problem and research has been carried out by several researchers [2, 3, 25–29].

In this case study the task is to develop a gearbox that fits into a defined bounding box and has a defined number of forward and reverse speeds. The grammars are formulated and implemented as graph grammars consisting of a metamodel and a rule set. The rule sets contain both topologic and parametric rules.

Application of GRAM to the Gearbox Rule Sets A and B

The data generation is conducted with 50 times 1,000 rule applications for each rule set. **Data generation** is carried out using a gearbox synthesis system developed by the authors based on GrGen, an open source graph rewriting tool [30] (http://www.grgen.net). The objectives defined in this case study are (a) the total mass of the components and (b) the amount of collision, a metric calculated based on axial and radial overlap of all components (see [2] for the exact formula). The number of forward speeds and the number of reverse speeds are defined as design characteristics. The initial design is a bounding box with the input and output shaft. **Data analysis** and **visualization** are carried out using Matlab®, the graph isomorphism check, to identify topologically identical designs, is carried out using GrGen.

Metamodel

The metamodel describes all elements that can be used as building blocks within a generative grammar [1]. For this case study, the same metamodel is used for grammar A and B. It consists of three different node types for gears, shafts and the bounding box and one directed edge type to connect nodes. All nodes have parameters to specify the components they represent, e.g. diameter, gear width and position for gears.

Implementation of the Rule Sets

Rule set A is a re-implementation of an existing published generative grammar [28, 29], rule set B is an extension of a later version of the grammar [2]. The rules are all implemented as graph grammar rules in GrGen. An overview of both rule sets is given in Fig. 6. The schematic images for the rules are for visualization purposes only. The schematic graph representations for rules B2, B4 and B6 represent the basic idea of the rule but not the exact LHS matches. Nodes of the LHS are, however, marked in red (dark grey) in the example graphs. Rule set A consists of four rules and was originally developed to generate watches and a winding mechanism in a camera, i. e. requiring only one speed. It considers only shafts that extend the width of the whole bounding box. Rule set B consists of 11 rules that are more sophisticated than those of rule set A in both their LHS and RHS. Rule set B was originally developed for automated gearbox synthesis, i.e. considering multiple speeds. It was further developed to generate valid designs in every rule application as long as the initial design is valid. This decision was made by the rule designers to ensure design evaluation after each rule application. Rule set B is different from the developed rule sets in Lin et al. [2] in that it has two additional rules to change the lengths of shafts and in that the LHS of several rules account for changed lengths of shafts.

The grammar, rule set B is based on, was originally developed for automated gearbox synthesis, i.e. considering multiple speeds. A small example of a sequence applying the rules B3 $\geq$ B1 $\geq$ B8 to the initial design is given in Fig. 7 (top), also showing the graph, a 3D representation for the designs generated and the corresponding objective values and design characteristics (bottom).

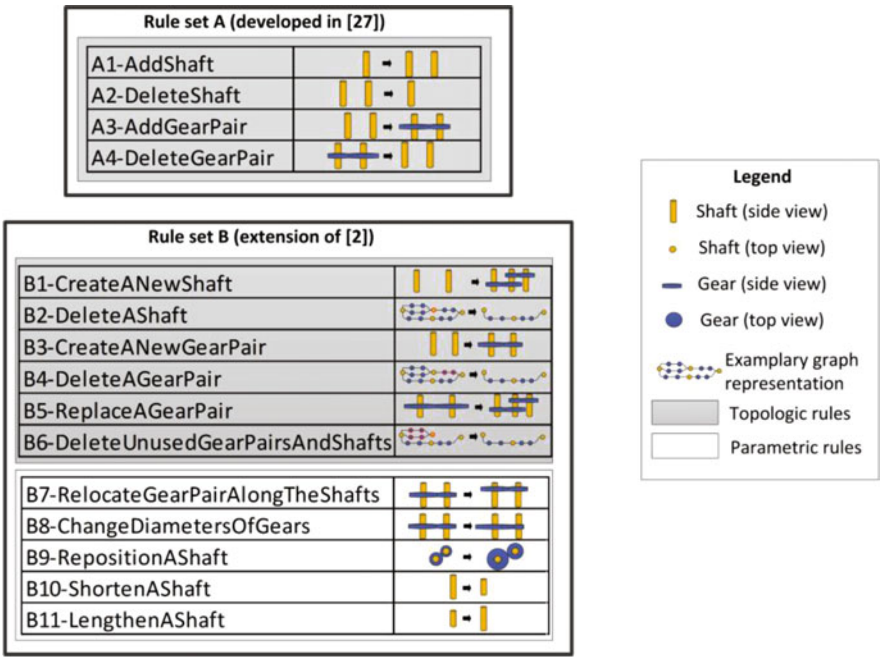


Fig. 6 Overview of both rule sets organized by their type (topologic or parametric); rule number (consisting of rule set label and rule number), name and a pictorial description

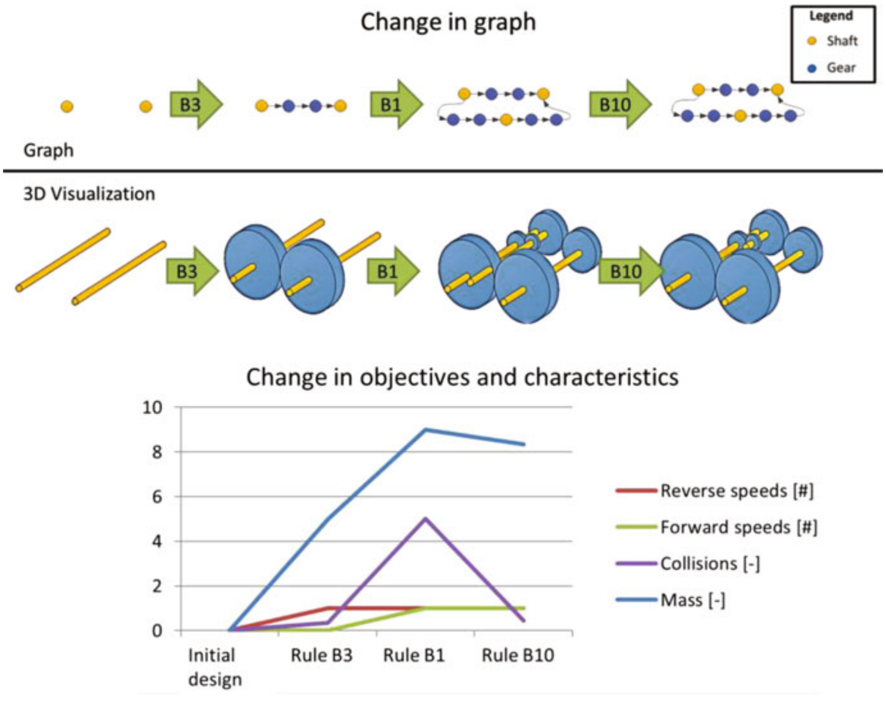


Fig. 7 Example of applying rules from rule set *B*; changes in the graph and a 3D representation (*top*), the respective objective values and design characteristics (*bottom*)

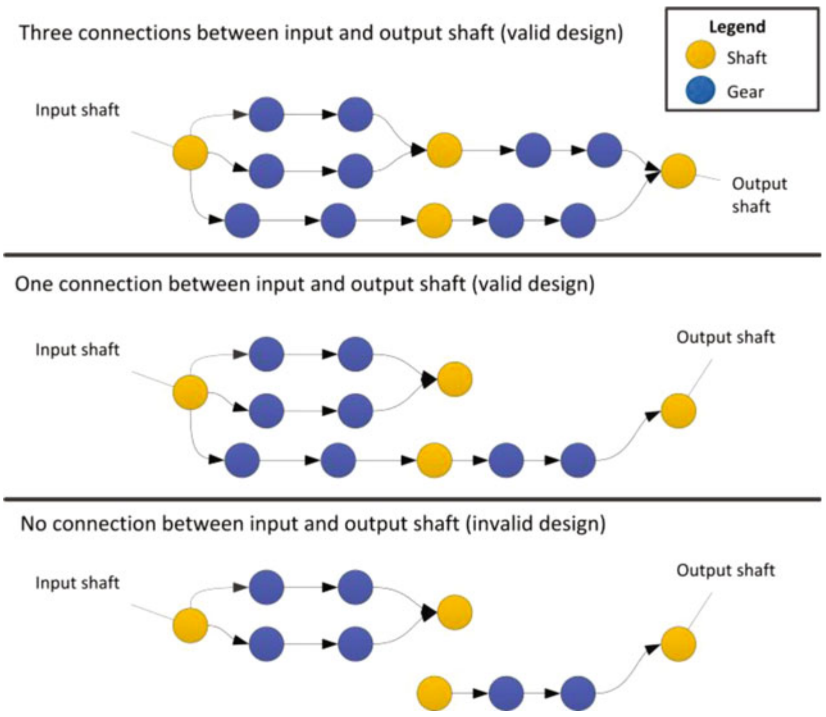

Three connections between input and output shaft (valid design)
Legend
Shaft
Gear
Input shaft
Output shaft
One connection between input and output shaft (valid design)
Input shaft
Output shaft
No connection between input and output shaft (invalid design)
Input shaft
Output shaft

Fig. 8 Three examples to show how valid designs, i.e. designs having a connection between input and output shaft, are defined

For this case study, a valid design must have at least one speed, i.e. there must be at least one path in the graph connecting the input and output shaft. Figure 8 visualizes this definition of valid designs for the case study.

Results

The results from the case study are presented below. Interpreting the analysis results, the questions Q1 to Q6 can now be answered.

Q1. Which impact does the rule have on which objectives?

The influence of each of the rules for rule set A is shown in Fig. 9 and for rule set B in Fig. 10. Adding components (rules A1 and A3) always increases (red (dark grey) color in boxplot) mass and collisions, deleting them (rules A2 and A4) reduces (green (medium grey) color in boxplot) both objectives.

Rules B1-B4 in rule set B show a do-undo behavior, where rules B1 and B3 add mass and collisions by adding components, rules B2 and B4 reduce both objectives (Fig. 10). Comparing rules A1–A4 to rules B1–B4, a difference in the

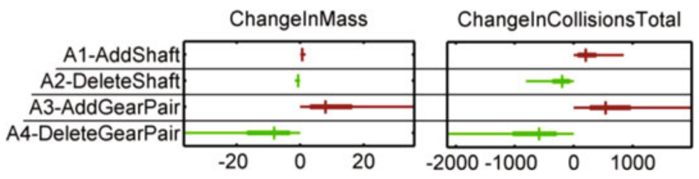


Fig. 9 Influence of rules in rule set *A* on the objectives mass and collisions

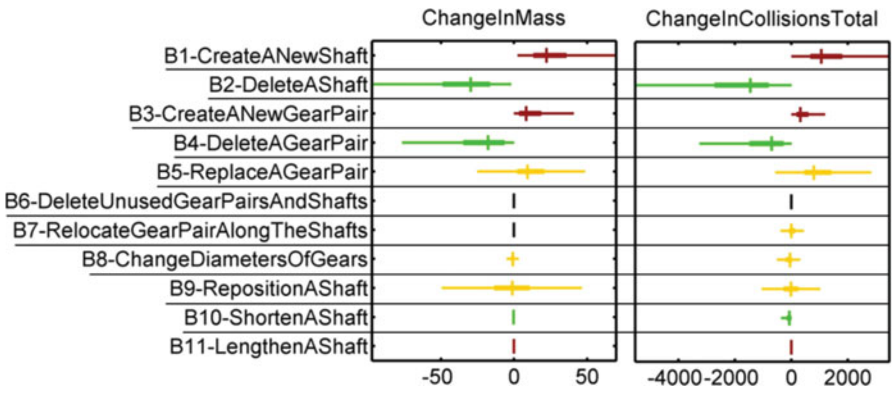


Fig. 10 Influence of rules in rule set *B* on the objectives mass and collisions

magnitude of the change in both objectives can be seen. This stems from the different implementation in rule set B. Rule A1 for example only adds a single shaft, whereas rule B1 adds a shaft and connects it to the existing design with a gear pair, i.e. more components are added within the rule which results in bigger changes in both mass and collisions.

Rule B6 has no influence on either mass or collisions. If the rules are implemented correctly, this should not occur. In this case it stems from a careful implementation of rule set B ensuring that after every rule application no dangling nodes remain in the design. So this rule from a previous implementation by Lin et al. [2], intending to repair designs, is not necessary any more. It can be removed from the rule set.

Q2. How probable are the applications of each rule?

For all rules in rule sets A and B, matching ratios, i.e. the number of successful matches of the LHS divided by the number of attempts to apply the rule, are represented as horizontal bars in Fig. 11.

Rules with simple LHSs are applied more frequently than those with more restrictive LHSs caused by constraints, e.g. on parameters of the nodes, or components relations, i.e. more complex sub graphs. This can be seen for example when comparing rules A2 and B2 that both delete a shaft. Rule set B is implemented to generate only topologically valid designs and the knowledge

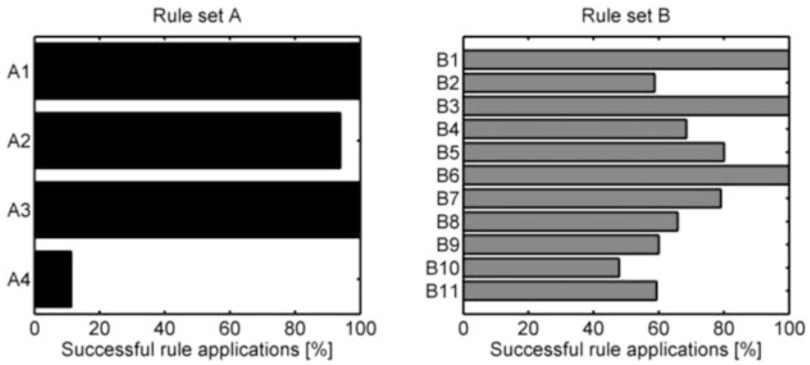


Fig. 11 Percentages of successful rule matches for rule sets *A* and *B*

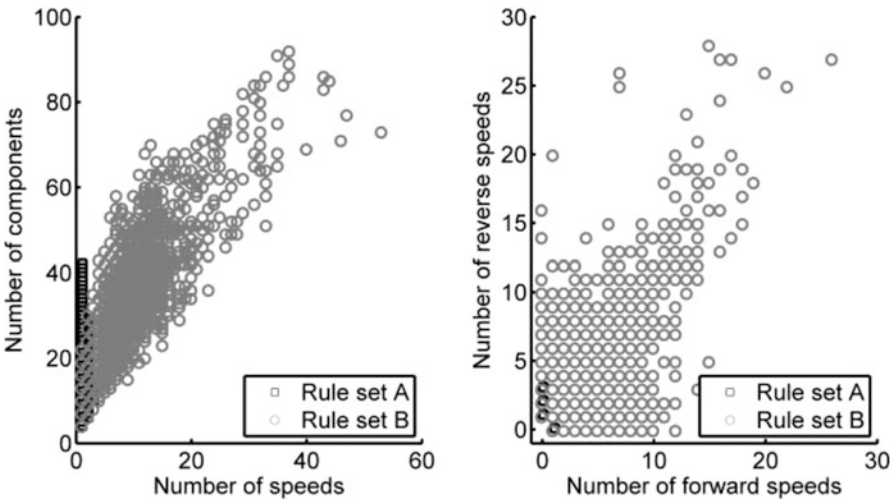


Fig. 12 Two representations of the design space generated by the two rule sets. Number of speeds versus number of components in a design (*left*) and number of forward speeds versus number of reverse speeds (*right*)

to do so is implemented in the LHS of the rules, thus allowing its application less frequently.

Q3. What solution space do the rules define?

Figure 12 shows the design space generated by each of the rule sets in 50 runs applying 1,000 rules selected randomly. On the left, the number of speeds, i.e. the sum of forward and reverse speeds is plotted versus the number of components, i.e. the sum of shafts and gears in a design. On the right, the two design characteristics, i.e. the number of forward and reverse speeds in a design, are plotted against each other; see also Fig. 13 for more details. Each point in

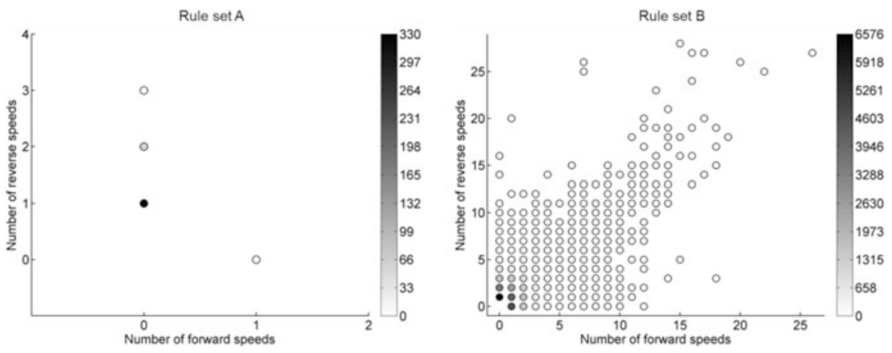


Fig. 13 Plots of the design spaces generated by rule sets *A* and *B*. The *color* indicates how often a design with this speed configuration was generated in 50,000 rule applications

these plots indicates that a design with the given characteristics was generated. The design spaces of rule set A is very small. Although designs with many components exist, they have low numbers of speeds. This can be explained by the simple grammar rules that are more dependent on an anticipated intelligent search and often do not produce good designs when applied randomly. Rule set B, with its rules developed to perturb but not destroy existing solutions, generates designs with much higher numbers of speeds. The designs are spread out more in both represented design spaces.

Q4. Does the rule set favor certain designs?

More detailed information on the design spaces for the two design characteristics is given in Fig. 13. Every point in the design space illustrates that at least one design with this speed configuration is generated with the color of the point indicating how often. From these diagrams it can be seen that both rule sets favor designs with few forward and reverse speeds.

Not all rule applications result in valid designs, so it is important to know how often a rule set produces invalid designs. For design synthesis it is also important how likely a newly generated design is different from already generated ones. Figure 14 gives a summary of both of these issues.

Q5. How many valid designs are generated?

On the left of Fig. 14 the validity ratios are given. For this case study a topologically valid design is a design that has at least one connection between input and output shaft (Fig. 8). It can be seen that this ratio increases from rule set A to B as the number of rules to connect shafts grows.

Q6. How many different designs are generated?

On the right of Fig. 14 the diversity ratios are shown. Comparing the rule sets underlines what has been found in the design space diagrams. Rule set A produces many designs of the same topology, rule set B produces more topologically different solutions.

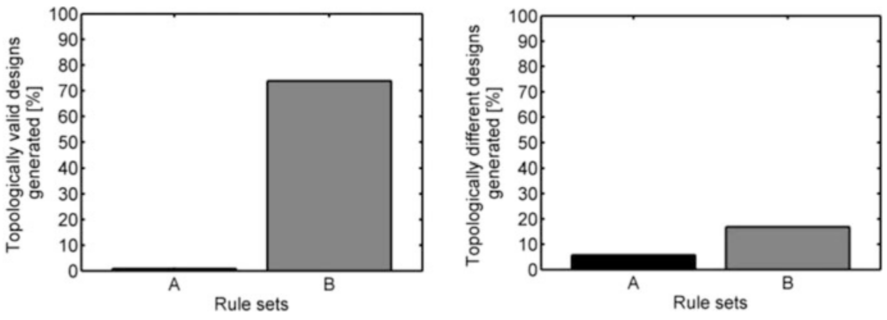


Fig. 14 Validity ratios (*left*) and diversity ratios (*right*) for rule sets *A* and *B*

Discussion

The case study shows that GRAM is capable of supporting design engineers to analyze their developed rule set. The influence of rules on the defined characteristics and objectives is shown and comparing the rules' performance to the rules' intended performance allows to test the language against the intent. The finding that rule set A generates designs with fewer speeds and rule set B generates designs with more speeds, for example, reflects the purpose for which the rule sets were originally developed, i.e. generating single speed gearboxes for rule set A, and generating gearboxes with multiple speeds for rule set B. No unintended performance was discovered in this case, which might be explained by the long history of improving and further developing the grammars of the case study. Non-influential rules can be detected to allow the reduction of the rule set, e.g. rule B6. This was a useful discovery due to GRAM and shows a secondary use of the method for rule set debugging.

The design space representation with the data from the experiment allows statements to be made about the ambiguity of the design grammar with respect to the defined design characteristics. The case study shows, for example, that the grammar is ambiguous as designs with the same design characteristics are generated several times and that for a future exploration of the design space using a search algorithm, the designer has to keep in mind, that the rules by the nature of their implementation favor simple designs, i.e. few forward and reverse speeds. Further, GRAM provides support for rule debugging. If, for example, an error occurred in the implementation of rule A2 (*delete shaft*) such that the metric for collisions increases, GRAM would show this unintended performance of the rule in the boxplots. Similarly, GRAM helps to identify rules that are never applied, e.g. through the matching ratios, or that have no influence on any of the objectives, e.g. through the boxplots. This visual feedback on the rules enables the rule designer to find errors in the implementation and identify starting points for improving the quality of the developed rule sets. Further it provides key inputs for selection and tuning of the search method. Although the case study uses graph grammars, any type of generative design grammar can be analyzed using GRAM.

Further, the method is independent of rule type and definition, i.e. simple vs. -knowledge-intensive and topologic as well as parametric grammar rules. As feedback is given based on each rule's performance with respect to defined objectives, it is necessary that intermediate as well as final designs can be evaluated. GRAM is developed to allow rule developers freedom in designing their grammar, while still enabling its applicability to the developed rules and supports the analysis of topologic as well as parametric grammar rules.

Issues to be tackled are related to the visualization for large scale problems as well as automating the interpretation. The visualization of the design space becomes more complex when more than two design criteria are defined. This is a known issue and research topic also in other domains and new visualization techniques have to be investigated. Additionally, analyzing not only the performance of individual rules but also rule sequences allows better understanding of rule sets and how sequences of rules impact design criteria, for example, analyzing the trade-off between rule B5 versus the necessary sequence A4 – A1 – A3 – A3 to achieve the same topology.

Future work is planned to address an automated interpretation of the statistics gained in the analysis using GRAM to overcome both the drawbacks in visualization for multi-objective problems and many rules as well as decreasing the interpretation effort by a human designer. All future directions aim to increase the understanding of developed grammars and further extend the idea to minimize the adaption effort of the search algorithm to the design problem while increasing the quality of the synthesis results.

Conclusion

The presented method, the Grammar Rule Analysis Method (GRAM), to support the human designer in the development of generative grammar rules for CDS is the first approach known to the authors that focuses on a systematic analysis of the rules in the development phase. This is a great difference to current research that focus especially on tuning of the search algorithm after rules are developed. GRAM enables the rule designer to gain detailed knowledge of the rules' performance, their relations to objectives, constraints and characteristics, and their interaction. Additionally, the easily readable feedback allows easy grammar debugging to find errors in the implementation of the rules before they are used in the synthesis process. Besides less effort to adapt the search algorithm due to better understanding of the solution space defined, superior synthesis results by the grammar rules are expected. With GRAM, future rule designers are given a means to reason about their grammar rule implementations in a systematic way.

References

1. Helms B, Shea K (2012) Computational synthesis of product architectures based on object-oriented graph grammars. J Mech Des 134(2):021008-1–021008-14
2. Lin YS, Shea K, Johnson A, Coultate J, Pears J (2010) A method and software tool for automated gearbox synthesis. In: ASME 2009 Int. design engineering technical conf. & computers and information in engineering conf. 2010. San Diego, CA. p 111–121
3. Swantner A, Campbell MI (2012) Topological and parametric optimization of gear trains. Eng Optim 44(11):1351–1368
4. Knight TW (1998) Designing a shape grammar – problems of predictability. In: Gero JS, Sudweeks F (eds) Artificial intelligence in design ′98. Dordrecht, Kluwer
5. Gips J (1999) Computer implementation of shape grammars. In: Workshop on shape. Computation MIT, 1999
6. Knight T, Stiny G (2001) Classical and non-classical computation. Arq: Archit Res Q 5(4):355–372
7. Chase SC (2002) A model for user interaction in grammar-based design systems. Autom Constr 11(2):161–172
8. Li X, Schmidt LC, He W, Li L, Qian Y (2004) Transformation of an EGT grammar: new grammar, new designs. J Mech Des 126(4):753–756
9. Chakrabarti A, Shea K, Stone R, Cagan J, Campbell M, Hernandez NV, Wood KL (2011) Computer-based design synthesis research: an overview. J Comput Inf Sci Eng 11(2):021003-1–021003-10
10. McKay A, Chase S, Shea K, Chau HH (2012) Spatial grammar implementation: from theory to useable software. Artif Intell Eng Des Anal Manuf 26(Special Issue 02):143–159
11. Chomsky N (1956) Three models for the description of language. IRE Trans Inf Theory 2(3):113–124
12. Cagan J, Campbell MI, Finger S, Tomiyama T (2005) A framework for computational design synthesis: model and applications. J Comput Inf Sci Eng 5(3):171–181
13. Gips J, Stiny G (1980) Production systems and grammars – a uniform characterization. Env Plan B Plan Des 7(4):399–408
14. Vale CAW, Shea K (2003) A machine learning-based approach to accelerating computational design synthesis. In: Int. Conf. Engineering Design, ICED '032003, The Design Society, Stockholm
15. Ruiz-Montiel M, Boned J, Gavilanes J, Jiménez E, Mandow L, Pérez-de-la-Cruz J-L (2013) Design with shape grammars and reinforcement learning. Adv Eng Inform 27(2):230–245
16. Schotborgh WO (2009) Knowledge engineering for design automation. University of Twente, Enschede, p 137
17. Poppa KR, Stone RB, Orsborn S (2010) Exploring automated concept generator output through principal component analysis. In: Proc. ASME 2010 Int. Design Engineering Technical Conf. & Computers and Information in Engineering Conf. 2010. Montreal, Canada. p 185–192
18. Orsborn S, Cagan J, Boatwright P (2008) Automating the creation of shape grammar rules. In: Gero JS, Goel AK (eds) Design computing and cognition '08. Springer, p 3–22
19. Orsborn S, Cagan J, Boatwright P (2008) A methodology for creating a statistically derived shape grammar composed of non-obvious shape chunks. Res Eng Des 18(4):181–196
20. Cagan J (2001) Engineering shape grammars. In: Antonsson EK, Cagan J (eds) Formal engineering design synthesis. Cambridge University Press, Cambridge, UK, pp 65–92
21. Rudolph S (2006) Semantic validation scheme for graph grammars. In: Gero J (ed) Design computing and cognition'06. Springer, Netherlands, pp 541–560
22. Shea K, Cagan J (1999) Languages and semantics of grammatical discrete structures. Artif Intell Eng Des Anal Manuf 13(04):241–251

23. Klein D, Manning CD (2002) A generative constituent-context model for improved grammar induction. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics 2002 Association for Computational Linguistics, Philadelphia. p 128–135
24. Königseder C, Shea K (2014) Systematic rule analysis of generative design grammars (in press). Artificial Intelligence for Engineering Design, Analysis and Manufacturing. 28(3)
25. Pomrehn LP, Papalambros PY (1995) Discrete optimal design formulations with application to gear train design. J Mech Des 117(3):419–424
26. Schmidt LC, Shetty H, Chase SC (2000) A graph grammar approach for structure synthesis of mechanisms. J Mech Des 122(4):371–376
27. Li X, Schmidt L (2004) Grammar-based designer assistance tool for epicyclic gear trains. J Mech Des 126(5):895–902
28. Starling AC (2004) Performance-based computational synthesis of parametric mechanical systems. PhD dissertation. Cambridge University Engineering Department 2004. University of Cambridge, Cambridge
29. Starling AC, Shea K (2005) A parallel grammar for simulation-driven mechanical design synthesis. In: ASME 2005 Int. design engineering technical conf. & computers and information in engineering conf. 2005. Long Beach. p 427–436
30. Geiß R, Batz G, Grund D, Hack S, Szalkowski A (2006) GrGen: a fast SPO-based graph rewriting tool. In: Corradini A et al (eds) Graph transformations. Springer, Berlin, pp 383–397

From Shape Rules to Rule Schemata and Back

Athanassios Economou and Sotirios Kotsopoulos

Abstract Shape rules and rule schemata are compared in terms of their expressive and productive features in design inquiry. Two kinds of formal processes are discussed to facilitate the comparison. The first proceeds from shape rule instances and infers rule schemata that the shape rules can be defined in. The second proceeds from rule schemata and postulates shape rule instances that can be defined within the schemata. These two parallel processes mirror our intuition in design: the conceptual need to frame explicit actions within general frameworks of principles, and the productive need to supply general principles with an explicit system of actions.

Introduction

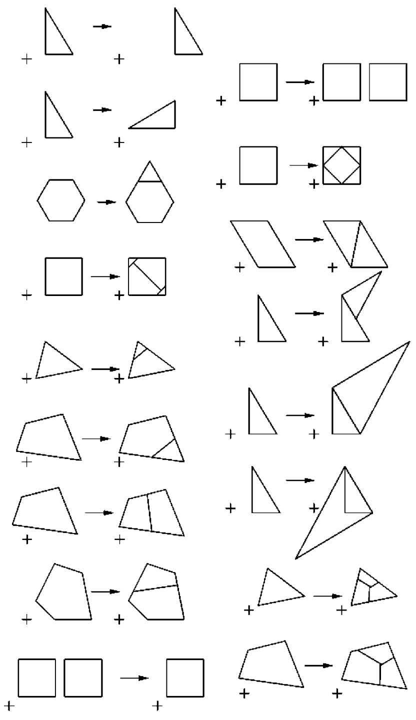
Shape rules and rule schemata have always been at the center of shape computation discourse [1, 2]. The algebraic foundations, mechanisms and conventions underlying both constructs have been carefully crafted over time and have provided a formidable framework for the study of visual calculation with shapes and design at large. Recently shape rules, parametric shape rules and the rule schemata within which these are defined have all been generously recast to provide a more comprehensive approach to design formalism [3, 4]. Perhaps the most significant new idea is the reformulation of the schema as an abstract symbolic expression that takes on shapes in its variables. In prior discourse the schema was shape-specific (parametric shape); all the shapes that were defined in this schema were determined by an assignment of real values to the variables of the schema. The new extended formulation of a shape schema allows for the representation of any parametric shape as an assignment to the variables of the schema. Clearly the power of

A. Economou (✉)
Georgia Institute of Technology, Atlanta, GA, USA
e-mail: economou@coa.gatech.edu

S. Kotsopoulos
Massachusetts Institute of Technology, Cambridge, MA, USA

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_22

Table 1 Shape rules and rule schemata

Shape rules	Rule schemata		
		$\rightarrow$	
	x	$\rightarrow$	
		$\rightarrow$	x
	x	$\rightarrow$	x
	x	$\rightarrow$	$t(x)$
	x	$\rightarrow$	$t^*(x)$
	x	$\rightarrow$	$p(x)$
	$p(x)$	$\rightarrow$	x
	x	$\rightarrow$	$b(x)$
	$b(x)$	$\rightarrow$	x
	x	$\rightarrow$	$d(x)$
	$d(x)$	$\rightarrow$	x
	x	$\rightarrow$	$x+t(x)$
	$x+t(x)$	$\rightarrow$	x
	x	$\rightarrow$	$d(t(x))$
	x	$\rightarrow$	$x+t^*(x)$
	x	$\rightarrow$	$b(b(x))$
	x	$\rightarrow$	$x+\sum t(x)$
	$b(b(x))$	$\rightarrow$	x
	$x+t^R(x)$	$\rightarrow$	$x+t(x)+t^R(x+t(x))$

symbolic expressions in either side of a rule to function as variables that can instantiate shapes suggests an entirely new way at looking at shape rules that nicely complements the existing visual approach in shape grammars. A list of shape rules and a list of rule schemata are juxtaposed in Table 1. Note that the list of shape rules and the list of rule schemata are independent.

The juxtaposition of these two representations of rules – and the possibilities they suggest when they are set one against the other is quite telling. The key characteristics they foreground – one emphasizing shapes, geometry and visual representation, the other emphasizing abstraction, and symbolic/discursive perspective, – together indeed suggest a rich structure to be explored and contrasted. Intuitively the contrast between pictorial rules and symbolic rules given in shape rules and rule schemata suggests the useful dichotomy between visual and discursive symbols [5]. The analogy is clear. Shape rules are given in terms of visual means including specific shapes and other visual tokens as needed. Rule schemata are given in terms of symbolic means including symbols, operations, and other indexical tokens as needed. Shape rules come as visual devices devoid of any structure; shapes fuse and split in any way desired. Rule schemata appear as conceptual devices devoid of any shape; schemata combine by sums and products and are visualized by predicates and assignments (more on this later in the paper). Furthermore, the symbolic forms that the recursive definitions of schemata assume appear all as atomic units that can combine in specific and well-constrained ways – a seemingly very different world from the world of shapes and the constant fusion that shapes invite.

But there are more ways that we can look at these sets of rules. Perhaps we could look at their usage and fit in creative design settings and in particular at their adaptation in studio, see for example [6–10]. Clearly some designers do things (that is, draw or make three-dimensional models) without being able to exactly describe what they are doing, i.e., whether a specific action they do is a particular transformation or operation. Still other designers opt for a more systematic approach (that is, they outline general principles of action) without knowing in advance how exactly they will use them to resolve the problem at hand. In that sense the usefulness of shape rules and rule schemata to capture specific actions that designers do (formal composition as visual process) or general principles that designers discuss of (formal composition as conceptual process) may be quite rewarding to pursue. And similarly the ability of shape rules and rule schemata to model formal strategies in design including bottom up processes determined within explicit, narrow contexts, or top down processes framed by open-ended principles applicable to wide variety of contexts, could also be rewarding to pursue. Intuitively, both types of formal systems are deployed in design to solve particular kinds of problems in spatial composition: shape rules are mostly deployed because of their visual specificity; rule schemata because of their conceptual generality. And still both shape rules and rule schemata are just alternative ways to describe the very same thing from a different vantage point.

The work here provides a tentative comparison between shape rules and rule schemata and attempts to show how these two modes of visual computation inform one another. A brief account of both forms of rules is given and a series of pictorial examples illustrate their similarities and differences. Both types of rules are seen within a general theory of computational design structured around the notion of a design algorithm $<a, k>$ where the input a is information for construction and the output k is the design description [11]. The account is necessarily fleeting and impressionistic and it is used primarily as a scaffold for a brief examination of both forms and the possibilities that each provides in design inquiry. The account is structured around two vectors pointing from shapes to schemata and from schemata to shapes. These two vectors are used here to structure the discourse and to suggest possibilities for merging seeing and reflecting in visual computation. The work concludes with a brief discussion of a computational framework that facilitates both views of design inquiry – the pictorial redescription of existing shape rules as explicit assignments in rule schemata and the modeling of rule schemata from scratch.

Two Directions

The underlying algebraic framework within which the shape rules and the rule schemata are defined, including the algebras of shapes U_{ij}, the algebras of labels V_{ij}, the algebras of weights W_{ij}, their combinations and the ways all facilitate computations of all sorts, has been given in various sources (see for example [1, 3, 12, 13]. The following discussion provides a brief overview of shape representation in shape grammars using shapes in a Euclidean space, but it equally well applies to labeled shapes and weighted shapes as well as parametric labeled shapes and parametric weighted shapes. An extended discussion of the formalism and especially the recent work on rule schemata is given in [3].

Shape Rules

A shape rule consists of a pair of shapes. Shapes consist of four basic elements: points, lines, planes and solids and their combinations. The basic elements and the shapes they define are readily described using linear equations and higher degree equations and the resources and conventions of analytic geometry. These descriptive devices are enough to capture all shapes, from the rudimentary polygons and polyhedra described by linear equations, to higher degree equations representing curves, b-splines, NURBS and so forth. The basic elements are related one to another through boundary conditions. A three-dimensional solid is bounded by two-dimensional planes. A two-dimensional plane is bounded by one-dimensional lines. An one-dimensional line is bounded by zero-dimensional points. And the zero-dimensional points have no boundaries (and no parts). Moreover, shapes are always defined in a Euclidean space that has a dimension equal or bigger than the dimension of the basic elements that make the shapes – the shapes are always parts of the Euclidean space they are defined in. The structure is quite elegant: A solid can be a defined only in a three-dimensional Euclidean space. A plane can be defined in a two- and/or three-dimensional Euclidean space. A line can be defined in a one-, two- and/or three-dimensional Euclidean space. And a point can be defined in any dimensional Euclidean space up to three dimensions. These elements combine to produce a generous structure of ten spatial systems U_{ij}, for i = basic elements and j = dimension of space, with specific algebraic attributes [2]. The ten algebras of shape are presented in Table 2.

A shape consisting of lines and defined in the two-dimensional Euclidean space in the algebra U_{12} is given in Fig. 1.

Any pair of specific shape instances A and B determines a shape rule. The notation of a shape rule follows the convention of an arrow (→) separating the two shapes, on the left and right hand side of the rule, with the additional convention of registration marks (+) to fix the spatial relation between them. An example of a shape rule is shown in Fig. 2.

Table 2 The ten algebras of shape

U_{00}	U_{01}	U_{02}	U_{03}
	U_{11}	U_{12}	U_{13}
		U_{22}	U_{23}
			U_{33}

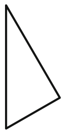


Fig. 1 A two-dimensional shape containing lines

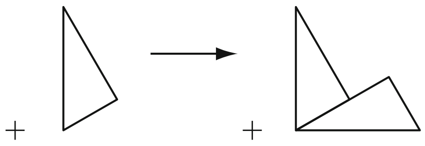


Fig. 2 A shape rule

Symbolically, for the two shapes *A*, *B* the shape rule is expressed as:

$$A \rightarrow B$$

Shape rules apply in a design process when there is a match of the rule to a part of the design at hand. The left hand side of the rule shows the shape that is matched in the design. The right hand side shows the shape that substitutes the shape that has been matched by the left hand side of the rule. If there is no match, then the rule cannot be applied in the particular design context. More technically, for shapes *A*, *B*, *C*, the shape rule $A \rightarrow B$ can apply to a shape *C* whenever there is a transformation *t* that makes the shape *t*(*A*) part of the shape *C*. If the shape *t*(*A*) is part of the shape *C* the rule subtracts the shape *t*(*A*) away from the shape *C* and replaces it by the shape *t*(*B*). The resulting shape C' and the corresponding computation are given then as:

$$C' = [\mathrm{C} - t(A)] + t(B)$$

The application of the rule is distinguished from the expression of the rule itself by the convention of a double arrow (=>) showing to the left the initial shape *C* and to the right the derived shape C', after the application of the rule. The sequence of shapes (designs) generated by the rule $A \rightarrow B$ in the manner shown above is symbolically expressed as $C => C' => C'' => \ldots => C'^{\cdots}{}'$. The same sequence of shapes may be taken as a finite set of shapes that all are productions of the rule $A \rightarrow B$. In this sense, the finite sequence (derivation) of the shapes $C, C', C'', \ldots,$ $C'^{\cdots}{}'$ has as members shapes that all share as their common property that they are all productions of the same shape rule. All sequences of shape rule applications bring to the foreground the compositional machinery (rules) used to produce the

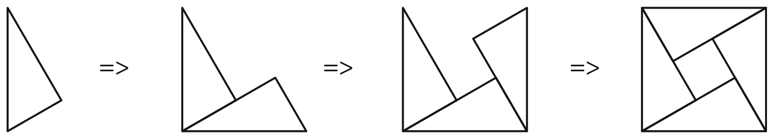


Fig. 3 A visual computation with the shape rule of Fig. 2

design. For example, a sequence of applications of the shape rule in Fig. 2 can generate a sequence of designs shown in Fig. 3.

Rule Schemata

A rule schema consists of a pair of schemata. Schemata are comprised by variables, and/or combinations of sets of variables. Specific parametric shape instances can be determined when values are assigned to these variables by an assignment *g* restricted in some way by a predicate. The values assigned to the variables are shapes consisting of points, lines, planes and solids and any combination of them (as well as labeled and/or weighted shapes). Different assignments define different shapes and additional conditions can be added at will to define families of shapes with specific attributes. On the other extreme, constant values may be assigned to variables to define a single representation. For example, the shape in Fig. 1 can be defined in a schema that is restricted by the predicate $g(S_1)$: S_1 is a triangle

the set of variables

S_1 (L_1, L_2, L_3)
L_1 (V_1, V_2)
L_2 (V_2, V_3)
L_3 (V_3, V_1)

and the list of values of the assignment *g*

V_1 $(0, 0)$
V_2 $(3.46, 2)$
V_3 $(0, 8)$

Clearly different predicates can change the attributes and the number of assignments on the schema, thus specifying different shapes. And furthermore, different assignments of numeric values to the variables can specify different shape instances. Three shape instances for different assignments are given in Fig. 4. The assignments are: g_1: V_1 $(0, 0)$; V_2 $(3.46, 2)$; V_3 $(0, 8)$; g_2: V_1 $(0, 0)$; V_2 $(3, 6)$; V_3 $(0, 8)$; and g_3: V_1 $(0, 0)$; V_2 $(0, 9.24)$; V_3 $(4.62, 8)$ respectively. The initial one is the one illustrated in Fig. 1.

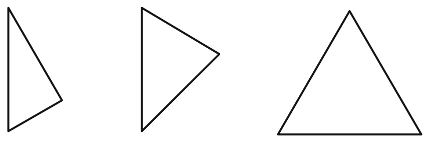


Fig. 4 Three different instances of the schema

Any pair of schemata can determine a rule schema. A symbolic expression of a parametric rule schema is given as:

$$x \rightarrow y$$

Rule schemata are formal generalizations of rules that specify a particular treatment for an entire family of shapes instead of specific shape instances. The variables in the pair of parametric shapes x and y are assigned values by an assignment g, the properties of which are determined by a predicate, specifying a certain class of shapes. When specific shapes are defined by g the schemata x and y become the shapes $g(x)$ and shape $g(y)$ respectively, and the rule schema is recast as a shape rule. Different constraints expressed in the predicate may lead to the formation of more or less constraint parametric shape rules. The constraints determining the instantiation of schemata to shapes do not affect the shape instances themselves, and do not determine how these shapes partake in spatial composition. Hence, visual ambiguity is preserved, shapes remain structure-less, rules unconstrained, and their productions open to interpretation. This makes descriptive (symbolic) precision and spatial (visual) ambiguity simultaneously possible in the same process of calculating. A symbolic expression of the resulting rule is given as:

$$g(x) \rightarrow g(y)$$

For example, the shape rule in Fig. 2 can be defined in a rule schema $x \rightarrow x + t(x)$, whereas x is a parametric triangle and $t(x)$ an isometric copy of the parametric triangle in a specific spatial relation to the initial parametric triangle. Different rule schemata can be formed after spatial relations between any type of triangles, or parallelograms, trapezoids, quadrilaterals, pentagons, hexagons and any other shape desired.

Rule schemata apply in a design process when there is a match of the assignment of the parametric shape at the left hand side of the rule to a part of the design at hand. If there is a match then the assignment of the parametric shape is substituted with the corresponding transformation of the assignment of this shape in the right hand side of the rule. More technically, for parametric shapes x, y, and a shape C, the rule schema $x \rightarrow y$ applies to the shape C whenever there is a transformation t that makes the shape $g(x)$ – for some assignment g that assigns values to the variables of x – part of the shape C. If the $t(g(x))$ is in fact part of shape C, then the rule subtracts the shape $t(g(x))$ from C and replaces it by the shape $t(g(y))$. The corresponding computation is given as:

$$C' = [C - t(g(x))] + t(g(y))$$

As before, the application of the rule schema is distinguished from the expression of the rule itself by the convention of a double arrow (=>) showing to the left the shape C and to the right the derived shape C', after the application of the rule. The design process and the productions of the rule schemata share the same conventions that apply in shape rule computation.

Back and Forth

There is a strong affinity between the two formal devices – and a tension too. The formal structure of both types of rules is identical. They are both determined by a pair of things in a relation: a pair of shape instances in the case of shape rules and a pair of schemata whose constraints and variables instantiate shapes. Intuitively this extra layer of abstraction, involving predicates and variables, suggests and invites a closer look.

The pivotal role and significance of predicates and variables become evident when a rule is given in a recursive form. In this form the variables of x and y may be recursively related to produce an indefinite number of symbolic expressions that associate x and y in desired ways. For example, if y and its variables in the right hand of the rule is a transformation t of x and its variables in the left hand side of the rule, then the rule $x \rightarrow y$ can be rewritten as $x \rightarrow t(x)$. Alternatively, if y and its variables in the right hand of the rule is related through some operation, say a division d, with x and its variables in the left hand side of the rule, then the rule can be rewritten as $x \rightarrow d(x)$. In general, if the variables of x and y can be associated through some design operation f, then y becomes a function $f(x)$ and the rule schema can be rewritten in the form:

$$x \rightarrow f(x)$$

The question then is what are the possible operators f that can relate the two variables of x and y in meaningful and constructive ways. Clear candidates are: (a) the transformation operation t; (b) the boundary operation b; (c) the part operator p; and the division operation d. More could be envisioned, but more productively, more could be constructed from those through compositions and additions. A nice set of rule schemata to start the discussion is found in [4]. The rule schemata presented there illustrate a set of discrete design processes that can be taken individually, reversed when possible, and combined under addition and composition. A list of basic rule schemata and their inverses is shown in Table 3.

Table 3 A list of basic rule schemata

Schema	$x \rightarrow$	$x \rightarrow x$	$x \rightarrow t(x)$	$x \rightarrow b(x)$	$x \rightarrow p(x)$	$x \rightarrow d(x)$
Inverse	$\rightarrow x$		$t(x) \rightarrow x$	$b(x) \rightarrow x$	$p(x) \rightarrow x$	$d(x) \rightarrow x$

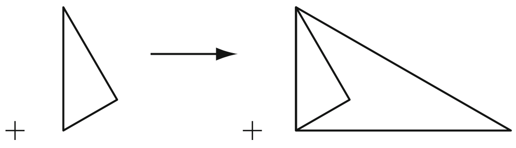


Fig. 5 A shape rule generating spiral patterns

The possibilities are bewildering. New combinations and products can be produced to structure shape rules that can be defined within them and to suggest new trajectories in design. For example, a rule schema like

$$x + t^R(x) \rightarrow x + t(x) + t^R(x + t(x))$$

could be very useful to account for the generative specification of a bilateral or rotational growth of modular patterns. And any other combination or product of variables might provide a useful structure to model shape rules. Rule schemata appear indeed to have a generative power because of their ability to form compositions and combinations in sequences that are potentially novel and meaningful in terms of the shape rules that these might be defined in. Still shape rules appear to resist the design interpretation that the rule schemata endow them. A constructive comparison between these two formal devices is briefly discussed below, in two sets of exercises. The first looks at an existing shape rule and infers rule schemata that this rule could be defined in. The second looks at an existing rule schema and postulates shape rules that can be defined within the schema.

From Shapes Rules to Rule Schemata

The trajectory from shape rules to rule schemata is straightforward. In this inquiry shape rules are considered as instances of particular assignments in rule schemata, and in extension as pictorial instances of particular rule schemata. Any shape rule from existing shape grammars and any shape rule constructed from scratch could do to illustrate this inference. A nice set of shape rules to start the discussion is found in [14]. The shape rules presented in this work are divided in two sets. The first intents to produce existing designs (plus some additional designs that potentially belong in the same set). The second intents to produce novel things without paying attention to existing designs. The former types of rules are illustrated using squares and the latter triangles. A shape rule selected from this work is given in Fig. 5. All labels associated with the original shape rules are omitted here or rather are substituted by

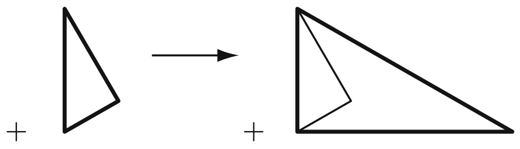


Fig. 6 A description of the shape rule of Fig. 4 in terms of the schema $x \rightarrow x + t(x)$

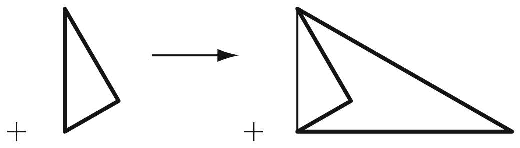


Fig. 7 A description of the shape rule of Fig. 4 in terms of the schema $x \rightarrow x + y$

a singular cross to denote the fixing of the application of the rule in the Cartesian plane.

Clearly this shape rule can be described in a variety of ways. An intuitive reading could result in a description given by the rule schema $x \rightarrow x + t(x)$. Here x is a right-angle triangle, t a similarity transformation including scales, reflections and rotations about edges and/or vertices of the triangle and $t(x)$, a similar copy of the initial triangle. The context of the original paper clearly suggests that the shape rule above (and the rest of the shape rules in the paper) all illustrate aspects of an additive process in design. For every shape in the left hand side of the rule, an isometric and/or scaled copy of the shape is added in the right hand side in a specific spatial relation to the former one. The shape rule of the example shows a possible way that a triangle x can be combined with a similar copy of itself $t(x)$ so that the short side of the large copy matches the long (hypotenuse) of the original triangle. The description of this shape rule in terms of the schema $x \rightarrow x + t(x)$ is given diagrammatically in Fig. 6.

The description of the shape rule in terms of a rule schema can easily be cast in alternative ways. For example, the rule can be recast as $x \rightarrow x + y$, whereas y some other shape arbitrarily related to the initial shape x. A possible interpretation of the added shape y could be a concave quadrilateral carefully chosen to match two of its edges to the small and medium sides of the initial triangle x. The description of the shape rule of Fig. 4 in terms of the schema $x \rightarrow x + y$ is given diagrammatically in Fig. 7.

It is interesting to note that the added shape y need not be a gestalt shape, say, the concave quadrilateral above. The two longer lines of this quadrilateral could do it too. In this sense the emphasis seems to shift from the addition of a closed polygonal shape to the addition of an open polygonal shape y that does not share any edges or part of edges with the initial triangle in the left hand side of the rule. A different way of casting this rule could start from the selection of a point outside the initial triangle and its joint with two lines y and $t(y)$ with two of the vertices of the triangle in the left hand side of the rule. The description of the shape rule of Fig. 4 in terms of the schema $x \rightarrow x + y + t(y)$ is given diagrammatically in Fig. 8.

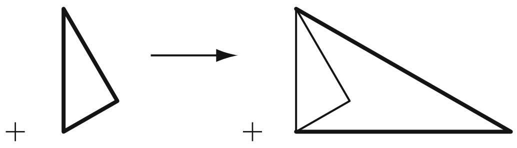


Fig. 8 A description of the shape rule of Fig. 4 in terms of the schema $x \rightarrow x + y + t(y)$

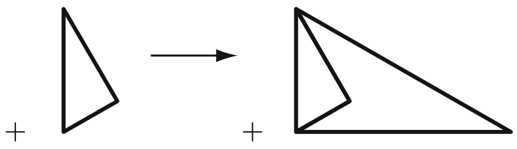


Fig. 9 A description of the set of shape rule of Fig. 4 in terms of the schema $p(x) \rightarrow x$ or alternatively $x \rightarrow p^{-1}(x)$

And this is not all. The rule can also be recast as $p(x) \rightarrow x$, for $p(x)$ a part of a shape x, and x a shape. In this case, the shape $p(x)$ in the left hand side of the rule is the right-angle triangle, and the shape x in the right-hand of the rule is a shape that has the shape illustrated in the left hand side of the rule as it s proper part. This rule schema can be alternatively cast as $x \rightarrow p^{-1}(x)$, for $p^{-1}(x)$ the inverse of $p(x)$, meaning that a shape x goes to a shape with x as a part [3]. The algebraic notation of $p^{-1}(x)$ might alienate the visual thinkers but it provides a uniform treatment in the classification of the basic schemata and consistency in notation too. In this case, the shape x is the right-angle triangle and the $p^{-1}(x)$ is the shape that has the right-angle triangle as its proper part. The description of the shape rule of Fig. 5 in terms of the schema $x \rightarrow p^{-1}(x)$ is given diagrammatically in Fig. 9.

From Rule Schemata to Shape Rules

The trajectory from rule schemata to shape rules is straightforward too. In this inquiry rule schemata are considered individually or in various combinations and/or compositions and shape rules are introduced that are restricted in some way by a predicate and a set of assignments to pictorially instantiate the rule schemata. Any rule schema from existing rule schemata classifications could do to illustrate this inference. A nice set of rule schemata to start the discussion is found in [4]. The rule schemata presented there encapsulate a set of discrete processes in design – that can be taken individually, reversed when possible, and combined under addition and composition. In the following example, a schema already encountered in the previous section is selected to help us draw comparisons between the exercise in the previous section and the exercise in this section.

$$x \rightarrow x + t(x)$$

This schema is perhaps the most frequent deployed in shape grammar discourse: it specifies to add a transformed copy $t(x)$ of a shape x in a specific way constrained by some predicate contained in the schema. The schema could be intuitively cast as:

"Look at a design, find a part x that is of interest to you, and repeat it in some way". More formally, the rule schema could be recast as: "look at a design A and if there is a transformation T such that the shape x is part of A, then replace the occurrence of shape $T(x)$ in A with the shape $T(x) + T(t(x))$ or better, with the shape $T(x + t(x))$.

An indefinite number of shape rules can be defined based on this rule schema. One way to look at the possible classification of shape rules fixed within this rule schema is to look at the spatial relation between the initial shape x and the transformed copy of the shape $t(x)$, and the transformation t under which the copy $t(x)$ was constructed. The possible spatial relations between the shape x and its copy $t(x)$ can be classified in families of spatial relations with respect to the dimensionality of the basic elements that comprise the spatial relation between the two shapes. For example, for a shape x a right triangle and an isometric copy of itself $t(x)$, there are four sub-families of spatial relations that can be defined between the right triangle x and its copy $t(x)$: the two triangles x and $t(x)$ may share the empty shape, a point, a line or a plane. Clearly for each of these conditions there are many more sub-conditions to discern with respect to the spatial transformations that specify the spatial relation; i.e., whether the transformation is, say, a translation, a rotation, a reflection, a glide reflection and so forth [15, 16]. In all cases, these spatial relations provide the blueprints for the specifications of the shape rule in the rule schema. One instance of a shape rule for each of these four families of spatial relations is shown in Fig. 10.

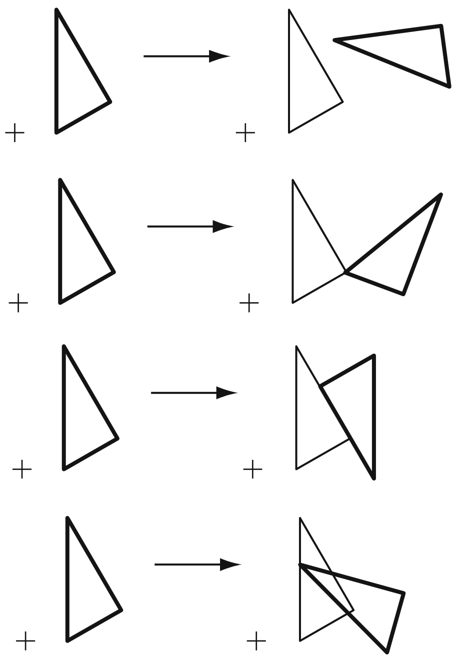


Fig. 10 Four shape rules defined with the rule schema $x \rightarrow x + t((x)$ that satisfy the predicate x is a 3-gon and t an isometric transformation

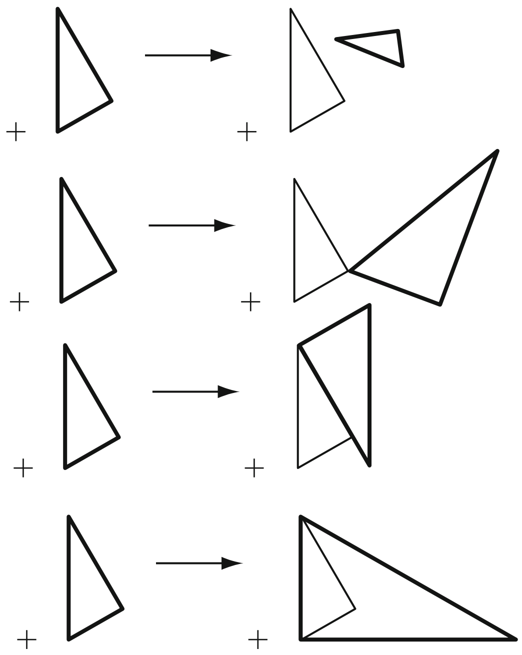


Fig. 11 Four shape rules defined with the rule schema $x \rightarrow x + t((x)$ that satisfy the predicate x is a 3-gon and t a similarity transformation

Note that these four families of spatial relations can be nicely expressed as intersections (.) of the basic spatial elements of the shapes, their boundaries and their boundary inverses too. More specifically, in the first spatial relation, the intersection $x.t(x)$ of the shapes x and $t(x)$ is the empty shape. In the second spatial relation the intersection $b(x).\,b(t(x))$ of the boundaries of the shapes x and $t(x)$ is a single point (basic element in the algebra U_{02}). In the third spatial relation the intersection $x.t(x)$ of the shapes x and $t(x)$ is a single line (basic element in the algebra U_{12}). And in the fourth spatial relation, the intersection $b^{-1}(x).b^{-1}(t(x))$ of the inverses of the boundaries of the shapes x and $t(x)$, is a single plane (basic element in the algebra U_{22}). It should be noted that in the last case the shape $b^{-1}(x)$ is part of $b^{-1}(t(x))$, a condition that implies the definition of the part relation $\leq$.

The families of shape rules that can be defined within the rule schema $x \rightarrow x + t$ (x) can be significantly extended with respect to the transformation t that specified the geometry of the $t(x)$. Figure 11 shows samples of the four families of shape rules defined in this schema for t a similarity transformation, that is, a scale transformation combined with any isometric transformation including any combinations of translations, rotations, and reflections.

The next families of transformations that may be added on the similarity transformations of the Euclidean space are the affine, linear, and topological transformations of the so-called non-Euclidean space [17]. Figure 12 shows samples of four families of shape rules defined in this schema for t an affine transformation, that is, a stretch/compress transformation combined with any similarity transformation, and $t(x)$ an affine copy of the shape x.

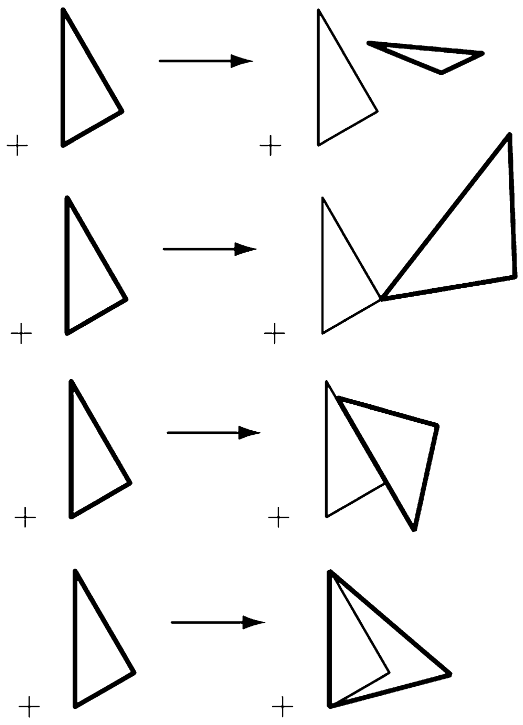


Fig. 12 Four shape rules defined with the rule schema $x \rightarrow x + t((x)$ that satisfy the predicate x is a 3-gon and t an affine transformation

Still the expressive power of the rule schema can go beyond the transformational property of the spatial relations between the shape x and its copy $t(x)$. If for example, the shape x is defined as any n-gon, then a long list of possible families of shapes can be used to define shape rules within this rule schema, all very different from the ones seen so far. Within this framework, spatial relations between say, squares, rectangles, parallelograms, rhombi, kites, trapezoids, quadrilaterals of all sorts, and so forth, are all spatial relations between a shape x and a shape $t(x)$, for a shape x and $t(x)$ any of those and t a Euclidean or parametric transformation. Pentagons and hexagons and heptagons and so forth, all wait to be tried for they provide an inexhaustible really list of visual conditions to explore. Figure 13 shows samples of the four families of shape rules defined in this schema for x a square, t a scale transformation combined with any isometric transformations and $t(x)$ a similar copy of x.

Clearly, the predicate can be as elaborate as desired. The quest for provision of tools to facilitate the construction and instantiation of such spatial relations is an ongoing project in the design of software packages geared for visual composition. A comprehensive treatment of such rules and the taxonomies they will produce as pictorial illustrations of schemata and their products and sums is a welcome project for design inquiry. The goal here was to suggest such an inquiry and illustrate some initial first steps towards this direction.

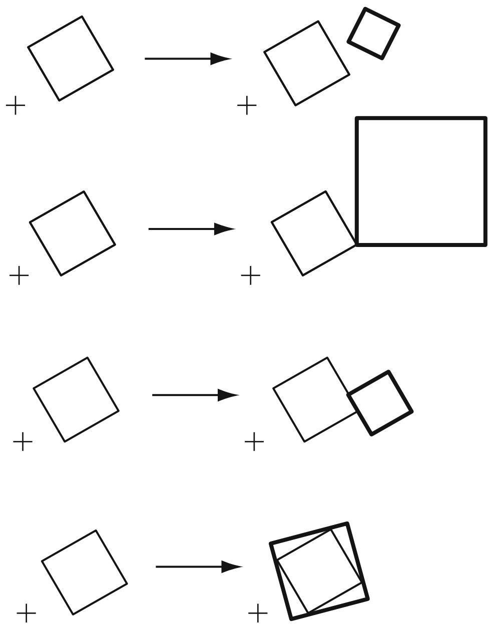


Fig. 13 Four shape rules defined with the rule schema $x \rightarrow x + t((x)$ that satisfy the predicate x is a 4-gon and t a similarity transformation

Discussion

Shape rules and rules schemata are useful to work with because of the compositional relations they foreground and the ways they facilitate distinct views of design inquiry. Both formal devices provide a rich repertory of means to support expressive and productive calculation in design respectively. And both provide powerful insight when they are contrasted one against the other and suggest new ways of interpretation. Intuitively, both types of formal devices are deployed in design to solve particular kinds of problems in spatial composition: shape rules are mostly deployed because of their visual specificity; rule schemata because of their conceptual generality. And still both shape rules and rule schemata are just alternative ways to describe the very same thing from a different vantage point.

An exciting aspect of the exercise of looking at existing shape rules and attempting to infer rule schemata that these shape rules could be defined in, is that such redescriptions of an existing rule, or set of rules, as predicates and assignments in rule schemata, provide novel descriptions of the given corpus of shape grammars. They also suggest interpretations of the existing sets of grammatical rules that may be potentially diverse and distinct from those envisioned from the authors of the grammars. In this sense this act of redescription of the pictorial rules of shape grammars as assignments in different schemata facilitates their novel re-appropriation and re-usage in alternative contexts. A possible corollary of this conclusion is that this shift in representation allows for the rules and the grammars to emerge above specific domains such as residential architecture, public architecture, ecclesiastical architecture, landscape architecture, ornamental design, furniture design, product design, automobile design and any subcategories within these fields. Instead this account focuses on the compositional schemata that can discursively explain what the shape rules do, and the problems they address.

An exciting aspect of the exercise of looking at existing rule schemata and attempting to define shape rules within them, is that such illustrations of the schemata in terms of shape rule instances, provide concrete descriptions of the given corpus of schemata. They may also suggest interpretations of these schemata that are potentially diverse, and even non-intuitive, with respect to the schemata. In this sense this act of redescription of the symbolic schemata as pictorial assignments facilitates their novel re-appropriation and re-usage in alternative contexts.

The major goal in this work has been to look at the pair of the shape rules and the rules schemata from either side foregrounding each in the relation. This back-and-forth between show and tell is what this is all about. In fact it is suggested here that such pictorial redescriptions of shape rules as assignments in rule schemata, and instantiations of schemata in visual symbols is the heart of design inquiry and that it should underlie any computational framework for design.

A significant motivation underlying this work has been the systematic inquiry on both aspects of rules so that they can both be implemented in shape rules and/or in assignments in rule schemata and be freely instantiated, edited, used and tested in an automated computer setting. The technical problems associated with the design of a framework to implement shape recognition and shape rules are formidable. A recent model based on an underlying graph theoretical representation of shape has successfully managed to address a great deal of these problems [18, 19]. The next step is the seamless implementation of shape rules in terms of rule schemata in an interactive framework that allows for the free definition of any shape rule none so ever and the immediate testing of its expressive power in the design at hand.

References

1. Stiny G (1980) Introduction to shape and shape grammars. Environ Plan B: Plan Des 7 (3):343–351
2. Knight TW (1994) Transformations in design: a formal approach to stylistic change and innovation in the visual arts. Cambridge University Press, Cambridge
3. Stiny G (2006) Shape, talking about seeing and doing. MIT Press, Cambridge, MA
4. Stiny G (2011) What rule(s) should I use? Nexus Network Journal 13:15–47
5. Langer SK (1957) Philosophy in a new key: a study in the symbolism of reason, rite, and art. Harvard University Press, Cambridge
6. Knight T (2000) Shape grammars in education and practice: history and prospects. Int J Des Comput 2:67
7. Economou A (2001) Shape grammars in architectural design studio. In: Mitchell W, Fernandez J (eds) Proceedings of the 2000 ACSA Technology Conference. ASCA, Washington, DC, pp 75–81
8. Kotsopoulos S (2005) Constructing design concepts: a computational approach to the synthesis of architectural form. PhD Thesis, Massachusetts Institute of Technology, Cambridge, MA

9. Kotsopoulos S (2008) From design concepts to design descriptions. Int J Archit Comput 06 (03):335–360
10. Kotsopoulos S (2005) A computational framework of composition, Proceedings of the 23th conference in education and research in computer aided architectural design in Europe. In: Duarte JP, Ducla-Soares G, Sampaio AZ (eds) Digital design: the quest for new paradigms. Technical University of Lisbon, Portugal, pp 515–536
11. Stiny G (1979) Algorithmic aesthetics: computer models for criticism and design in the arts. University of California Press, Berkeley
12. Krstic D (2001) Algebras and grammars for shapes and their boundaries. Environ Plan B: Plan Des 28(1):151–162
13. Krstic D (2005) Shape decompositions and their algebras. AIE EDAM 19(04):261–276, Cambridge University Press
14. Stiny G (1976) Two exercises in formal composition. Environ Plan B: Plan Des 3:187–210
15. March L, Steadman P (1974) The geometry of the environment. MIT Press, Cambridge, MA
16. Economou A (1999) The symmetry lessons from Froebel building gifts. Environ Plan B: Plan Des 26(1):75–90
17. Mitchell W (1990) The logic of architecture. MIT Press, Cambridge, MA
18. Grasl T, Economou A (2013) Unambiguity: difficulties in communicating shape grammar rules to a digital interpreter. In: Stouffs R, Sariyildiz, S (eds) Computation and performance: proceedings of the 31st eCAADe conference. Delft University of Technology, Delft, vol. 2, pp 617–620
19. Grasl T, Economou A (2013) From topologies to shapes: parametric shape grammars implemented by graphs. Environ Plan B: Plan Des 40(5):905–922

The Inference of Generic Housing Rules: A Methodology to Explain and Recreate Palladian Villas, Prairie Houses and Malagueira Houses

Deborah Benros, José Pinto Duarte, and Sean Hanna

Abstract This paper describes the setting out of generic housing rules. These rules were synthesized using a grammar formalism as described in previous papers. This study focuses on the parametric shape rules and its application.

Generic grammars were applied in works such as the urban generic grammar for the purpose of describing urban patterns. However the application of generic grammars to other scales has not been performed to date. A generic grammar is a formalism that allows the design of a diverse solutions, unlike a typical grammar which focuses on a specific design language. This work analyzed three languages: Palladian villas, Prairie and Malagueira houses and proposed a single grammar to replicate the examples. This work will showcase the set of generic rules and the strategy used to parameterize each shape rule. The contribution of the work lies in the way each rule is parameterized to cater for each language whilst the shape rule remains the same.

Introduction

This research paper describes the structure and set of generic shape rules of a generic shape grammar applied in housing. The grammar comprises eight stages: the first stage arranges the boundary setting out; the second is responsible for the spatial subdivision; the third stage wall thickening; the fourth stage the functional

D. Benros (✉)
University College London, London, UK
e-mail: deborahbenros@gmail.com

J.P. Duarte
TU Lisbon, Lisbon, Portugal

S. Hanna
Bartlett School of Architecture, University College London, London, UK

© Springer International Publishing Switzerland 2015
J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_23

assignment; the fifth stage the creation of adjacencies by connecting door creation; the sixth stage responsible for window design; the seventh stage boundary; and finally the eighth stage the termination with the deletion of labels. For the production of this generic grammar the previous comparative and analysis work was fundamental to set the standards and foundations.

We consider a generic grammar to be a grammar that allows the generation of a set of designs from multiple styles rather than defining a particular feature or style. If different shape grammars each recreate only a particular design corpus all of them are independent; however, a more generic description covering a number of separate grammars would allow relationships to be seen between them. A similar analogy can be placed to describe spoken languages, which is the analogy that inspired in the first place the idea of shape formulations. If several languages have separate and distinct grammars but in many cases share a common ground, and if some distinct languages also share simple phrase constructions, then a higher level grammar can be elaborated to describe that branch of languages.

In this instance an attempt was made to create a shape grammar that would encode the parameters required to design three types formerly considered independent: the Palladian villas, the prairie houses and the Malagueira houses. A new grammar that represented the different types was recreated, by considering the sequence of grammar rules and incorporating parametric functions. The grammar proposed does not fit the criteria of an unrestricted type of grammar because it implies an ordered structure and rule ordering process and clearly presents restrictions [6]. Knight (as quoted by Prats [9]) classified the three types of grammars as Additive, Grid and Subdivision. This grammar should be used intelligently if not assisted by a computer program. The function of buildings raises another issue. The question of typology and use constitutes a key factor in the choice of corpora, simply because different uses cannot be relevantly compared (e.g., housing and retail). Therefore, taking into account shape grammars that were previously inferred and available, and the relevance, program and type of use, housing seemed to be a good candidate.

An analysis was performed on three shape grammars selected with regards to bottom-up approach, containment of external fabric and type of grammar. The grammars were classified not only by the number of occurrences of shape rules but also by the rules that help design the house basic features. In the Palladian grammar the first stage and first nine rules determine the basic house layout. These rules help design a grid that will set standard for the spaces later detailed. In the Prairie houses the first third of the grammar allows for additive rules followed element by element to design the house. This is generated in a progressive way and the house grows in detail and in size. The Malagueira type showcases a radically different approach. The first steps are additive but soon change, such that all later rules related with spatial creation are subdivision rules, by which space is designed by sub-dividing a generic space and then detailing a portion by assigning function.

Once the grammar type is determined the type of house label is considered. Most of the house types selected coincide with the notion of single detached housing isolated in the plot (Malagueira is mostly semi-detached). Nevertheless, and regardless of the detachment condition, some of these houses are self-contained and packed within an external boundary and others are not inscribed or not contained. The first case of compact or contained houses are present in all house types studied.

This paper is composed of three sections. The first section reflects on the methodology used to create the generic rules and its parameterization, followed by a section that describes the validation process and the conclusion section where the results are discussed and future work described.

Methodology

A case study was selected among three different single housing languages: Palladian villas, Prairie and Malagueira houses. Previous work demonstrated how these pre-existing grammars were analyzed and their combined rule set derived [1, 2]. The most frequent rules were extracted and identified as generic rules.

The methodology is based on three tasks: the development of generic shape rules, the creation of a generic grammar formalism and lastly the development of specific parameterization to represent different languages. The strategy used to create the generic grammar followed certain principles:

1. Top-down approach
2. Self-contained strategy based on a polygonal boundary
3. Subdivision as a method to provide detail
4. Common shape rules to address the generation process
5. Variation and detail conferred by the parameterization in each shape rule

The first stages of design creation have a greater level of abstraction while the latest promote specific detailing. In addition the first stages propose a more abstract formulation where only the house floorplan outline is illustrated and the latest stages confer specific detail.

The generic grammar developed proposes a total of eight stages.

As illustrated in, Fig. 1, the first stage is prompted by the incorporation of the polygonal boundary. This boundary and its proportions vary from language to language but at this point a great range of de-signs is allowed. Stage 2 promotes subdivision. With subdivision the first zoning exercise takes place and the first spaces are created. Sub-division is a process commonly used in housing design. Its use is intuitive and prescriptive and for that reason was chosen as primary meth-od of design in the generic grammar. Stage 3 focuses on space merging or concatenation, allowing adjacent rooms to be merged creating different geometries. Stage 4 provides wall thickening by an offset process. The original abstract lines are doubled according to the tectonic nature of the housing language.

These four stages constitute the common branch and the true generic body of the grammar. This level of abstraction can be easily observed since the design at this point is still mono dimensional and representative. It is not conferred with wall

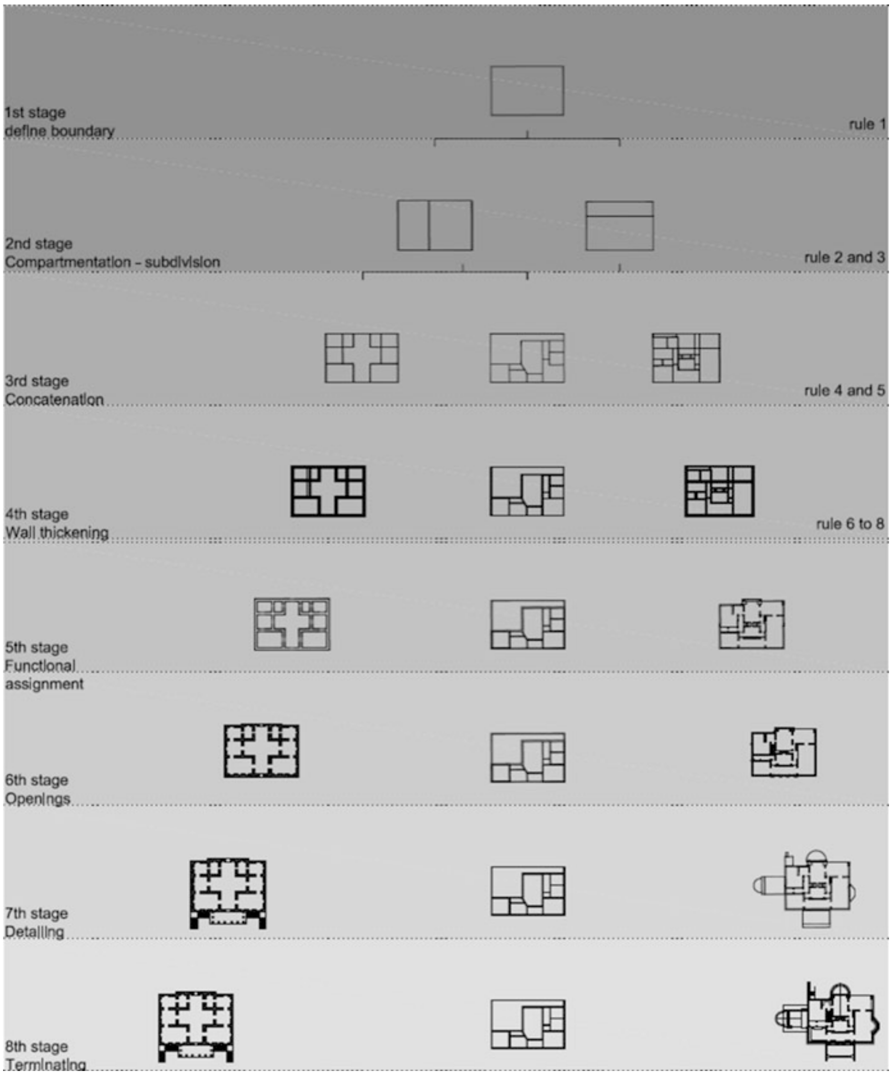


Fig. 1 Generic grammar tree diagram

thicknesses, openings or functions. It is a simple representation of a schematic house floorplan.

Stages 4–8 propose similar formulation but are specific to each language and have particular shape rules—the attempt has not yet been made at generic rules for these. Respectively they are: stage 5 functional assignment, stage 6 inclusion of openings, stage 7 detailing, stage 8 completion.

The diagram below illustrates the application of the generic grammar. The common branch is clearly illustrated on top with a similar containing shape branching out to three district solutions. Illustrated on the left, the Palladian villa 'la Malcontenta', in the middle the Malagueira house type Ab as described by Duarte [4], and on the right the Winslow prairie house by Frank Lloyd Wright.

The tree diagram shows the application of the generic grammar to replicate three different house languages, all of them part of the pre-existing corpus of the styles. It illustrates how each solution can be created and developed in order to effectively describe an output of each language.

The development of generic shape rules involved the observation and synthesis of some of the most common shape rules used by grammarians. Previously inferred shape grammars focused on particular languages of design such as the Palladian villas [10], the prairie houses [7], the Malagueira houses [5], the Wren city churches, buffalo bungalows and Queen Anne houses to name a few.

From a structural point of view these grammars follow either a grid, addition or subdivision methodology.

Despite the differences between the discussed grammars with regard to their structure and top-down or bottom-up approach, all rules can be reduced to a set of shape rules that obey either addition, subdivision, concatenation, subtraction or replacement.

Previous work showed how these rules are applied showcasing various differences but expressing a similar essence [1, 2]. This can be represented by schemas (as introduced by [11]). Schemas try to trans-pose the graphic description into a simple algebraic expression. This allows for a certain level of abstraction while applying a rule. Schemas represent the shape rule without using graphical symbols. Often grammar users get stuck into a graphical representation and restrict the use of a particular rule but its abstract notion can avoid these misconceptions.

So for the four rules identified a specific schema is proposed:

1. Addition:

$$\emptyset \rightarrow \mathrm{X}$$
$$\mathrm{X} \rightarrow \mathrm{X} + \mathrm{t}(\mathrm{X})$$

2. Subdivision:

$$\mathrm{X} \rightarrow \mathrm{div}(\mathrm{x})$$
$$\mathrm{X} \rightarrow \mathrm{prt}'(\mathrm{x}) + \mathrm{prt}''(\mathrm{x})$$
$$(\mathrm{X}' + \mathrm{X}'') \rightarrow \mathrm{prt}(\mathrm{b}(\mathrm{X}' + \mathrm{X}''))$$

3. Concatenation:

$$(\mathrm{X}' + \mathrm{X}'') \rightarrow \mathrm{X}$$
$$\mathrm{prt}'(\mathrm{x}) + \mathrm{prt}''(\mathrm{x}) \rightarrow \mathrm{X}$$

4. Subtraction:

$$X \rightarrow X - \text{prt'}(x)$$
$$(\text{prt'}(x) + \text{prt''}(x)) \rightarrow \text{prt'}(x)$$

In addition to the identified generic rules specific parameterization was developed. Along with the graphic representation and each schema the parameterization for each rule was specifically developed to cater to each language. In this study three languages were used as case studies, which led to three different expressions with particular variables catered for each language. This constitutes the novelty of this generic grammar and what allows the generic rule to work when applied particularly to each family of solutions. The generic shape rules and their parameterization are clearly presented in Table 1. Exemplified are stages 1–4 and rules 1–8 which constitute the common branch. For each generic shape rule a common graphical representation is pro-vided and a specific algebraic expression presented for each language of the case study: Palladian villas, Prairie and Malagueira houses. A good example of the addition rule is rule 1. This rule introduces the first shape into the design. This shape follows a specific parameterization presented for each one of the three styles. This first rule showcases how the expressions work. The graphical rule introduces a rectangular boundary defined by the width (x) and height (y). The Palladian villas adopt specific rectangular ratios, commonly 2:3, 3:4, 3:5 and others predefined by the language. On the other hand Malagueira houses apply a specific fixed ratio that defines the available house plot of 12 × 8 m (a 2:3 ratio). This first generic rule (boundary addition) constitutes an example of an addition rule with a standard schema ø → X. The parameterization is then targeted to provide the correct areas, ratios and proportions of each style. It is evident from Palladio's extensive descriptions, drawings from 'Il Quattro libri' and observation of the existing corpus that the Palladian villas allow ratios of 1:1, 3:2, 4:3, 3:5, 4:5 and 3:7.

This mandatory rule will be used by all types and therefore the labels PAL, PRA, and MAL will be applied symbolising respectively Palladian villa, Prairie or Malagueira house. This parametric rule designs a X × Y rectangle with specific formulations and ratios for each type as described by the rule schema:

$$\emptyset \rightarrow X$$

To an existing shape X, you introduce a new shape that can translate a transformation on the initial design stage:

$$X \rightarrow X + t(X)$$

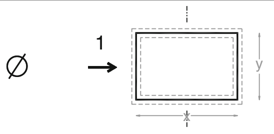
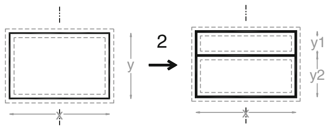
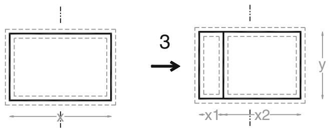
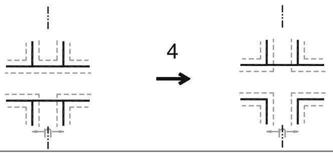
Respectively:

PAL: $y/x = n/m \rightarrow m = [3,5,7] \rightarrow n = [2,3,4,5] \rightarrow$ allowed ratios: 1:1, 2:3, 4:3, 3:5, 4:5, 3:7

PRA: [x, y]

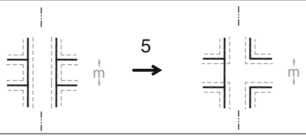
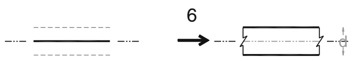
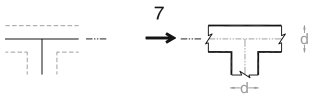
MAL: $X = 8$ m, $Y = 12$ m $\rightarrow$ allowed ratio: 2:3

Table 1 Generic grammar shape rules

Stage 1	Boundary definition	
Rule 1	Adding boundary	Parameterisation
Palladian villas		Y/X = N/M
		M = [3, 5, 7]
		N = [2,3,4,5]
		Permitted ratios: 1:1, 2:3, 4:3, 3:5, 4:5, 3:7
Prairie houses		[X, Y]
Malagueira houses		X = 8 M
		Y = 12 m
		Permitted ratio: 2:3
Stage 2	Spatial subdivision	
Rule 2	Horizontal subdivision	Parameterisation
Palladian villas		Y/X = N/M
		M = [3, 5, 7]
		N = [2,3,4,5]
		Y = Y1 + Y2
		Y1 = Y/N V Y1 = Y/3n
		Y1 = Y2 (for N = 2 V N = 4)
Prairie houses		[X, Y]
		Y = Y1 + Y2
Malagueira houses		Y1 = Y2 = Y/2 V
		Y1 = N X Y/4 N = [1–4]
Rule 3	Vertical subdivision	Parameterisation
Palladian villas		y/x = n/m
		m = [3, 5, 7]
		N = [2,3,4,5]
		X = 2.X1 + X2
		X1 = N V X1 = 3.N/2 V X2 = 2n
		Y1 = Y2 (For N = 2 V N = 4)
Prairie houses		[X, Y]
		X = X1 + X2
Malagueira houses		X1 = X2 = X/2
Stage 3	Space merging	
Rule 4	Horizontal merging	Parameterisation
Palladian villas		X = N
Prairie houses		
Malagueira houses		
Rule 5	Vertical merging	Parameterisation
Palladian villas		Y = m
Prairie houses		

(continued)

Table 1 (continued)

Malagueira houses		
Stage 4	Wall thickening	
Rule 6	Single wall	Parameterisation
Palladian villas		D ≈ 2 vicentine feet
		D = 600 Mm
Prairie houses		D Ext ≈ 100 + 1/4''
		D Int = 100''
Malagueira houses		D Ext ≈ 250 Mm
		D Int = 200 Mm
Rule 7	Wall 'T' junction	Parameterisation
Palladian villas		D ≈ 2 vicentine feet
		D = 600 Mm
Prairie houses		D Ext ≈ 100 + 1/4''
		D Int = 100''
Malagueira houses		D Ext ≈ 250 Mm
		D Int = 200 Mm
Rule 8	Wall corner junction	Parameterisation

The same principles apply for the subdivision rules. The graphical representation of rules 2 and 3 describe the generic subdivision rule. Both rules propose subdivisions thereby allowing diverse solutions. The allowance of the subdivision process is conditioned by the parameterization. The prairie houses impose particular restrictions which confer symmetry. The 'vertical subdivision' is ruled by the bi-symmetrical principle imposed on the floorplan, and therefore the subdivision is replicated symmetrically. On the other hand floorplans like the Malagueira obey specific proportion ratios such as 1:2, 2:3 and 3:4.

The second stage is responsible for the basic layout of the house floorplan. This stage proposes four shape rules. Half of the rules proposed are subdivision rules and are responsible for the generation of great part of the house fabric. The reminder are merging rules that deal with particular conditions and help designing spaces with more complex geometries by spatial concatenation. Shape rule 2 is responsible for the horizontal subdivision. This type of subdivision can be placed in any of the three house types and allows the creation of two separate spatial/functional zones by splitting the space horizontally. Regardless of the house type the rule schema can be represented by the following expressions:

$$X \rightarrow \mathrm{div}(x)$$

$$X \rightarrow \mathrm{prt}'(x) + \mathrm{prt}''(x)$$

$$(X' + X'') \rightarrow \text{prt}\,(b(X' + X''))$$

Therefore the resulting rule parameterisation can find intervals in:

$$N : [x, (y' + y'')]$$

Similarly rule 3 is responsible for the creation of two separate spaces using a subdivision method. The difference lies in the direction of the split, in this case placed vertically. In normal circumstances the rule could be equally applied for the three case scenarios, however the Palladian villas pose some singularities which require special address. The issue of symmetry patent in the Palladio language requires that a vertical split has to be performed and copied symmetrically across the floorplan using the North to South direction as an axis. Therefore the rule has to allow for the proper parameterisation of this case.

$$N : [(x' + x''), y]$$

Or in the Palladian grammar:

$$N : [2.(x' + x''), y] \ldots$$

The concatenation rule is represented on the third stage by rules 5 and 6. These represent a space merging operation by the deletion of one border. This rule is commonly used by designers to generate spaces with a certain degree of geometric complexity and is used frequently in grammars (ref). This rule which uses a schema similar to $\text{prt}'(x) + \text{prt}''(x) \rightarrow X$ allows several specificities such as the parameterisation proposed in Table 1.

The latest stages propose language specific shape rules. The development of parameterisation caters to particular housing detailing features. Within these rules can be found:

1. Particular features for Palladian language
2. Particular features for Prairie language
3. Particular features for Malagueira language

Derivation

The recreation of three original designs from start to finish by the phased application of the shape grammar rules is illustrated in Figs. 2, 3, 4, and 5. In this experiment three existing houses designed by the original architects were selected to illustrate the generic grammar. Villa Malcontenta is an example of a typical Palladian villa, the Winslow House, one of Wright's most famous creations, illustrates the existing corpus of Prairie houses and the Malagueira two-bedroom

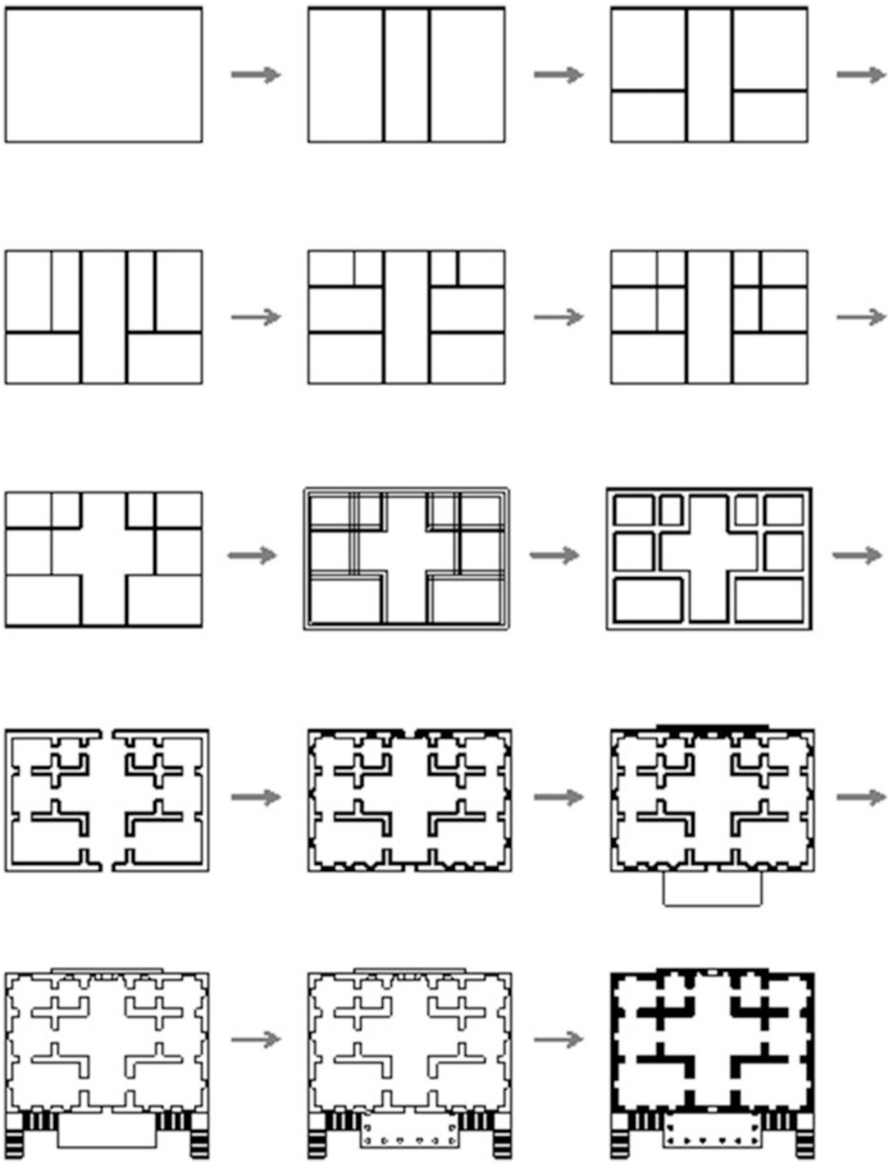


Fig. 2 Generic grammar derivation – Palladio's Villa La Malcontenta

Ab type house (according to Duarte's labelling) exemplifies a typical Malagueira family housing unit [5].

La Malcontenta, Fig. 2), was originally designed, built and completed in Venice's outskirts between 1559 and 1560 and is pictured in the 'Il quatro libri' [8]. Its orthogonal features and grid-like floor plan features a matrix that resembles

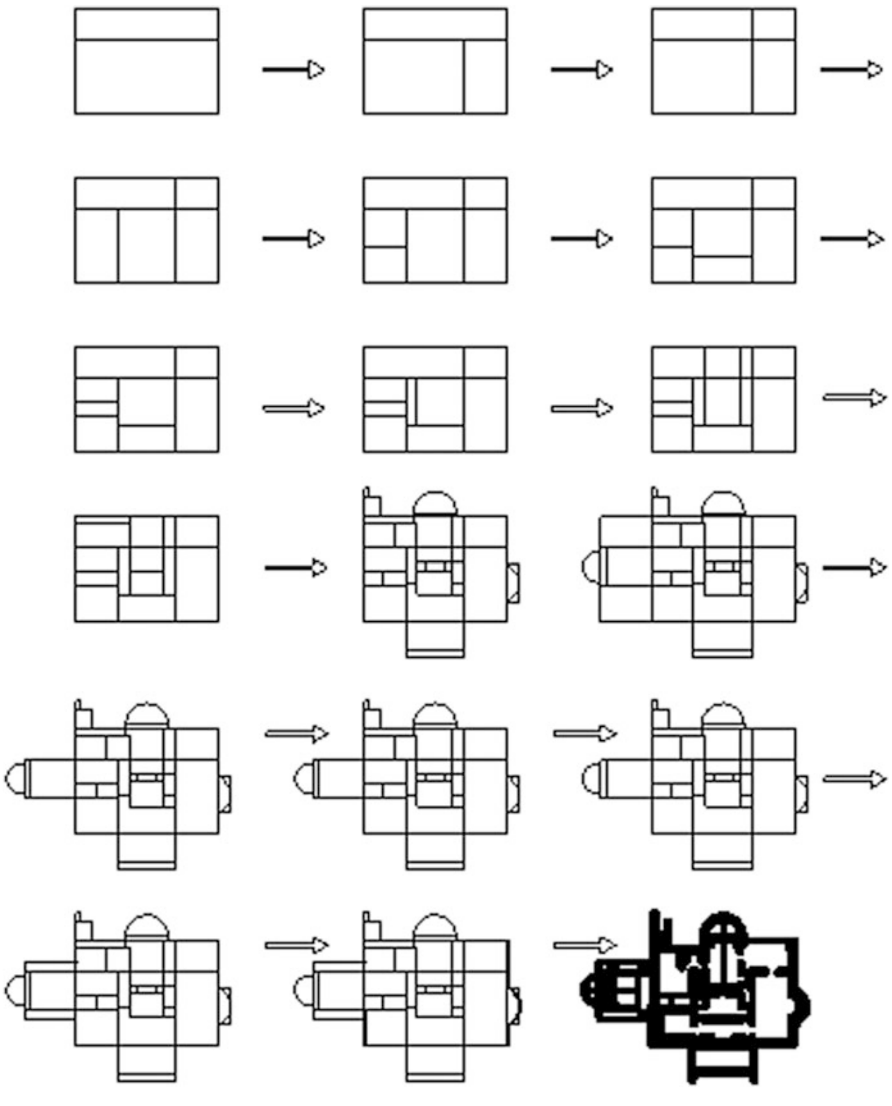


Fig. 3 Generic grammar derivation – Prairie Winslow house

a 5×3 grid organisation. Whereas the original grammar used a grid process, achieving the same design with subdivision allows us to economise on certain steps (namely extensive concatenation). The envelope is thus designed and established from the start. As shown, this subdivision is doubled to address the symmetrical nature of the design. Steps 3–6 use the division rules 2 and 3 recursively (in the case of Rule 3, repeated again and again). Steps 7 and 8 start the space merging or concatenation process. This is a fundamental step for spatial

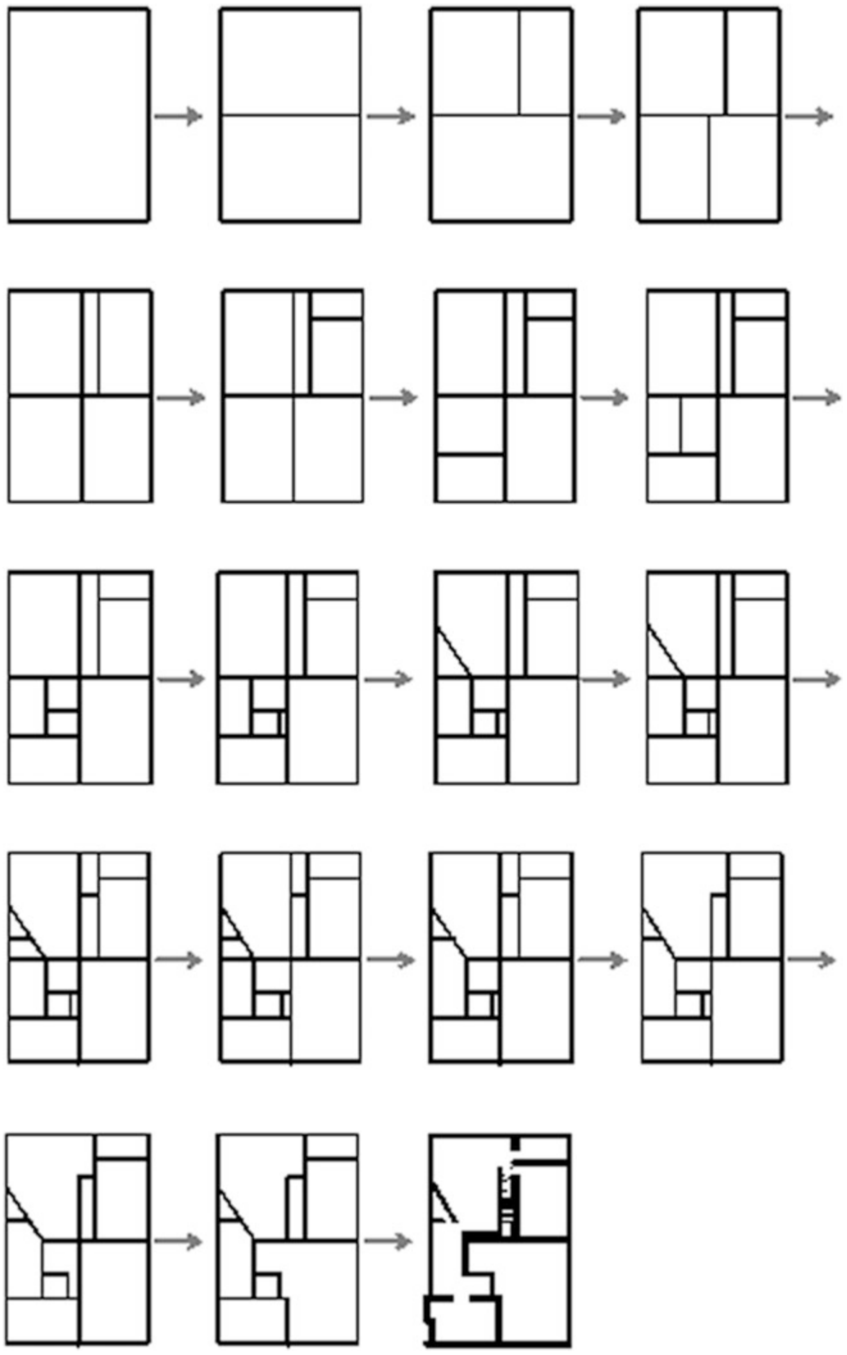


Fig. 4 Generic grammar derivation – Malagueira house Ab

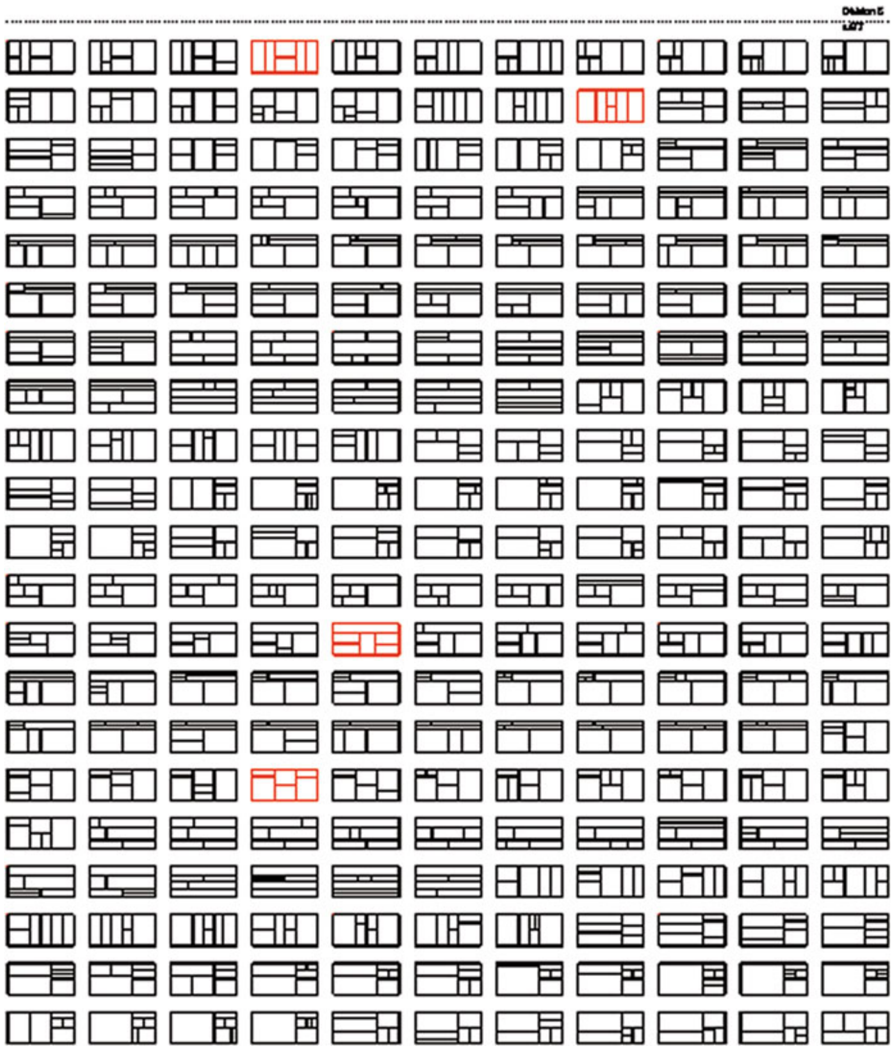


Fig. 5 Generic grammar corpus of solutions for five divisions (part 1)

configuration in a Palladian villa. In comparative terms, the derivation of the Malcontenta using this alternative method proves to be faster.

The Winslow house (Fig. 3) constitutes an appropriate case of prairie houses because is mainly self-contained. Contrary to most of the typical layouts of prairie houses where normally a crossed (or butterfly shaped) array is performed, Winslow could be underlined by a rectangular bounding shape. The inside layout could be easily described by a series of orthogonal axes very much alike a grid or matrix. This was also one of the first houses of the style used in this study as a case for derivation and possible conversion from additive grammar to subdivision. There are some resemblances shared with a Palladian typology: the rectangular outline, the grid like interior, the use of orthogonal elements, the emphasis on the social area of the house which occupies the core of the dwelling and extends itself through communicating rooms towards the outside creating progressive areas of privacy versus exposure, the careful and strategic addition of external elements that host entrance points and terraces occupying the main symmetry axis.

The derivation of Winslow can be summarized in 16 steps from start to completion. The first step is the boundary settlement, a bounding rectangular shape that already abstractly describes the final outcome. The second step uses a typical rule used in the Palladian alternative grammar. It is a rule to ensure the symmetry between the east and west wings. This subdivision rule proposes vertical divisions by placing two vertical cuts through the outline created. This divides the space into three areas, a central entertainment zone and two peripheral spaces east and west. This is the first draft for the social area. Steps 2–8 provide a series of divisions that allow further detailing of the space. In this Winslow house case spatial merging processes are especially useful for creating the distribution corridor. The last Step concludes the design process by adding some external areas of design. Very much like a Palladian villa, porticos are added to the main design for entrances, verandas and communicating terraces, with the entertainment zones creating transitional spaces between interior and exterior.

The derivation of Malagueira houses using the generic grammar involves an adaptation of the original Malagueira grammar rules [5] that served as a reference type for the conception of the generic grammar. The original grammar is a typical subdivision grammar and, as explained, is the driving force behind the design of this generic grammar.

The example in Fig. 4 illustrates a typical two-bedroom, two-storey, terraced, semi-detached house, type Ab under the classification system devised by Duarte. The proposed derivation uses the subdivision rules previously explained, plus particular shape rules that address Siza's spatial configuration. After the subdivision is performed, the steps that follow diverge from the original grammar and are closer to those tested in the previous derivations. Step 1 is the plot insertion, which involves applying a self-contained rectangular shape. In the case of the Malagueira houses the envelope shape is not parametric, but has a fixed size that reflects the available plot space with the same dimensions and area for each house. This is determined by the plot dimensions of 12×8 m, a perfect 3:2 proportional ratio and a resultant plot area of 96 m^2. Step 2 applies Rule 3 for horizontal subdivision, segregating interior from exterior space. At this stage the yard/exterior space is allocated. Step 3 applies the vertical division, creating a division between the interior functional areas. The house layout now begins with the allocation of (service versus living) zoning. Due to the true nature of this subdivision, recursive vertical and horizontal divisions are performed to carry out the zoning and spacing. These rules are no more than parameterizations of the division rules exemplified. In comparison with the other grammars where only vertical and horizontal divisions are performed, Malagueira houses allow for diagonal divisions as showcased in this house type.

Generic Grammar Validation

Figures 5 and 6 represent the corpus of solutions generated from recursive subdivisions of the outline with up to five subdivisions. The result is an extended corpus of 477 possible solutions for 5 divisions. This makes a total of 569 solutions if the generation process only applies the division rules 5 times. Experience has shown that the generation of Palladian villas, Malagueira and particularly Prairie houses take several recursions to be successful. As previously seen Malagueira house Ab allows for 16 division steps (Fig. 4) and Winslow Prairie house allows 16 steps (Fig. 3).

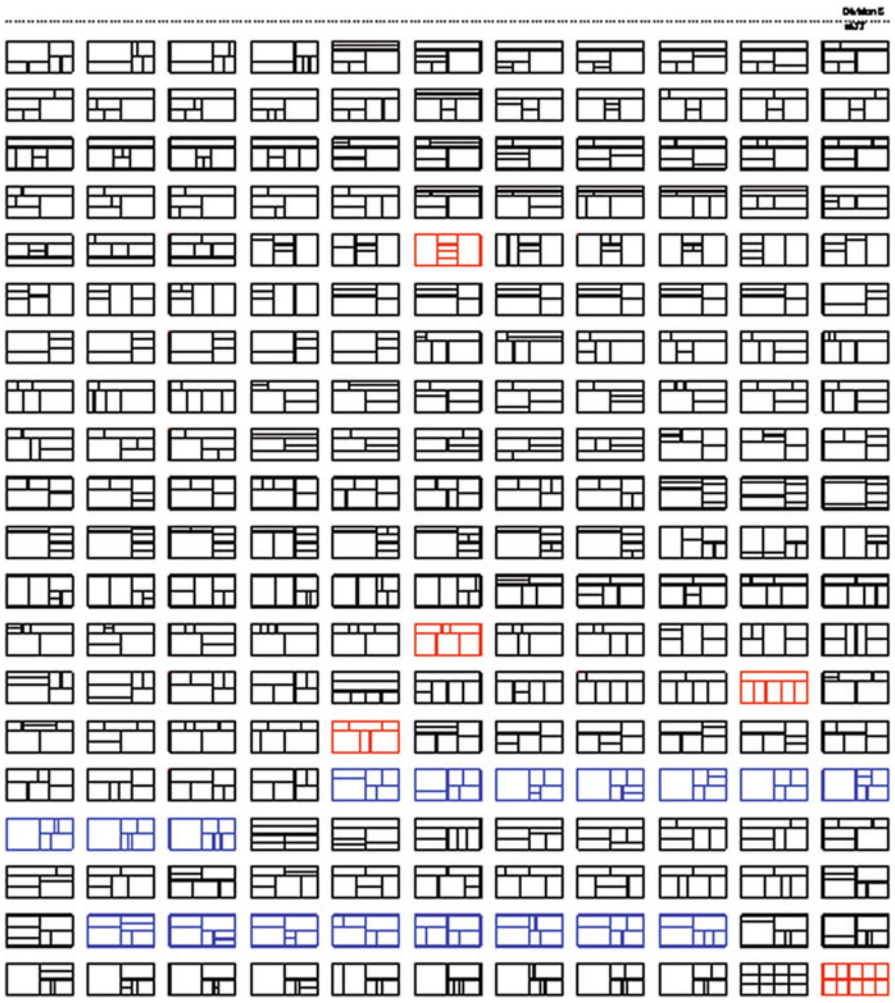


Fig. 6 Generic grammar corpus of solutions for five divisions (continued part 2)

This only shows the potential of a generic grammar of this sort using subdivision as its generative tool. A great deal of designs can be generated as shown in Figs. 5 and 6 and the feasible generation of pre-existing solutions shown in the partial tree diagram with the delineation of possible divisions. This grammar can potentially generate any solution that is self-contained in a rectangular outline. Figure 5 and 6 show the possible patterns allowed by the Palladian villas and the Malagueira. These are incorporated into the shape rules parameterization as shown in the shape rules sequence demonstrated in table 8.1. These patterns allow for underlined grids of 3×2, 3×3, 3×4, 5×5, 5×3, 5×4, 5×5. Despite the abstract grid principle, this is only a rationalization used to maintain the proportion ratio of the divisions because in practice only the desired divisions are performed. Malagueira allows for eight basic layouts that derive from vertical and horizontal divisions. This way the basic layout is determined in three divisions.

There are 477 possible solutions for the corpus of 5 divisions, combining vertical and horizontal. The solutions in red represent potential Palladian villas solutions, green Prairie houses and blue Malagueira. This corpus only focus on configuration, the shapes and proportions vary parametrically. The bottom red example resembles in configuration the Palladian villa Ragona and the immediate above resembles villa Angarano designed and built by Andrea Palladio with underlining 3×3 grids.

There are 477 possible solutions for the corpus of 5 divisions, combining vertical and horizontal. The solutions in red represent potential Palladian villas solutions, green Prairie houses and blue Malagueira. This corpus only focus on configuration, the shapes and proportions vary parametrically. The bottom red example resembles in configuration the Palladian Godi designed and built by Andrea Palladio with underlining 5×2 grid. Additionally are blue examples which fit Malagueira criteria and good starting points for houses Ab and Bb.

This corpus only focus on configuration; the shapes and proportions vary parametrically. The diagram represented replicates the possible sub-divisions that occur using the subdivision grammar by mapping the horizontal and vertical divisions. The solutions in red represent potential Palladian villas solutions, green Prairie houses and blue Malagueira.

The range of potential solutions illustrated for five divisions illustrate, among others, examples of the original corpus of Palladian villas Prairie and Malagueira houses. Among the pre-existing corpus of solutions can be found the Palladian villa Godi designed and built by Andrea Palladio with underlining 5×2 grid, and examples which fit Malagueira criteria for houses Ab and Bb.

Conclusion

Previous shape grammar work has tended to propose a unique grammar that describes a particular corpus of designs or, for that matter, an alternative grammar for a grammar already developed, even though the range of work produced by the grammar also has intrinsically common features. This implies that finding and studying this grammar tells us something about the essence of the corpus of work.

The present work refutes a common assumption of this approach, namely the uniqueness of the design style that one grammar can produce. Given that there is more than one way to reproduce designs, more than one suitable grammar and that one grammar that can produce more than one style, many different representations are potentially viable. Shape grammars can thus potentially be manipulated to generate a larger corpus of new designs. This may allow for efficiency in exploring shapes and analysing results, thus widening the scope of grammars. The advantages and limitations of shape grammars can be considered in two domains: advances for the research field and advances useful for architectural practice. The generation of shape grammars is useful in practice primarily for their potential in exploring potential design options. This could be used as a way to tackle diversity in mass customization as demonstrated in various works. In addition shape grammars have proved successful in industrial design such as in the car industry by Cagan and Osborn. For research shape grammars are important testimonials and tools that concentrate know how and architectural 'best practice'. For art historians these could be useful tools in the identification and classification of non-assigned authorship of buildings.

If shape grammars with specific languages can recreate (or replicate) design solutions within a specific family of solutions, a generic grammar can offer a range of design solutions and several families of results. This could be finely tuned to allow for a design consistency using parameterization as shown in previous sections of this work. A generic grammar such the one exposed allows for an additional level of flexibility to the current shape grammar without losing coherence. This derives from one of four rule types: addition, subtraction, subdivision and concatenation. Fine tuning can be achieved by the restrictions and conditions provided. Applications of this could be easily tested in practice where the designer would only require the adjustment of the parameterization as best suited. Variables such as the plot size, or the area available for each room or rooms could be predetermined. It is important to note that in the generic grammar only the fourth stage has a common branch. From there ramifications confer upon each design family its signature details. Such details can also be customized. The three original case studies and their parameterization are merely indicative and are presented to illustrate and describe the grammar. Others can be added. An obvious limitation to the rules as demonstrated is the geometry allowed. Due to the case studies selected, most of the designs propose an orthogonal setting and rectangular enclosure. This does not have to be a condition for other generic grammars but efficiently described the corpus that was selected. Other variables can be integrated to allow for other geometries by increasing the number of parameters used.

In addition, due to the two-dimensional nature of the majority of the grammars used the generic formulation was also represented two-dimensionally, but this could be easily converted into a three-dimensional formulation through a more extensive survey of the corpus of original solutions. Future work will focus on automatic inference of housing grammars by using the generic grammar as a base.

The original grammarians have extracted their rule set through observation, identification of common design patterns and analysis of the many solutions, but what is not clear is the rule inference process, as this is empirical, creative, and has not been to this point successfully explained. The generic grammar derived from a set of three languages in this paper leads to several observations:

A grammar can generate more than one language. The set of shape rules are not only a combination of the original rules but an optimization of the of the design intent. The careful synthesis of these shape rules can generate not only the original languages but other corpuses of solutions. It is thought that the generic grammar will serve as a foundation from which specific housing grammars can be described not as rule sets, but as parameter ranges. Future work will focus on the effectiveness and implementation of the generic grammar, with a focus on the meaning of the regions of design space between existing corpora. It is expected that the mutation of these design styles or the overlapping of rules will produce new consistent designs with a new hybrid style. Moreover, computerised implementation will represent a positive development, allowing for the exploration of design solutions and even the enumeration of design corpus results. The potential of this generic grammar will be fully tested with a computerised tool, as was the case with previous work developed for housing shape grammars, such as the ABC system and the Haiti gingerbread house grammar [3].

References

1. Benros D, Hanna S, Duarte J (2012) Towards a generic shape grammar for housing: revisiting the grammars for the Palladian, Malagueira, and Prairie houses, DCC Conference, College Station, Houston, Texas, USA
2. Benros D, Duarte JP, Hanna S (2012). An alternative Palladian shape grammar: a subdivision grammar for Palladian villas. Proceedings of the 17th International Conference on Computer Aided Architectural Design Research in Asia/Chennai 25–28, pp. 415–424
3. Benros D et al (2011) Integrated design and duilding system for the provision of customized housing: the case of post-earthquake Haiti. In: Proceedings of the 14th International Conference on Computer Aided Architectural Design Futures. CAAD Futures 2011. Liege, Belgium, pp 247–264
4. Duarte J et al (2006) An urban grammar for the medina of Marrakech. In: Design computing and cognition'06, pp 483–502. Available at: http://dx.doi.org/10.1007/978-1-4020-5131-9_25. Accessed 18 Jun 2010
5. Duarte JP (2001). Customizing mass housing : a discursive grammar for Siza's Malagueira houses. Massachusetts Institute of Technology. Available at: http://theses.mit.edu:80/Dienst/UI/2.0/Describe/0018.mit.theses/2001-328. Accessed 27 May 2010
6. Knight TW (1999) Shape grammars: six types. Environ Plan B Plan Des 26(1):15–31
7. Koning H, Eizenberg J (1981) The language of the prairie: Frank Lloyd Wright's prairie houses. Environ Plan B Plan Des 8(3):295–323
8. Palladio A (1997) The four books on architecture. MIT Press, Cambridge

9. Prats M (2007) Shape exploration in product design: assisting transformation in pictorial representations. The Open University, London, Available at: http://design.open.ac.uk/prats/index.htm [Accessed December 3, 2010]
10. Stiny G, Mitchell WJ (1978) The Palladian grammar. Environ Plan B Plan Des 5(1):5–18
11. Stiny G (2011) What rule(s) should I use? Nexus Network Journal 13(1):15–47

Language of the Rascian School: Analyzing Rascian Church Plans via Parallel Shape Grammar

Djordje Krstic

Abstract A parallel shape grammar has been defined to study Serbian medieval monastic churches of the Rascian School. It combines church plans with the associated justified permeability graphs. The new grammar does what shape grammars traditionally do: explores the richness of the language by enumerating its possible instances—there are 62 different types of Rascian church plans enumerated by the grammar. In addition to this, the grammar also pinpoints possible change in social use of the church space evident in churches built after Serbian Orthodox Church gained independence in 1219.

Introduction

Many years ago (1987) R.B. Boast [1] argued in favor of using two representations: topological and formal (i.e., a representation of the geometric form) in analyzing spatial organization of the archeological built space artifacts. In particular, he proposes justified graphs—of space syntax theory [2]—as a former representation and shape grammars [3] as a latter one. Boast further suggested an integration of the two representations as a means to understanding the social use of the built environment—"through the integration of the two aspects, a deeper insight into the significance of built organization may be gained" [1]. Unfortunately, he falls short of achieving this goal. "Though no complete integration of the relational and formal is achieved, it is hoped that this paper (i.e. [1], comment by D.K.) will serve as catalyst and a challenge for work towards such an integration" [1].

In contrast, the desired integration may be achieved with the aid of a parallel shape grammar [4] capable of simultaneously generating plans and related justified graphs of built space. An instance of such a grammar will be developed here in order to analyze/define the style of Serbian Medieval churches of the Rascian

D. Krstic (✉)
Krstic Design, USA
e-mail: gkrstich@aol.com

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_24

School. We will also use this platform to test Boast's claim that the method—that is, the integration of the two representations—provides "a deeper insight" into possible social use of built space.

Method

Following Boast, two representations of churches are used:

i. A plan as a geometrical representation, or in Boast's terms a representation of the form, and
ii. A justified permeability graph as a characterization of the relation among the spaces in terms of their accessibility—possible passage from one to another. Boast considers this a topological or relational representation.

Desired integration may be achieved via parallel shape grammars—an extension of the shape grammar formalism—that can handle simultaneous generation of multiple representations of an object.

Justified Permeability Graphs

Justified permeability graphs are tools widely used in space syntax theory, to highlight probable social uses of the spaces they represent. Nodes of a permeability graph represent spaces while edges represent passages from one space to another. A permeability graph is justified with respect to one of the nodes so that this node is placed at the root and all other nodes appear on different levels based on how far they are from the root. Nodes directly connected to the root appear on level 1. Nodes directly connected to nodes on level $n-1$ appear on level n, provided that they are not on levels $n-2$ or $n-1$. That is, if going from the root one has to pass through at minimum $n-1$ space in order to get to a certain space then the latter appears on level n. Any node of a permeability graph may be chosen as a root. We will use justified graphs with roots placed at the outside space only. These indicate how well the inside spaces are protected: the further a space is from the root the better protected it appears to be.

Five Rascian churches represented by their plans and associated justified permeability graphs are depicted in Fig. 1.

Note how justified (permeability) graphs have a very strong pictorial aspect. Slender and tall graphs characterize the strong hierarchy of spaces with well-protected/well-controlled spaces close to the top. In contrast, in wide and shallow graphs there is less hierarchy and less protection/control. Built environments characterized by the latter graphs invite more democratic social uses.

While (permeability) graphs are well represented by simple symbolic devices—like connectivity matrices—their justified counterparts require more elaborate ones. The latter graphs may benefit from a pictorial representation i.e. being seen as shapes.

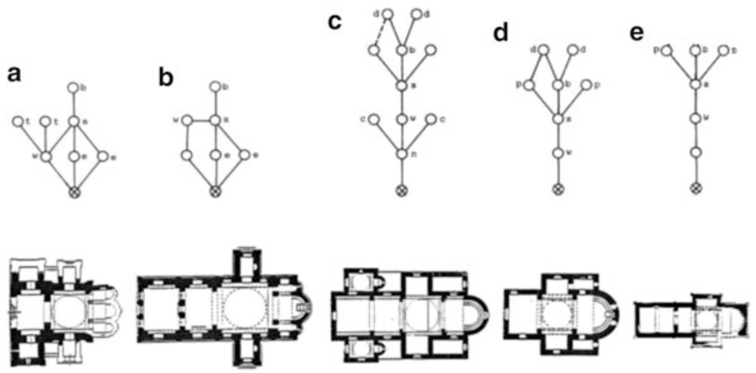


Fig. 1 Plans and associated justified graphs: St George, Ras (**a**), Virgin, Studenica (**b**), Ascension, Zicha (**c**), Pridvorica (**d**), and Hvostan Virgin, Pech (1219) (**e**)

Parallel Shape Grammars

Because justified permeability graphs may be seen as shapes it is possible to generate both plans and graphs via shape grammars. We need two grammars one for plans and another for justified permeability graphs. We also need to match their rules so that we can run them in parallel. The simultaneous generation of the two representations guaranties their integration. The two synchronized grammars may be seen as one parallel grammar.

More formally, parallel shape grammars are production systems similar to shape grammars. As shape grammars they are sets of production rules. The rules are of the same form as the shape grammar rules and are applied in the same way. Each rule has a left-hand and a right-hand side connected with an arrow. Whenever there is a transformation that makes the left-hand side of a rule a part of the object, to which the rule is applied, the transformed left-hand side of the rule is removed from the object and replaced with the right-hand side of the rule transformed in the same way the left-hand side was. The only difference is that the objects of parallel shape grammars are not shapes but *compound shapes*, which are ordered *n*-tuples of shapes. Their production rules are of the form

$$(\boldsymbol{a_1}, \ldots \boldsymbol{a_n}) \rightarrow (\boldsymbol{b_1}, \ldots \boldsymbol{b_n}),$$

where $\boldsymbol{a_1}, \ldots \boldsymbol{a_n}, \boldsymbol{b_1}, \ldots \boldsymbol{b_n}$ are shapes. A rule can be applied to compound shape $(\boldsymbol{c_1}, \ldots \boldsymbol{c_n})$ to produce compound shape $(\boldsymbol{c'_1}, \ldots \boldsymbol{c'_n})$ if there exists an *n*-tuple $(\boldsymbol{t_1}, \ldots \boldsymbol{t_n})$ of geometric transformations, that act on shapes, such that

$$(\boldsymbol{t_1}, \ldots \boldsymbol{t_n})(\boldsymbol{a_1}, \ldots \boldsymbol{a_n}) \leq (\boldsymbol{c_1}, \ldots \boldsymbol{c_n}) \tag{1}$$

holds and

$$\begin{aligned}&(\boldsymbol{c'_1}, \ldots \boldsymbol{c'_n})\\ &= [(\boldsymbol{c_1}, \ldots \boldsymbol{c_n}) - (\boldsymbol{t_1}, \ldots \boldsymbol{t_n})(\boldsymbol{a_1}, \ldots \boldsymbol{a_n})] + (\boldsymbol{t_1}, \ldots \boldsymbol{t_n})(\boldsymbol{b_1}, \ldots \boldsymbol{b_n}).\end{aligned} \tag{2}$$

Note that operations and relations on compound shapes are defined in terms of their components so that

$$\begin{aligned}&(\boldsymbol{t_1},\ \ldots \boldsymbol{t_n})(\boldsymbol{x_1},\ \ldots \boldsymbol{x_n}) = (\boldsymbol{t_1}(\boldsymbol{x_1}),\ \ldots \boldsymbol{t_n}(\boldsymbol{x_n})),\\ &(\boldsymbol{x_1},\ \ldots \boldsymbol{x_n}) \pm (\boldsymbol{y_1},\ \ldots \boldsymbol{y_n}) = (\boldsymbol{x_1} \pm \boldsymbol{y_1},\ \ldots \boldsymbol{x_n} \pm \boldsymbol{y_n}),\ \text{and}\\ &(\boldsymbol{x_1},\ \ldots \boldsymbol{x_n}) \leq (\boldsymbol{y_1},\ \ldots \boldsymbol{y_n})\ \text{is equivalent to}\ \boldsymbol{x_1} \leq \boldsymbol{y_1},\ \ldots \boldsymbol{x_n} \leq \boldsymbol{y_n},\end{aligned}$$

where $\boldsymbol{x_1}, \ldots \boldsymbol{x_n}, \boldsymbol{y_1}, \ldots \boldsymbol{y_n}$ are shapes, $\boldsymbol{t_1}, \ldots \boldsymbol{t_n}$ are geometric transformations that act on shapes, + and – are operations of sum and difference for shapes, while $\leq$ is the part relation. Each component shape, $\boldsymbol{x_i}$ and $\boldsymbol{y_i}$ together with the component transformation $\boldsymbol{t_i}$ belongs to an algebra of shapes $\boldsymbol{A_i}$ which is partially ordered by $\leq$ and closed under operations + and – as well as under transformations. The latter algebra is itself a component of the direct product algebra $\boldsymbol{A} = \boldsymbol{A_1} \times \ldots \times \boldsymbol{A_n}$, which provides the framework for the above computations.

Parallel computations may also be carried on in algebras smaller than $\boldsymbol{A}$. Such is a subset $\boldsymbol{A'}$ of $\boldsymbol{A}$ in which the transformations are restricted to n-tuples with equal components, or $(\boldsymbol{t_1}, \ldots \boldsymbol{t_n})$, where $\boldsymbol{t_1} = \boldsymbol{t_2} = \ldots \boldsymbol{t_n}$. The restriction is useful to prevent the shapes consisting of components defined in different algebras to be torn apart when moved, rotated or otherwise transformed. Algebra $\boldsymbol{A'}$ is distinguished as the sum of algebras $\boldsymbol{A_i}$, or $\boldsymbol{A_1} + \ldots + \boldsymbol{A_n}$ [5]. Parallel computations may be carried on in any combination of sum and direct product algebras.

Each compound shape in the language defined by a parallel shape grammar is generated by recursive rule applications. A generation starts by first applying the initial rule, which is a distinguished element of every grammar. An empty left-hand side characterizes the latter rule so that it may be applied to any compound shape under any transformation to create an initial compound shape. This way a complete freedom is allowed in placing the initial compound shape, as well as choosing its scale and orientation. Consecutive rule applications are done to the initial compound shape and to the compound shapes generated by the previous rule applications. The generation stops when no rule—other then the initial rule—can be applied. That is, compound shape $(\boldsymbol{c_1}, \ldots \boldsymbol{c_n})$ becomes an instance of the language defined by the grammar whenever condition (1) cannot be satisfied for any rule and transformation combination.

Rascian Churches

Within the borders of modern states of Serbia and Macedonia, Byzantine monuments or monuments marked by the Byzantine inspiration, extend on a North-South axes, from Belgrade to the Macedonia-Greek border. On that territory Medieval Serbia had an independent existence spanning between 2 and 300 years. It culminated in 1346 when Serbian Empire is established and ended some hundred years later when all of the Serbian territories were concord by Ottoman Empire.

Even before Stefan Nemanja united Serbian tribes in the independent state Christianity was the main factor of their cohesion. For that reason church was the main authority, and monasteries played an important role in the life of Medieval Serbia. The most important Serbian churches were parts of monastic complexes.

The basic classification of Serbian churches was established by Gabriel Millet [6] at the beginning of the last century and is still valid today. Millet distinguishes three schools: Rascian, Byzantine, and Morava.

Our attention will be focused on the first one, the Rascian School, where Byzantine elements blend with Western producing an original style distinguished from the Byzantine architecture.

Rascian style is analyzed here in terms of configurations of spaces as they appear in the church plans. Consequently, some of the buildings with non Rascian plans, but considered Rascian on the basis of some other attributes, are omitted. The spatial logic of Rascian language is extracted from a body of 20 churches 9 of which are represented by their plans and longitudinal sections in Fig. 2. The drawings in Fig. 2 are done after Deroko [7], but modified to represent churches as they were originally built—without later additions. Because the ceiling and dome structures

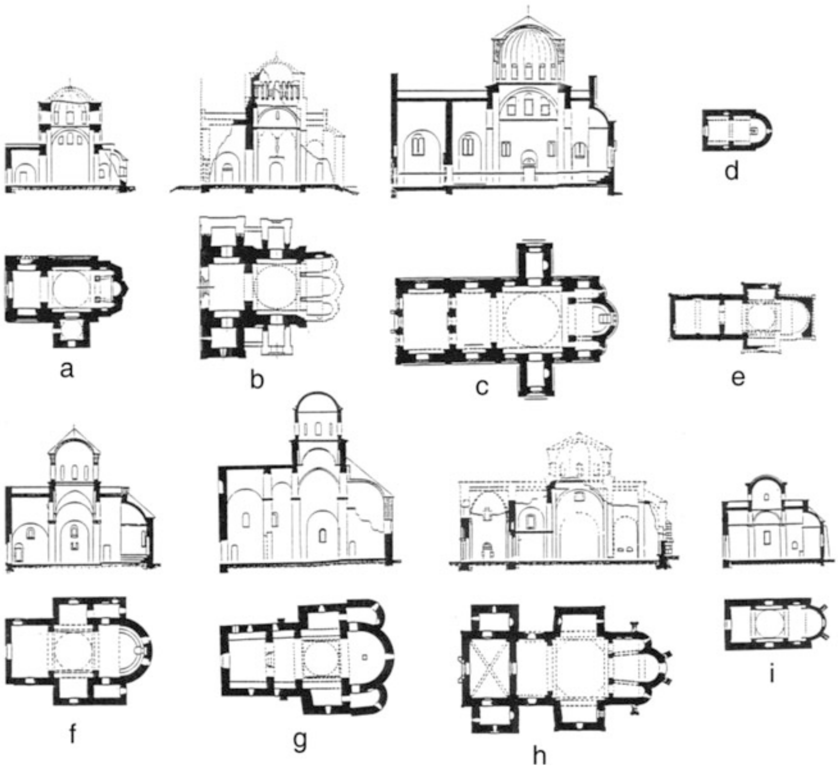


Fig. 2 Rascian churches: St. Nicholas, Kurshumlia (1168–1174) (**a**), St. George, Ras (1171) (**b**), Virgin, Studenica (1183–1220) (**c**), St. Peter and Paul, Zicha (**d**), Hvostan Virgin, Pech (1219) (**e**), Pridvorica (before 1220) (**f**), St. George, Ivangrad (before 1220) (**g**), Annunciation, Gradac (1270) (**h**), White Church, Karan (1332–1337) (**i**).

are essential attributes of Rascian church design their projections—represented by dashed lines—are also included in the plan.

Basic Rascian plan is a rectangular nave with the longer side on the East-West axes. The nave is partitioned into three spaces: western bay, solea in the middle, and bema on the East end. The basic plan may be elaborated by an addition of a narthex to the West and lateral annexes to the North and South sides of the nave.

There are five types of lateral annexes:

i. Chapels, which may be added ether to the solea, or narthex, or western bay—if there is no narthex. They are added in pairs, in symmetrical fashion, or as a single addition destroying the church plan symmetry. A church may have at most two chapels.
ii. Lateral porches, which are symmetrical additions to the solea. A plan may have ether two or no porches.
iii. Closed transepts, which are added to the solea in the same fashion as the lateral porches.
iv. Proskoumizos and diakonikos, always appear as a pair added, respectively, to the North and South sides of the bema extending the closed transepts. The existence of the latter is necessary for proskumizos and djakonikos to be added.
v. Bell towers, which are symmetrical additions to the narthex, or western bay of nave if there is no narthex. A plan may have either two or no bell towers.

As a consequence of (i) and (v) churches with narthexes have no additions to their western bays. Furthermore, solea annexes are prerequisite for the addition of any other lateral annex.

Rascian churches have masonry ceiling constructions. It typically consists of a barrel vault resting on the North and South walls or arches attached to the walls. In some churches a cross-vault occurs as the ceiling construction above the narthex or western bay.

Bigger churches with lateral annexes may have a dome above the solea. The drum of the dome rests on four connected arches set in the walls of a small tower carried by the main arches of the church. Throughout the development of the Rascian style the desire to raise the dome as high as possible on a successive series of arches that overhung the solea—in a step-wise fashion—is noticeable.

The Grammar

Based on the spatial configuration of Rascian churches—described above—a parallel shape grammar is defined to concurrently generate three representations: plan $\boldsymbol{p}_i$, projected ceiling $\boldsymbol{c}_i$, and a justified permeability graph $\boldsymbol{g}_i$. The first two representations are defined in the same space and manipulated with the same transformations. The third representation—the graph—is defined in a different space—which is pictorially distinguished by a gray background. Consequently, compound shapes that the grammar generates are ordered pairs $((\boldsymbol{p}_i, \boldsymbol{c}_i), \boldsymbol{g}_i)$

belonging to a direct product algebra. Their first component ($\boldsymbol{p_i}$, $\boldsymbol{c_i}$)—which is an ordered pair of shapes—belongs to a sum of two algebras of shapes, while their second component $\boldsymbol{g_i}$ is a shape, belonging to a different algebra of shapes. Thus, the framework for our computations is a compound algebra, which is a direct product of a sum of two algebras of shapes and an algebra of shapes.

A rule $((\boldsymbol{p_1}, \boldsymbol{c_1}), \boldsymbol{g_1}) \rightarrow ((\boldsymbol{p_2}, \boldsymbol{c_2}), \boldsymbol{g_2})$ can be applied to compound shape $((\boldsymbol{p}, \boldsymbol{c}), \boldsymbol{g}))$ whenever there is an ordered pair of transformations ($\boldsymbol{t_1}$, $\boldsymbol{t_2}$) such that ($\boldsymbol{t_1}$, $\boldsymbol{t_2}$) (($\boldsymbol{p_1}$, $\boldsymbol{c_1}$), $\boldsymbol{g_1}$) $\leq$ (($\boldsymbol{p}$, $\boldsymbol{c}$), $\boldsymbol{g}$), in accordance with (1). The latter is equivalent to three inequalities $\boldsymbol{t_1}(\boldsymbol{p_1}) \leq \boldsymbol{p}$, $\boldsymbol{t_1}(\boldsymbol{c_1}) \leq \boldsymbol{c}$, and $\boldsymbol{t_2}(\boldsymbol{g_1}) \leq \boldsymbol{g}$, belonging to three separate algebras of shapes. Similarly, the rule application—done in accordance with (2) and resulting in compound shape (($\boldsymbol{p}'$, $\boldsymbol{c}'$), $\boldsymbol{g}'$)—is equivalent to three computations $\boldsymbol{p}'$ $= [\boldsymbol{p} - \boldsymbol{t}_1(\boldsymbol{p}_1)] + \boldsymbol{t}_1(\boldsymbol{p}_2), \boldsymbol{c}' = [\boldsymbol{c} - \boldsymbol{t}_1(\boldsymbol{c}_1)] + \boldsymbol{t}_1(\boldsymbol{c}_2),$ and $\boldsymbol{g}' = [\boldsymbol{g} - \boldsymbol{t}_2(\boldsymbol{g}_1)] + \boldsymbol{t}_2(\boldsymbol{g}_2)$, carried out in their respective algebras of shapes.

It is interesting to note that shape grammar generation of a justified permeability graph differs from its construction via the standard space syntax procedure. The latter procedure requires some global knowledge about the building—namely a completed plan. Based on the plan a permeability graph is constructed first and then justified. In contrast, shape grammars work with the local knowledge only. They do not have a completed plan when generating the related justified permeability graph because both the plan and the graph are generated in parallel. The latter is generated directly omitting the permeability graph construction.

The Rascian grammar has 39 rules depicted in Figs. 3, 4, 5, 6, and 7. Each rule has two components; one that acts on plans and projected ceilings and the other that acts on graphs. The shapes on white background—appearing on both sides of the arrow—represent the former component while shapes on gray background represent the latter.

For example, rule 2 in Fig. 3, which adds bema, has a component that incorporates the plan of the bema into the plan of the church, and component that adds a node and an edge to the graph.

There are rules where one component does not parallel the action the other. This component then takes the identity form $0 \rightarrow 0$, where 0 denotes the empty shape.

For example, rules 6 and 7 in Fig. 4 act on a graph to reduce its depth while leaving the plan unchanged similarly, rule 21 in Fig. 6 elongates a chapel in a plan, but leaves the associated graph alone.

Without rules with identities ($0 \rightarrow 0$) we cannot generate plans with similar geometries but different topological descriptions or alter plans without changing their topologies. The very existence of such rules in a grammar that handles multiple representations of objects speaks of differences between the representations and justifies their choice. In contrast, each compound shape in a language may be a collection of different designs—not a collection of different representations of one. The grammar generating it may lack the rules with identities, but still be a sound one. Each instance of such a language is a collection of isomorphic designs, for example, a collection of different chair designs simultaneously generated by a parallel grammar.

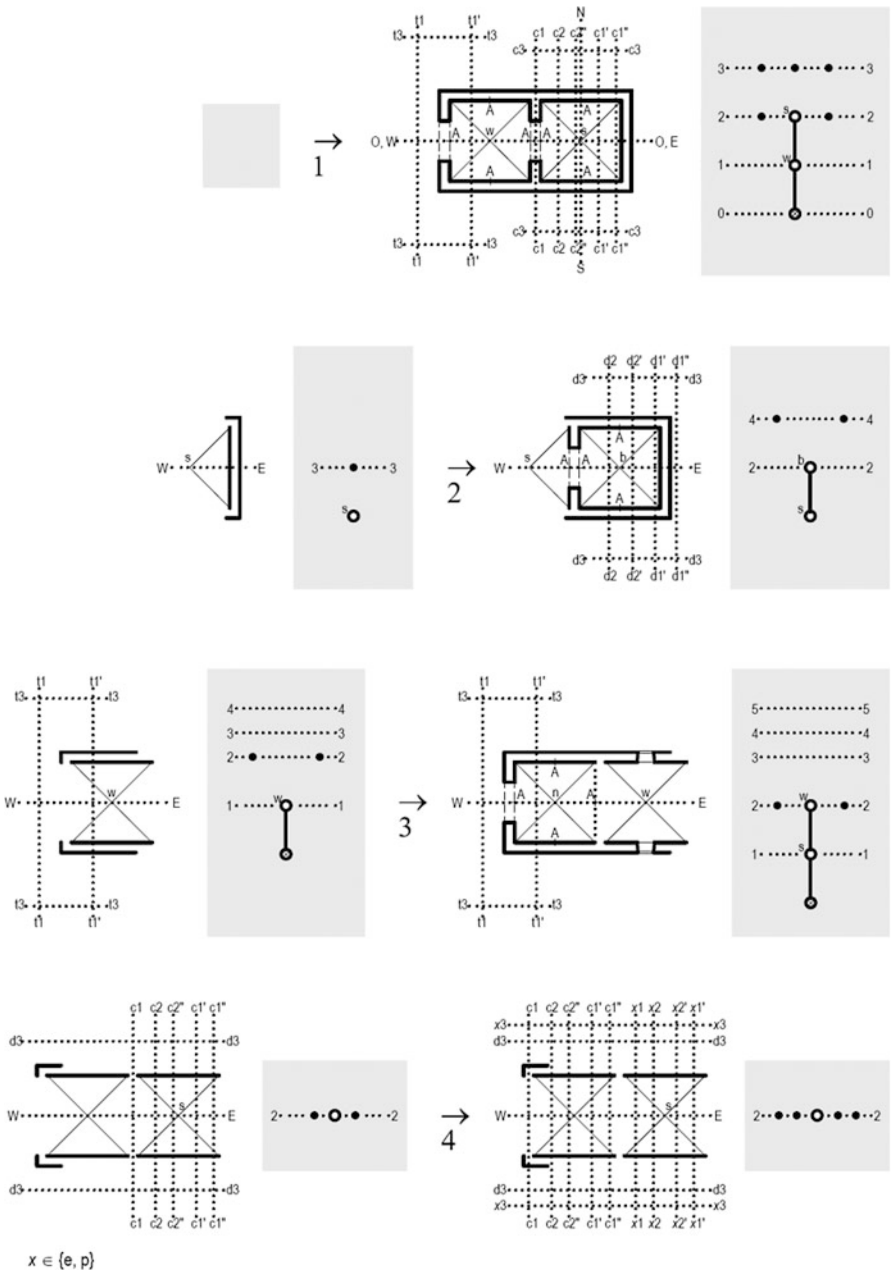


Fig. 3 The first four rules of the Rascian grammar

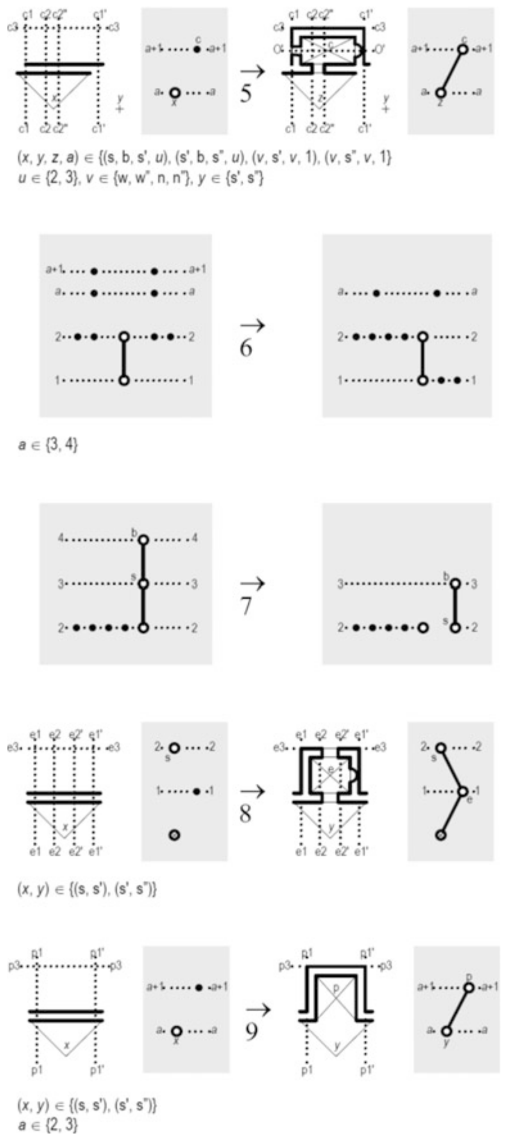

5
(x, y, z, a) ∈ {(s, b, s', u), (s', b, s", u), (v, s', v, 1), (v, s", v, 1}
u ∈ {2, 3}, v ∈ {w, w", n, n"}, y ∈ {s', s"}
6
a ∈ {3, 4}
7
8
(x, y) ∈ {(s, s'), (s', s")}
9
(x, y) ∈ {(s, s'), (s', s")}
a ∈ {2, 3}

Fig. 4 Rascian rules 5 through 9

Out of the 39 Rascian rules 27 are with identities. Such distribution of rules alone renders Boast's selection of representations a promising one.

Rule 1 in Fig. 3 is the initial rule of the grammar, which has the characteristic form ((0, 0), 0) → (($\boldsymbol{p_0}$, $\boldsymbol{c_0}$), $\boldsymbol{g_0}$). It's right-hand side represents a plan and the related justified permeability graph of a basic two space church. It also contains dotted grid lines labeled c_i and t_i, which guide future lateral additions. The lines align with the perimeter lines of the additions thus controlling their overall dimensions. This allows for a pair of symmetrical additions to be generated separately using the

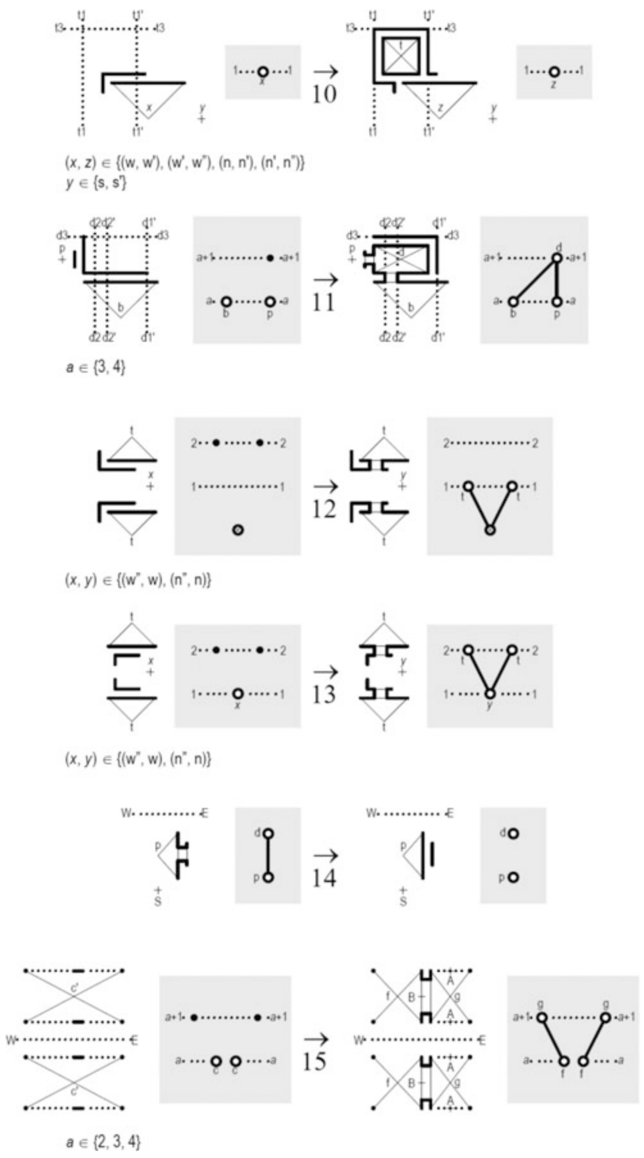

10
(x, z) ∈ {(w, w'), (w', w"), (n, n'), (n', n")}
y ∈ {s, s'}
11
a ∈ {3, 4}
12
(x, y) ∈ {(w", w), (n", n)}
13
(x, y) ∈ {(w", w), (n", n)}
14
15
a ∈ {2, 3, 4}

Fig. 5 Rascian rules 10 through 15

same rule twice. However, only the symmetry of perimeter lines of the additions is guaranteed. Similar asymmetries are evident in Rascian church plans due to the primitive surveying techniques used by the medieval builders.

The Rascian grammar in action—the derivation of the plan and the associated justified permeability graph of the church of the Virgin in Studenica—is shown in Fig. 8.

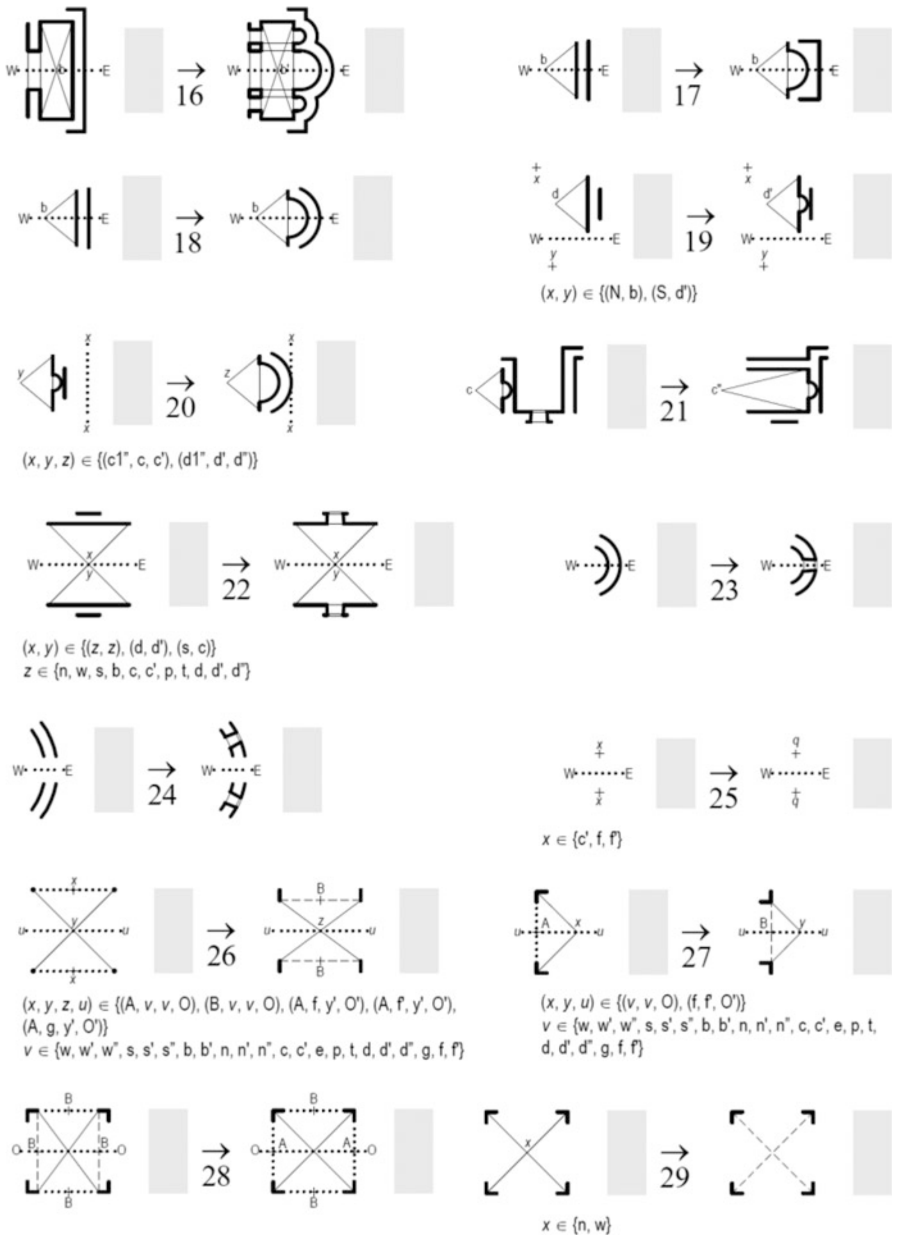

16
17
18
19
(x, y) ∈ {(N, b), (S, d')}
20
(x, y, z) ∈ {(c1", c, c'), (d1", d', d")}
21
22
(x, y) ∈ {(z, z), (d, d'), (s, c)}
z ∈ {n, w, s, b, c, c', p, t, d, d', d"}
23
24
25
x ∈ {c', f, f'}
26
(x, y, z, u) ∈ {(A, v, v, O), (B, v, v, O), (A, f, y', O'), (A, f', y', O'), (A, g, y', O')}
v ∈ {w, w', w", s, s', s", b, b', n, n', n", c, c', e, p, t, d, d', d", g, f, f'}
27
(x, y, u) ∈ {(v, v, O), (f, f', O')}
v ∈ {w, w', w", s, s', s", b, b', n, n', n", c, c', e, p, t, d, d', d", g, f, f'}
28
29
x ∈ {n, w}

Fig. 6 Rascian rules 16 through 29 with identities $0 \rightarrow 0$ in the graph component

The derivation takes 57 rule applications, which are here depicted in 12 steps. Each step consists of two compound shapes marked by consecutive roman numbers and separated by an arrow (⇒, ⇓, or ⇐). Note that the empty compound shape—the first shape in the derivation—which is depicted by the blank space is appropriately

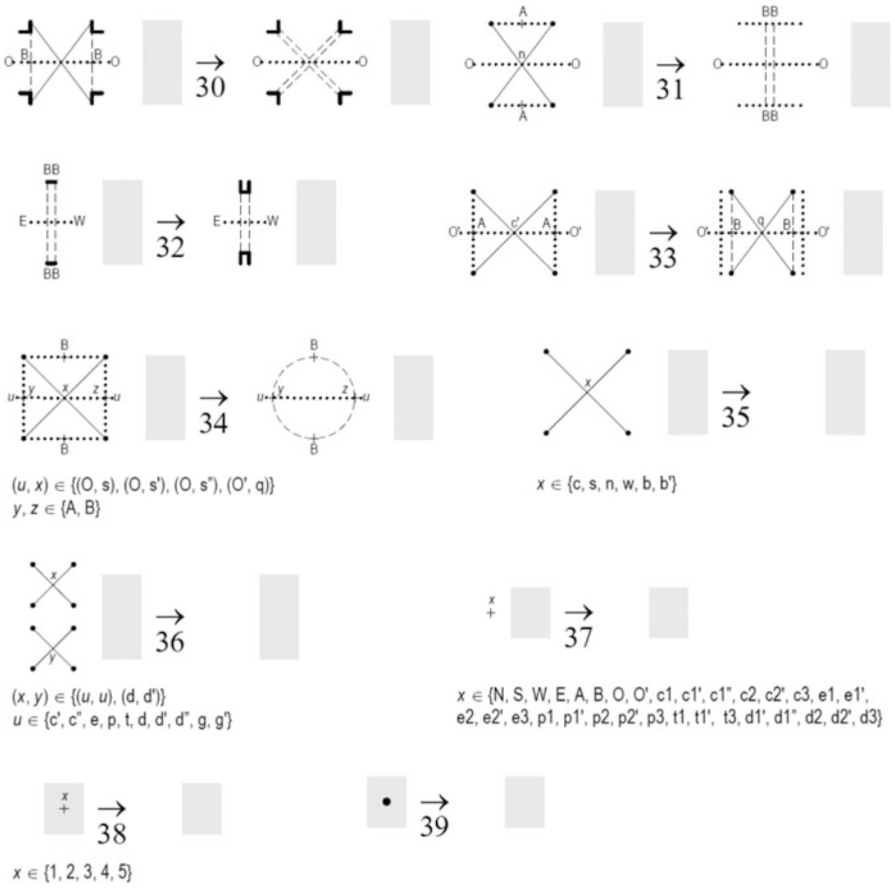

A
BB
B
O
n
c'
O'
q
E
W
u
x
y
z
30
31
32
33
34
35
36
37
38
39
(u, x) ∈ {(O, s), (O, s'), (O, s''), (O', q)}
y, z ∈ {A, B}
x ∈ {c, s, n, w, b, b'}
(x, y) ∈ {(u, u), (d, d')}
u ∈ {c', c'', e, p, t, d, d', d'', g, g'}
x ∈ {N, S, W, E, A, B, O, O', c1, c1', c1'', c2, c2', c3, e1, e1', e2, e2', e3, p1, p1', p2, p2', p3, t1, t1', t3, d1', d1'', d2, d2', d3}
x ∈ {1, 2, 3, 4, 5}

Fig. 7 Rascian rules 30 through 39 with identities $0 \rightarrow 0$ in the graph or plan components

marked by an empty roman number—again a blank space. The numbers appearing next to the arrow denote the rules applied to the compound shape appearing before the arrow to produce the compound shape the arrow points to. The steps are denoted by the numbers of the resulting shapes—the ones pointed to by the arrows.

Discussion

The grammar generates all of the existing Rascian churches as well as a number of new ones in the same style. This capability of grammars to extend the sample beyond the starting one has obvious advantages for the analysis. It is particularly useful to enumerate all of the instances of the style in the language defined by the grammar.

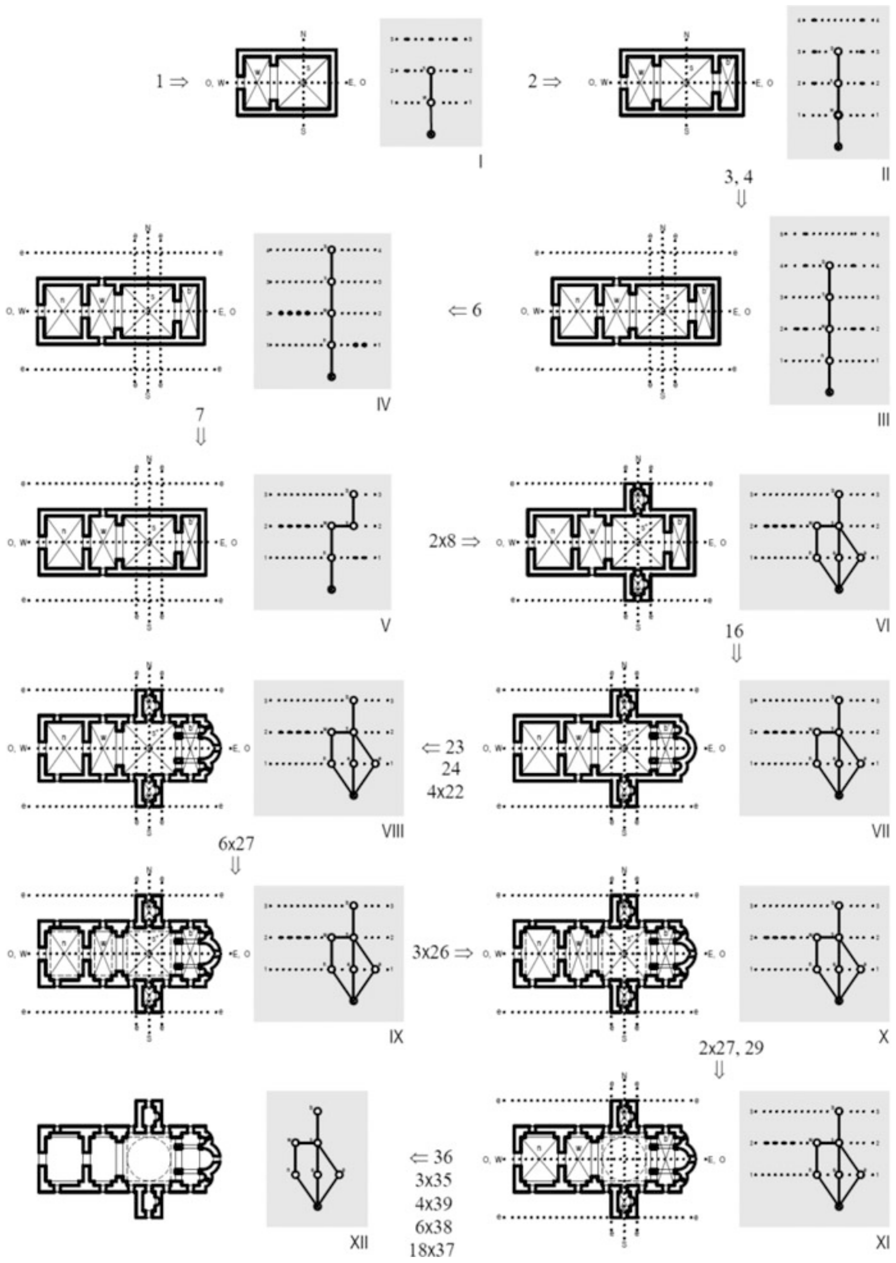


Fig. 8 Derivation of the plan and justified permeability graph of the church of Virgin in Studenca; 57 rule applications in 12 steps

The Rascian language itself is infinite if we allow certain parameters to vary continuously. It is than customary to enumerate classes of designs rather than the designs themselves. The first 11 rules of the grammar generate the basic spatial configuration of a church with the rest of the rules mostly articulating it by filling in the details.

The former rules are used to enumerate 62 classes of churches, 13 of which include the existing churches. The language of Rascian churches appears fairly rich considering its vernacular nature. This becomes even more evident when compared with the number of Palladian villas in the language developed by Stiny and Mitchell [8]. The number of classes of villa plans on 3×3 and 5×3 grids—the sizes that match the number of spaces of Rascian churches—is 230 [9]. The difference is not that significant given that Palladio was outstanding, well-educated, self-conscious architect and there is more freedom in designing villas than in designing churches.

The main point of this paper, the usefulness of integration of multiple representations, has to be justified on the case of the Rascian language. In other words, what insights if any, can one obtain by generating simultaneously a plan and a justified permeability graph, insights not available if say a plan was generated only? R. B. Boast argues that such integration will provide us with better understanding of social use of the built space. In order to test this, one has to look for two plans with similar plans, but different justified graphs. This should be an indication of different social use of otherwise similar space. It will further be useful to find some justification for the difference and also to show that the grammar generates the designs differently reflecting their different social use.

The plans of the churches in Fig. 1b, e feature the same configuration of spaces. Both have a narthex, western bay, solea, and bema as well as the two lateral additions to the solea. In contrast, their justified graphs are different: the second graph is more hierarchical than the first one. The first is shallow allowing for an easy access to the bema via three different paths. The second graph is deep with the bema placed one level higher than in the first one and with just one path leading to it. The desire to control access to the bema—the most sacred space of the church—is evident in the second plan. This may indicate that the hierarchy among the monks using the second church was much stronger then the hierarchy among the monks of the first one. Indeed, the first church—church of the Virgin in Studenica—was built between 1183 and 1196 while the Serbian Church was under Byzantine jurisdiction, where the second one—church of Hvostan Virgin in Pech—emerges in the year 1219 when Serbian Church gained independence. The newly independent church was reorganized and its new hierarchy was clearly reflected in the hierarchy of spaces of Hvostan Virgin.

If a grammar was defined to generate Rascian church plans only the derivations of the basic spatial configurations of the two churches—in Studenica and Pech—would take the same number of rule applications each and would appear isomorphic. The derivations would also be isomorphic to the derivation of the church in Pech by the parallel grammar (Fig. 9).

In contrast, the parallel grammar takes two more rule applications—steps IV and V of Fig. 8—to derive the basic spatial configuration of the church in Studenica

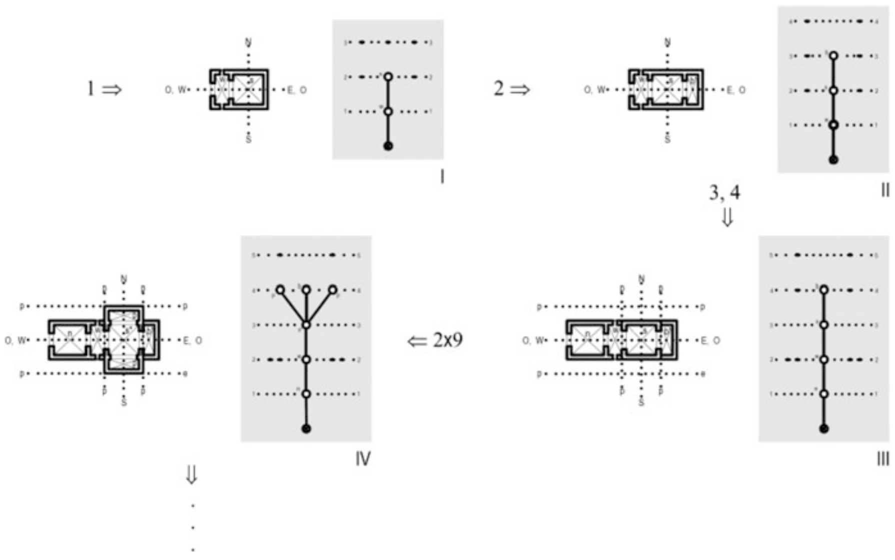


Fig. 9 Derivation of the basic spatial configuration of the church of Hvostan Virgin in Pech; six rule applications in four steps

than it takes for the church in Pech. Additional rule applications are needed to reduce the depth of the justified permeability graph. The difference in derivations of the two churches also reflects the difference in the social use of their spaces.

The composition of spaces introduced in the church of Hvostan Virgin in Pech remains unchanged in the later Rascian churches. They all have deep justified permeability graphs with well-controlled bema access. This remains unchanged even in churches where stylistic flashbacks are evident.

For example, church of Annunciation in Gradac (Fig. 2 h) completed 1270, in which stylistic clues from the past—including long abandoned triple apses of the first three Rascian churches—are contrasted with a deep graph and well-protected bema reflecting the new social use of the space.

Background

The Rascian parallel grammar has originally been presented on April 5, 1990 as a research seminar in Design and Computation series organized by Charles Eastman at UCLA, and has never been published.

Term, parallel shape grammar was first used by Stiny [10] to denote grammars that allow for non-sequential rule applications. This differs from parallel shape grammars defined in the framework of direct product algebras of shapes [4], which is the current meaning of the term. This concept has later been elaborated by Stiny [11], Knight [12], and Krstic [5, 13].

More recently different architectural styles have been analyzed and (or) constructed via parallel shape grammars. Li [14] analyzes Chinese medieval architecture by creating a grammar based on medieval building code, while Duarte [15] generates houses in the style of the contemporary Portuguese architect Alvaro Siza. Heitor et al. [16] uses parallel shape grammars to combine geometry and space syntax.

Books by Stiny [17] and Hillier [18] provide, respectively, recent accounts of shape grammar and space syntax theories.

References

1. Boast RB (1987) Rites of passage: topological and formal representation. Environ Plan B Plan Des 14:451–466
2. Hillier B, Hanson J (1984) The social logic of space. Cambridge University Press, New York
3. Stiny G (1980) Introduction to shape and shape grammars. Environ Plan B Plan Des 7:343–351
4. Stiny G (1991) The algebras of design. Res Eng Des 2:171–181
5. Krstic D (2014) Algebras of shapes revisited. In Gero JS (ed) Design computing and cognition DCC’12. Springer, Dordrecht
6. Millet G (1919) L’Ancient art serbe. Les églises. E de Boccard, Paris
7. Deroko A (1985) Monumentalna i dekorativna arhitektura u srednjevekovnoj Srbiji. Naucna knjiga, Beograd
8. Stiny G, Mitchell W (1978) The Palladian grammar. Environ Plan B Plan Des 5:5–18
9. Stiny G, Mitchell W (1978) Counting palladian plans. Environ Plan B Plan Des 5:189–198
10. Stiny G (1975) Pictorial and formal aspects of shape and shape grammars. Birkhäuser, Basel
11. Stiny G (1992) Weights. Environ Plan B Plan Des 19:413–430
12. Knight T (2003) Computing with emergence. Environ Plan B Plan Des 30:125–155
13. Krstic D (1999) Constructing algebras of design. Environ Plan B Plan Des 26:45–57
14. Li A (2002) Algorithmic architecture in twelfth-century China: the Yingzao Fashi. In: Williams K, Rodrigues JF (eds) Nexus IV: architecture and mathematics. Kim Williams Books, Florence, pp 141–150
15. Duarte JP (2005) Towards the mass customization of housing: the grammar of Siza’s houses at Malagueira. Environ Plan B Plan Des 32:347–380
16. Heitor DV, Duarte JP, Pinto RM (2005) Combining grammars and space syntax: formulating, generating and evaluating designs. IJ Archit Comput 2:492–515
17. Stiny G (2006) Shape: talking about seeing doing. MIT Press, Cambridge
18. Hillier B (1996) Space is the machine: a configurational theory of architecture. Cambridge University Press, New York

Generic Shape Grammars for Mass Customization of Ceramic Tableware

Eduardo Castro e Costa and José Pinto Duarte

Abstract Research is currently being developed on mass customization of ceramic tableware. The objective of this research is to allow end users to personalize their tableware sets, namely to determine the shape of its elements. The implementation of mass customization implies the articulation among three systems: design, production and computation. In this paper we will focus on the design system, which aims at two main goals: to help designers develop tableware collections, and to allow end users to customize these collections. The design system being developed controls the design of the elements of the tableware set in terms of their shape and decoration through the use of shape grammars and parametric modelling. The use of generic shape grammars is suggested as an effective way of handling a wide variety of tableware collections.

Context

Research is currently being developed on mass customization of ceramic tableware. The objective of this research is to allow end users to personalize their tableware sets, namely determining the shape of its elements.

Mass customization was anticipated as an evolution of mass production by Alvin Toffler [1, 2]. As a production paradigm, it combines elements of both craft and mass production. As in craft production, it features a high degree of flexibility in its processes; it builds to order rather than to plan and it results in high levels of variety and personalization. As in mass production, mass customization generally produces in large quantities, has low unit costs, and may rely on automated production [3]. In times when consumers become ever more demanding, and differentiation becomes ever more important, a correct implementation of the mass customization paradigm can boost both customer satisfaction and profit [4].

According to Piller [5], two factors determine the success of mass customization from the firm's point of view: (a) high production flexibility, and (b) elicitation of

E.C.e. Costa (✉) • J.P. Duarte
University of Lisbon, Lisbon, Portugal
e-mail: castroecosta@fa.ulisboa.pt

customer preferences. While the first factor has gained much from the current thriving development of digital fabrication, eliciting the preferences of customers automatically is still not straightforward. An adequate elicitation process implies the implementation of a co-design approach closely related to the customer. This justifies striving for a robust design system for the mass customization of ceramic tableware.

According to Duarte, who applied the mass customization paradigm to housing, the implementation of mass customization implies the articulation among three systems: design, production and computation [6]. In the mass customization system being developed, the design system controls the design of the elements of the tableware set in terms of their shape and decoration through the use of shape grammars and parametric modelling. The production system enables the materialization of the generated models through digital fabrication technology [7]. And finally, a computational system that coordinates the other two systems, resulting from an implementation of the design system.

In this paper we will focus on the design system, which aims at two main goals: to help designers develop tableware collections, and to allow end users to customize these collections.

Methodology

The use of shape grammars [8, 9], is suggested as the main methodology for the development of such design system. Shape grammars are rule-based systems that can generate a wide variety of designs while maintaining stylistic consistency. For this reason, they are used to encode the rules for designing the different elements of the tableware set.

For the development of these shape grammars, several collections have been analyzed and observed, and the rules that govern their design styles have been inferred.

By aggregating the rules of several styles, thus articulating the specific shape grammars, we were able to fine-tune them, as well as to develop new, more generic rules, towards a more generic shape grammar.

In this generic shape grammar lays the foundation of the design system for the mass customization of ceramic tableware. By selecting rules from the generic shape grammar and by defining parametric variation spaces, the designer creates a new, more specific shape grammar [10] that generate a family of collections, corresponding to a new style. By manipulating and defining parameters of the new collection family, the end user generates a new, customized collection, according to Fig. 1.

The application of generic shape grammars has been introduced in other areas of design, namely in architectural design [11, 12] and urban design [10, 13, 14]. Although shape grammars have been used in several fields of industrial and product design, such as automobiles [15], motorcycles [16], and chairs [17], the

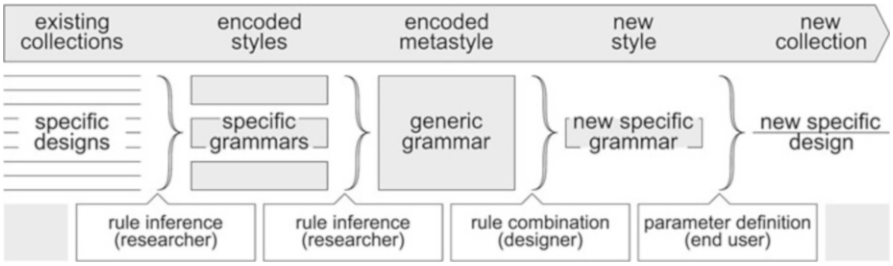


Fig. 1 Generic grammars for designing new collections

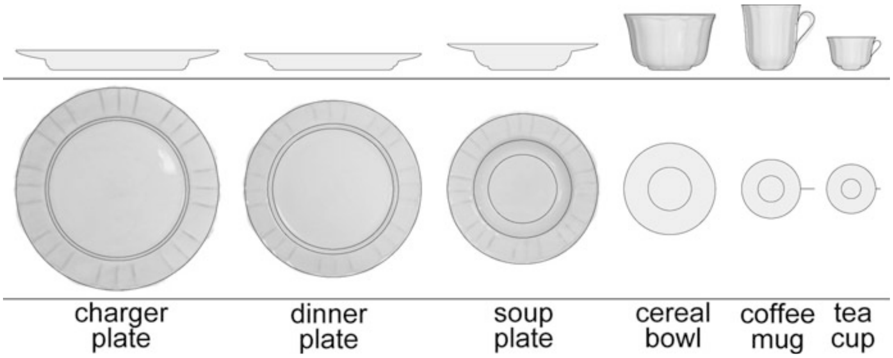


Fig. 2 Example of a tableware collection

application of generic shape grammars to three-dimensional curved shapes in the design of original products, which is described in this paper, is novel.

Single Collection Shape Grammar

The first step towards a design system corresponded to the analysis of one single collection. In this exercise, a shape grammar was developed to encode the design rules inferred from that selected collection [18].

A collection can be defined by (a) the types of tableware elements it contains (for example, dinner plates, cereal bowls, etc.), and (b) its style, which should be consistent across its constituent types, as seen in Fig. 2. Therefore each collection features its own set of rules, which constitute its specific shape grammar, which in turn should be able to generate every type contained by the collection.

The selected collection for this first experiment is composed of six different types, from charger plate to coffee cup, and it features relief-based decoration, due to the research interest on three-dimensional (3D) shape.

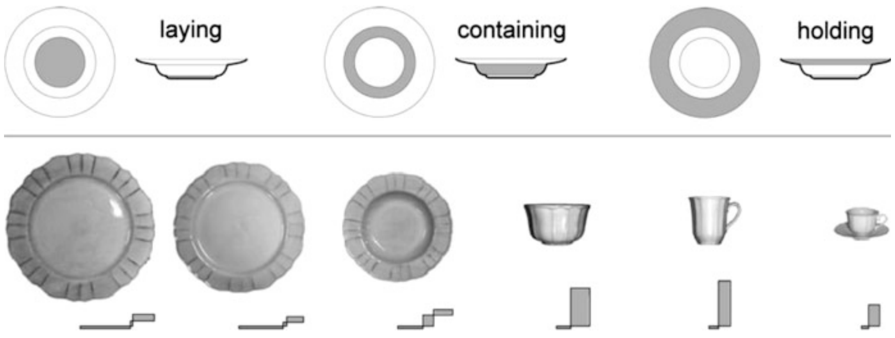


Fig. 3 Three functional parts exemplified in a soup plate (*top*) and functional configurations for the six elements of the analyzed collection (*bottom*)

Functional Parts

Observation of each of the collection's types has brought the attention to the distinction among three possible functional parts – laying, containing and holding, see Fig. 3, top – which were analyzed in terms of dimensions, namely height and radius. Different types feature different functional configurations, see Fig. 3, bottom. For example, to be able to contain liquids, the soup plate and the deep types feature a containing part that is taller than the dinner plate.

Generally, all three functions are present in each type. However, in some types they are assigned to parts other than the main body of the tableware element. For example, in the mug or the cup, the holding function is assigned to the handle. In its current state, this shape grammar is only encoding shape for the main body of the elements. Parts like the handles in the cups and mugs will be addressed in the future.

The resulting shape grammar, illustrated in Fig. 4, generates instances of the encoded collection, namely of its six types, taking into account the compositional and dimensional differences among them. Derivation of this shape grammar could be split into two phases: base shape definition and surface relief-based decoration, respectively illustrated in Figs. 5 and 6.

The grammar operates on 3D shapes, namely curved NURBS surfaces. For two-dimensional (2D) representation purposes, rules and derivations are presented in two different ways: for base shape definition, 3D surfaces are represented in cross-section drawings, while for decoration, they are represented in terms of their 2D parametric space, described by local coordinates (typically named 'uv parameters'), which can be continuously mapped into Cartesian space [19].

A Family of Collections

Since we are aiming at the customization of collections, the resulting shape grammar was built to be parametric. Therefore, the solution generated by it can

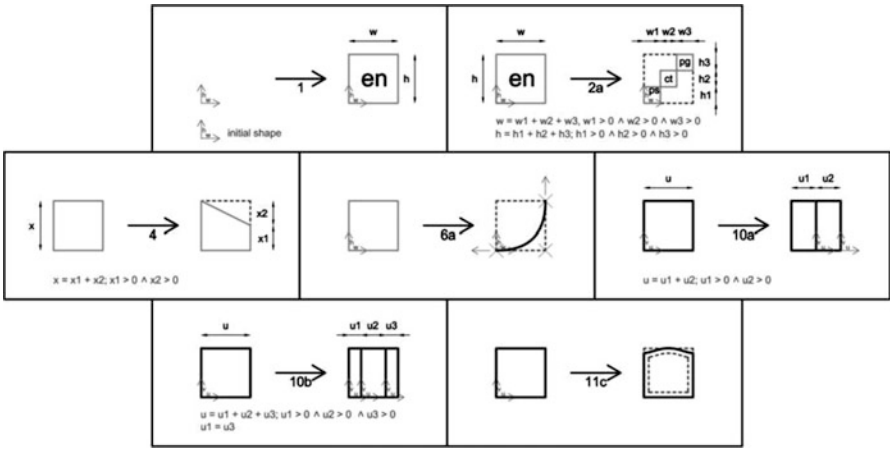

initial shape
1
en
w
h
2a
w1 w2 w3
pg
ct
ps
h3
h2
h1
w = w1 + w2 + w3, w1 > 0 ∧ w2 > 0 ∧ w3 > 0
h = h1 + h2 + h3; h1 > 0 ∧ h2 > 0 ∧ h3 > 0
x
4
x2
x1
x = x1 + x2; x1 > 0 ∧ x2 > 0
6a
u
10a
u1 u2
u = u1 + u2; u1 > 0 ∧ u2 > 0
10b
u1 u2 u3
u = u1 + u2 + u3; u1 > 0 ∧ u2 > 0 ∧ u3 > 0
u1 = u3
11c

Fig. 4 Shape grammar rules for generating a soup plate

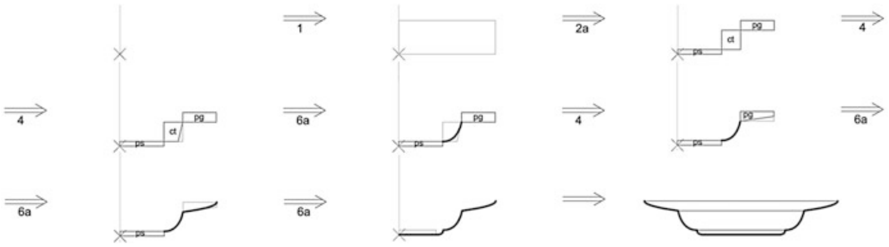


Fig. 5 Derivation of the base shape of a soup plate [18]

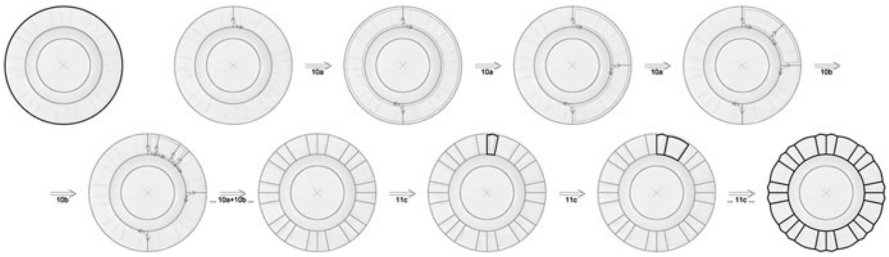


Fig. 6 Derivation of the decoration for a soup plate [18]

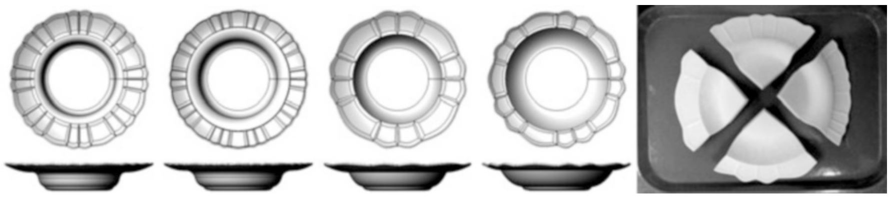


Fig. 7 Four parametric variations of the original collection: digital and physical models

be customized by the user, by specifying the solution's parameters [9]. Figure 7 is presented as an example, illustrating four different solutions, corresponding to four different configurations of parameters, namely regarding the number of subdivisions in the decoration.

So, to be precise, we should say that the shape grammar encodes not one collection but a family of collections, within the parametric space of solutions. Had not been parametric, this shape grammar would generate one single solution, which is usually considered pointless. Nevertheless, it was intended since the beginning to relate several collections, and thus such "pointlessness" would be justifiable.

The developed shape grammar was the basis for the implementation of a corresponding parametric model, in which parameters can be manipulated to generate different variations of the original collection, seen in Fig. 7, left. Some of these variations were prototyped using a powder-based 3D printer shown in Fig. 7, right.

Multiple Collection Shape Grammar

Despite only generating a single collection, this initial shape grammar was essential to determine a first set of principles for the further development of the design system, towards a generic shape grammar capable of generating several collections.

Following an approach similar to the one used by Benrós et al. [11] in the development of a generic shape grammar for housing, five additional different tableware collections were analyzed, encoded into corresponding specific shape grammars, adding to the first one. These first six collections were compared in order to understand and register the similarities and differences among them. Expectation was that the rules inferred from these first collections could be re-used to describe similarities identified in new collections, whereas differences would imply inferring new rules, thus further completing the generic grammar, and thus the design system.

As the new collections were analyzed, we verified that the base shape for each collection could be generated using the same rules inferred for the first encoded collection (henceforth named "collection A"), namely the rules governing the distribution of the functional parts, even if they were applied in different order or with different parameters.

Since it was an intention to explore different rules toward a generic grammar, this following exercise focused solely on the decoration of the ceramic tableware elements, whereas in collection A, both base shape and decoration were subject to analysis.

This comparative analysis was divided into two steps. In the first step, and similarly to the previous exercise, each of the five new collections was decoded into rules. However, contrary to the previous exercise, a shape grammar already existed, and therefore the rules for the new collection were inferred taking the existing grammar into account.

Let us consider as an example the partial derivation shown in Fig. 8, which concerns the decoration of a plate belonging to one of the five new collections

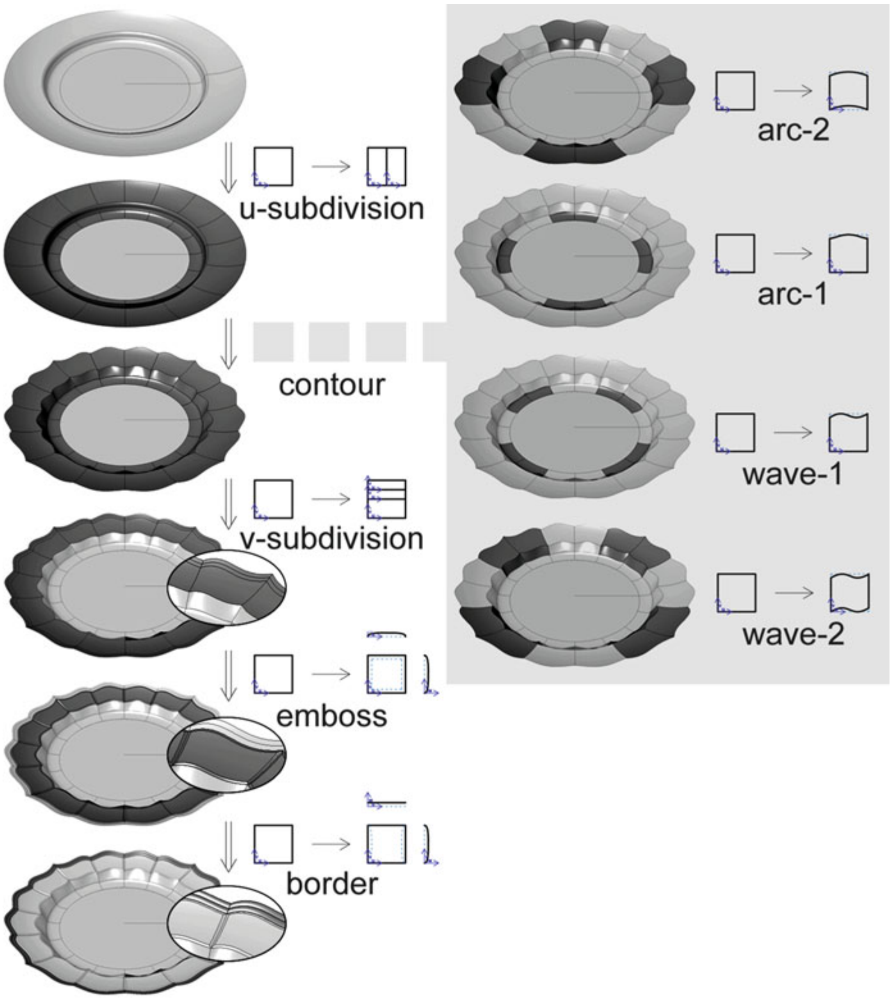


Fig. 8 Partial derivation of a plate from collection B

(henceforth named “collection B”). In this example, rules themselves are being shown instead of their corresponding numbers, since rule numbering has subsequently changed.

As expected, some rules from the grammar which generates collection A (or “grammar A”) were used, namely rules related to surface subdivision and substitution. However, rules from grammar A were not sufficient to describe collection B, and so new rules were created in order to obtain a grammar B. This method of obtaining a grammar B from grammar A follows Knight’s methodology for transformation of shape grammars through rule addition, rule deletion and rule change [20]. Therefore, in the previous example, grammar B can be considered a transformation of grammar A.

Though at first it was not intentionally applied, Knight’s methodology is easily recognizable in the process of inferring the rules for the new collections. Therefore, it will be knowingly referred to in future developments of the generic shape grammar.

This rationale was extended to all five collections. After having them inferred for each collection, the rules were laid out on a table (Table 1), including the rules of the original collection. This facilitates a visual analysis of the mentioned similarities and differences. In Table 1, columns correspond to collections (the shaded column corresponds to collection B), and rows correspond to rules. Rules were intuitively grouped together according to their similarity. These groups are separated by horizontal lines, and will be addressed further.

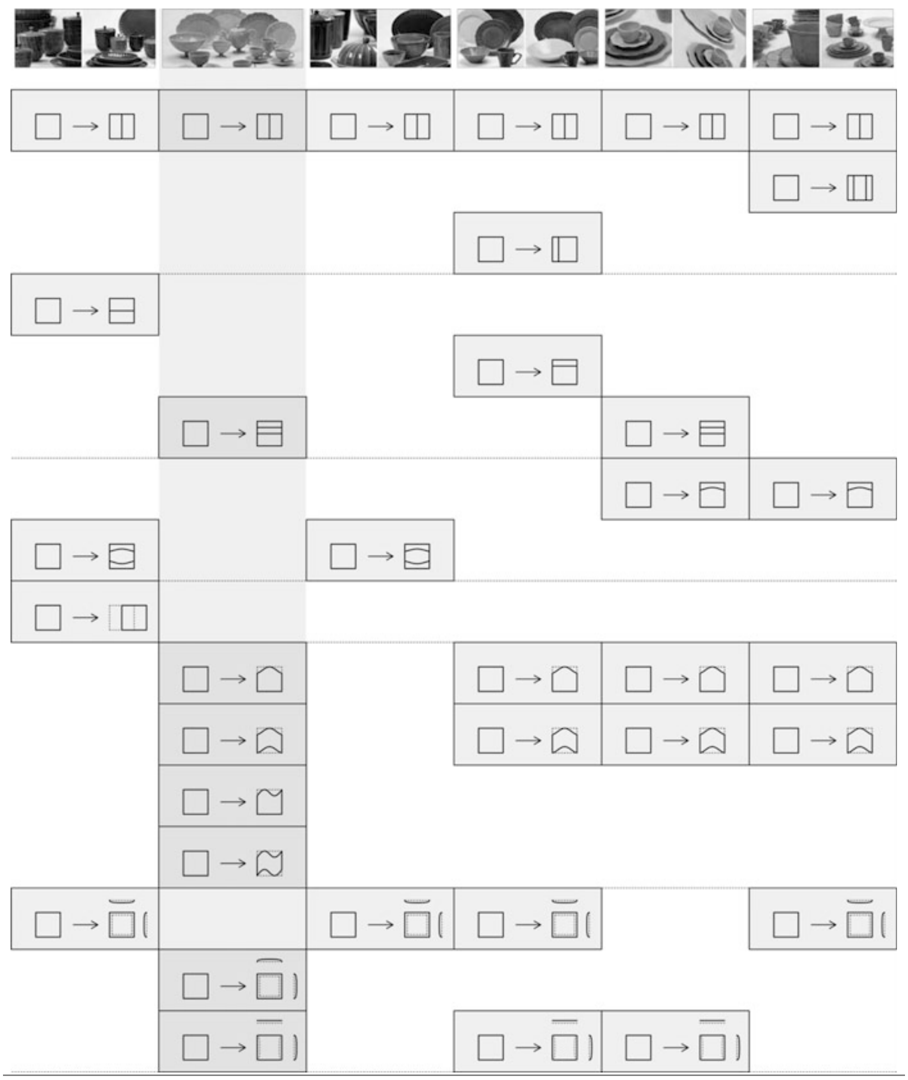
Grammar Debugging

Such comparative analysis, along with the implementation of the new collections into corresponding parametric models, brought the attention to some rules being very similar, suggesting that the grammar could be optimized, and rendering it leaner and thus more efficient. This led to the revision of some rules. Such revision typically implied deleting the revised rules, and adding new ones, towards a transformed generic grammar [20]. This resulted in a new arrangement of the specific grammars for each of the six collections. Table 2 presents these new arrangements in a similar fashion as in Table 1, identifying added rules with a shaded background, and deleted rules with dotted shapes.

Rule Decomposition

At this stage, revision of the rules consisted of decomposing them into typical atomic operations. Here, ‘typical operations’ refers to the operations that are recurrent in the process of surface modelling, which are found throughout several different modelling applications. This approach is similar to the one applied by

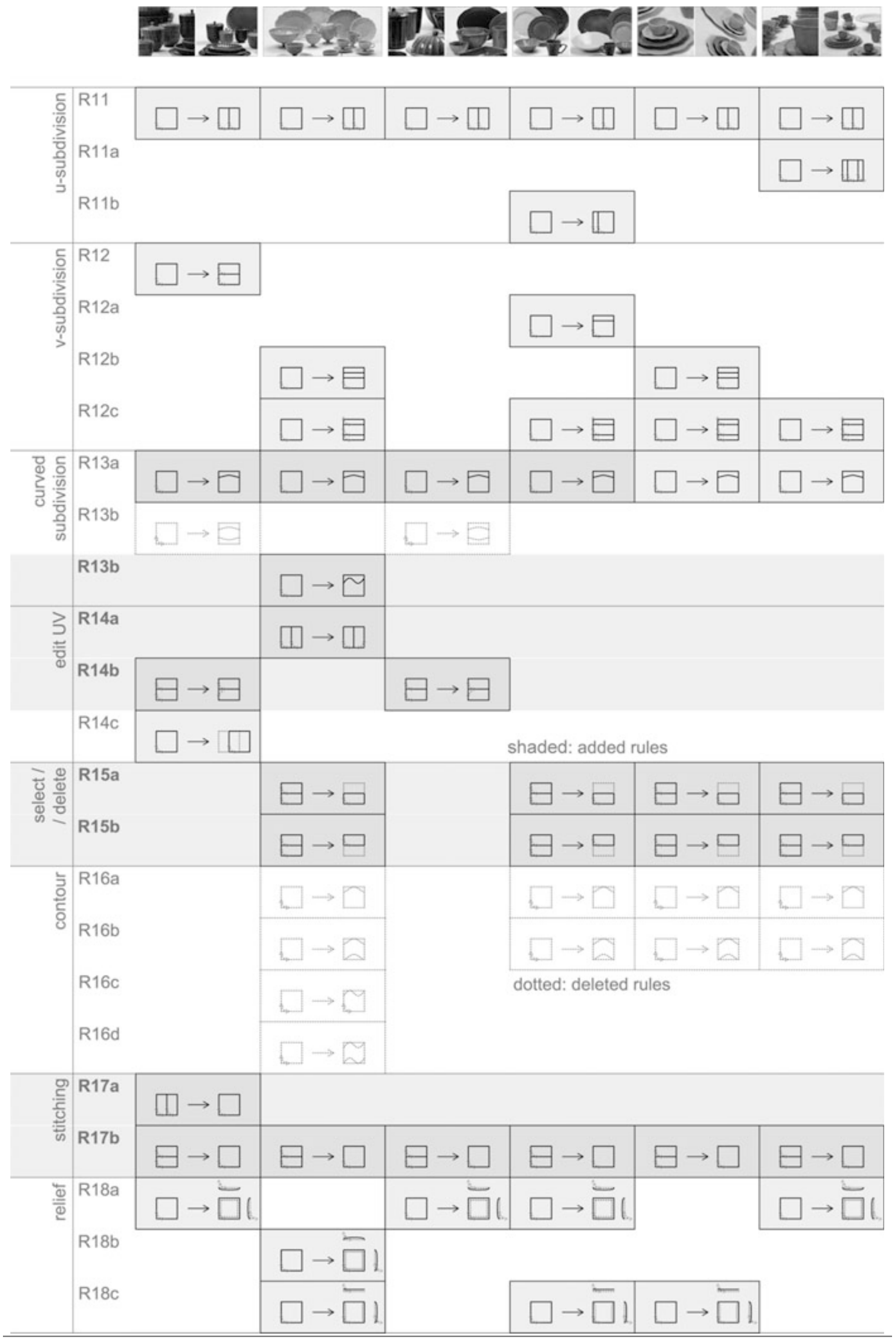
Table 1 Shape rules of the six encoded collections



Beirão et al. [10] to urban design, in which the concept of design patterns [21] is introduced to encode recurrent design moves. Such application of design patterns will be further addressed in future developments of the design system.

Figure 9 illustrates the rationale behind decomposition actions, showing, on the left, examples of four rules from Table 1 that were subjected to revision, and on the right, how they are decomposed into simpler rules. Beneath each example, it is demonstrated how to obtain the revised rule by derivating the simpler rules. Rules in Fig. 9 are numbered according to Table 2. Also accordingly, a shaded

Table 2 Shape rules after grammar revision



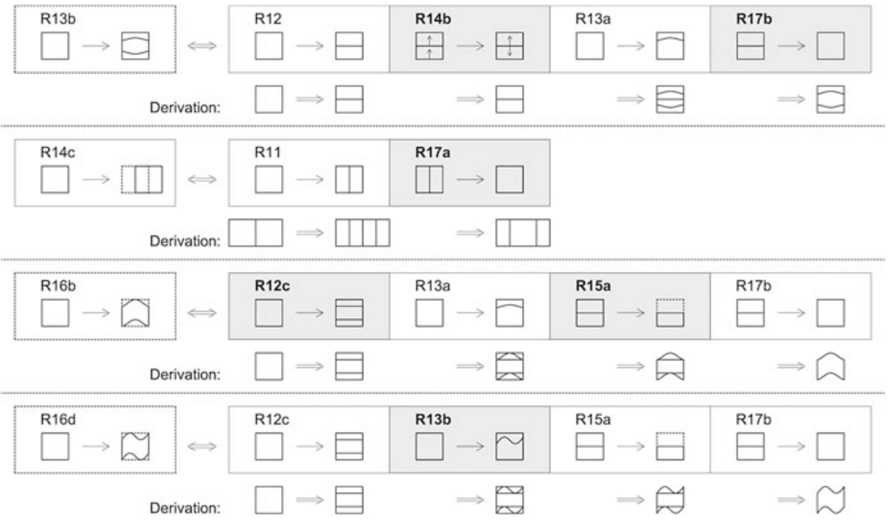


Fig. 9 Examples of rule decomposition

background corresponds to new added rules, and dotted shapes correspond to deleted rules.

Typically, revised rules were removed from the grammar, and replaced by their constituent simpler rules. Some of these simpler rules were part of the previous grammars, while other, new rules were inferred from the decomposition process.

Examples of this are shown in Fig. 9. In the first example, decomposition of Rule 13b implied the addition of Rule 14b for the horizontal flipping of a surface's parametric space, which is a typical operation for generating symmetry. In the second example, by decomposing Rule 14c, a stitching operation was inferred, implying the addition of Rule 17a that allows joining surfaces together. In this case, the original Rule 14c was not removed since it is expected to be more useful in the tableware design process than just its constituents. Decomposition of the other two rules, 16b and 16c follow the same logic.

The effort for decomposition brought a deeper understanding of what rules are "made of", on one side granting specific grammars more coherence among them, and on the other side making it possible to re-assemble the rules through implementation as subroutines into a computer program that represents the parametric model. This implementation process falls out of the scope of this paper, and so it will be further addressed in later publications.

Rule Generalization

The decomposition phase resulted on a re-arrangement of rules within the specific shape grammars that encode the six analyzed collections (Table 2). In order to

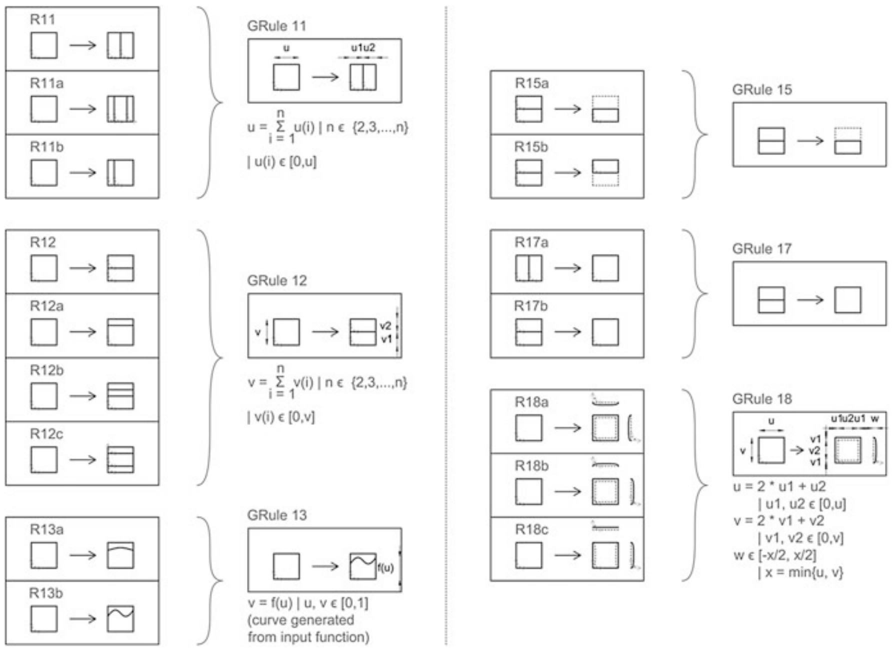


Fig. 10 Examples of rule generalization through parameterization

implement the updated grammar rules into parametric models, the number of rules would preferably be minimized.

Some work towards this minimization had already been done. Since the beginning of the comparative study, rules have been intuitively grouped according to the type of operation they correspond to (Table 1). Within the same group, rules are very similar amongst each other. It was quickly understood that these similar rules could be generalized into a single rule with different parameters. Therefore the number of rules could be minimized through generalization.

Figure 10 shows examples of generalization of rules through parameterization. The most paradigmatic cases are Rules 11 and 12, which govern the parametric subdivision of surfaces along u- and v-isoparametric curves respectively.

Rules 11, 11a and 11b subdivide a surface into a given number (n) of parts along a number (n-1) of isoparametric curves, which correspond to the given parameters u (n). Therefore, the parameters for a possible general form of Rule 11 (GRule 11) could be (n) numbers, corresponding to the relative sizes of the parts. For example, as seen in Fig. 11, given the parameters 25 %/50 %/25 %, GRule 11 returns three sub-surfaces, two smaller ones and a larger one – for the case of a uniformly parameterized surface. Given these parameters, GRule 11 returns the same result as the previous Rule 11a. The same applies for GRule 12, which governs subdivision of surfaces along curves that are isoparametric in v.

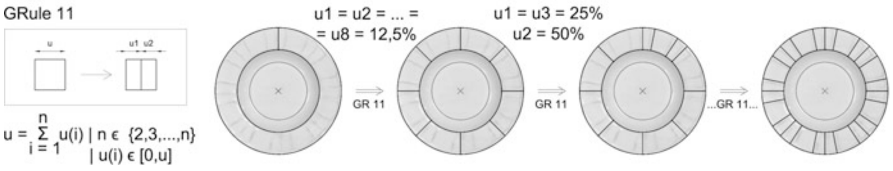

GRule 11
u
u1 u2
u = Σ u(i) | n ϵ {2,3,...,n} | u(i) ϵ [0,u]
u1 = u2 = ... = = u8 = 12,5%
u1 = u3 = 25%
u2 = 50%
GR 11
GR 11
...GR 11...

Fig. 11 Example for application of GRule 11

GRule 13 is a special case, since the required parameter is a curve, instead of a scalar parameter. Such input curve will determine the shape of the splitting curve, and it can be generated by a mathematical function, given its own parameters, or it can correspond to a shape drawn by the user.

GRule 15 is generalized from Rules 15a and 15b, which are equivalent rules, and therefore redundant. According to Krstic [22], "two rules are equivalent if whenever applied to the identical shapes the resulting shapes are identical". Also, two rules are equivalent if one is a transformed version of another such that the same transformation is applied to both left-hand side and right-hand side of the rule. Such is the case of Rules 15, which select or delete part of a surface that has been split along a horizontal line. In that case, mirroring Rule 15a along a horizontal axis generates Rule 15b. Analogously, a 90° rotation enables the use of general rule GRule 15 on surfaces split along vertical lines. The same applies to GRule 17, which joins, or stitches, two adjacent surfaces.

In the case of Rules 18, generalization allows to create different relief effects on surfaces, by manipulating parameters such as border thickness and depth. For example, assigning a positive or negative value to the depth parameter generates a high or low relief, respectively.

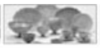
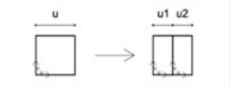
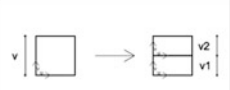
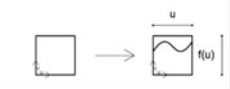
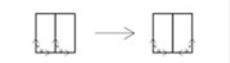
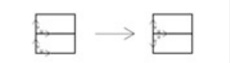
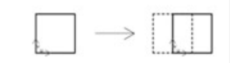
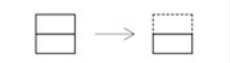
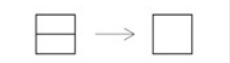
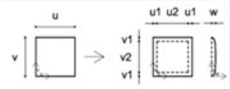
Shape Grammar Application

The general rules obtained with the generalization phase constitute a first draft of the generic grammar (Table 3).

Along the development process of this generic grammar, its rules had been implemented as parametric geometrical operations into a visual programming interface. By manipulating such operations, combining them and defining their parameters, collections are generated as digital models.

In order for this first generic grammar to be validated, it should be able to execute two kinds of tasks. On one hand, it should be able to reproduce the original collections and their designs. This was tested successfully along with the implementation. On the other hand, it should allow to creatively generating new collections. In order to test this ability, three new original collections were generated through derivation, using the general rules, illustrated in Fig. 13. The combinations that generated these collections were somewhat random, lacking any intention to resemble more or less the original collections.

Table 3 Six collections encoded with general rules and parameters used

							
GR11		1:1	1:1	1:1	1:1 1:3	1:1	1:1 1:2:1
GR12		1:1	1:1:2 1:2:1		1:3 1:2:1	1:1:2 1:2:1	1:2:1
GR13		(gauss)	(gauss) (sine)	(gauss)	(gauss)	(gauss)	(gauss)
R14a			USED				
R14b		USED		USED			
R14c		USED					
GR15			(top) (bottom)		(top) (bottom)	(top) (bottom)	(top) (bottom)
GR17		(horiz) (vert)	(horiz)	(horiz)	(horiz)	(horiz)	(horiz)
GR18		-u:-v	+u:+v +v	-u:-v	-u:-v +v	+v	-u:-v

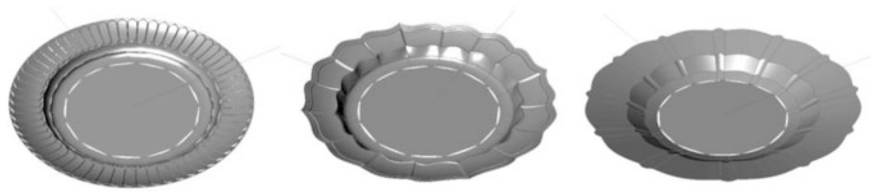


Fig. 12 Elements of the original collections generated by the generic shape grammar

Observing the new designs in Fig. 13, and considering examples from the original collections in Fig. 12, we can verify that (a) the new designs are relatively different from the original collections, and (b) despite that, their constituting elements – namely decorative ones – can be recognized from the original collections. This suggests that the generic grammar has the potential for generating new designs, while ensuring formal coherence.

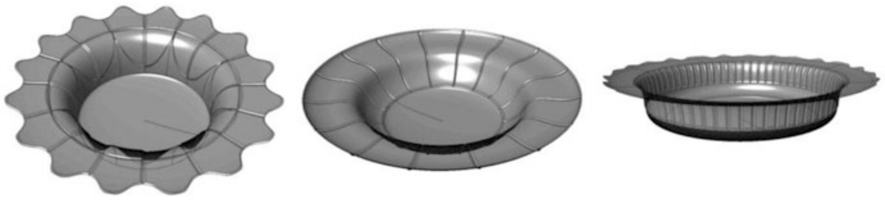


Fig. 13 Elements of three new collections generated by the generic shape grammar

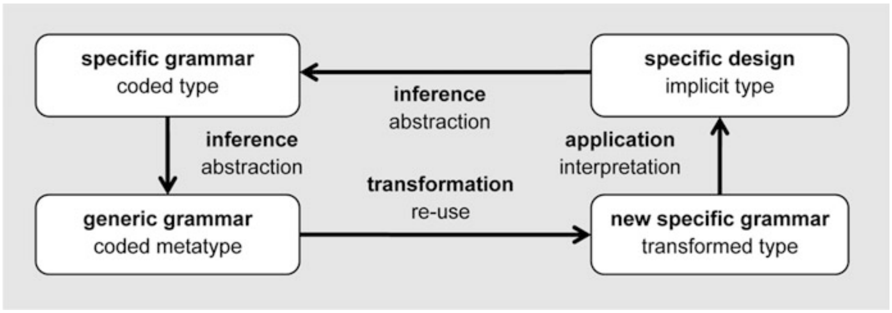


Fig. 14 Coding processes for grammars and types (Adaptation from [23])

Discussion

Until now, the development of a generic grammar for the mass customization of ceramic tableware has been documented. Along this task, a question has emerged: how generic is a generic shape grammar?

In related literature, the terms ‘generic’ and ‘specific’ have often been used as absolute properties of shape grammars. For example, in Benrós et al. [11] three specific shape grammars are abstracted into one generic shape grammar. Likewise, in Mendes et al. [13, 14] specific grammars are analyzed for future abstraction into a generic grammar. In both cases, the reduced number of grammars in question justifies the polarization of the terms ‘generic’ and ‘specific’.

The relationships between ‘specific’ and ‘generic’ are synthesized by Duarte [23] in Fig. 14, explaining the states and actions for generalizing and specifying shape grammars into new designs. However, the polarity between ‘specific’ and ‘generic’ is still present in this model.

In Beirão et al. [10, 24], a hierarchy of shape grammars structures the relationships among Urban Induction Patterns and generic and specific Urban Grammars. Such a hierarchy, together with expressions like “very generic grammar”, suggests the use of ‘generic’ and ‘specific’ as relative terms. In the case of the tableware shape grammar, we anticipate the need for handling a wider range of hierarchical levels.

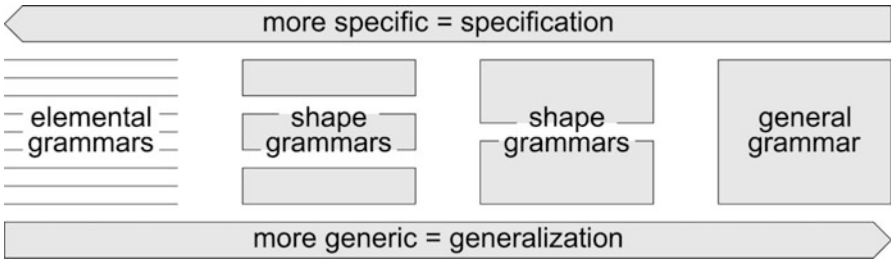


Fig. 15 Shape grammar genericness spectrum

Until now, six collections have been encoded into corresponding specific shape grammars, which in turn have been combined into the current generic grammar. However, in order to extend the design system, many more collections are to be encoded. Therefore, it can be said that the generic grammar resulting from this extension is ‘more generic’, or ‘less specific’, when compared to the current one. Therefore, in the scope of this investigation, we consider a grammar to be ‘more generic’ when it is able to generate a wider range of collections than a ‘more specific’ grammar.

As seen previously in this paper, a more generic grammar can be inferred from combining two or more specific grammars. On the other hand, a more specific grammar can be extracted from a more generic grammar through a process of specification, by selecting rules from the latter, and by specifying the parameter interval for these rules.

We suggest the use of the expression ‘general grammar’ to designate the most generic grammar at a given time, thus corresponding to the actual state of the design system. The general grammar contains all the rules inferred from every analyzed collection, thus being able to generate each and every one of those collections.

It is possible to constrain the generation of each of these collections through specification of the general grammar. We suggest designating the resulting specific shape grammars, which are only able to generate a single family of collections, as ‘elemental grammars’.

‘Genericness’ is the term suggested for a shape grammar’s quality of being more generic or more specific. Although genericness has not been quantified, it seems possible to order two shape grammars relative to each other according to that quality. Therefore it seems to be a relevant parameter for controlling a wider hierarchy of shape grammars, needed for extending the mass customization experience of ceramic tableware. The relations between general, generic, specific and elemental shape grammars are shown in Fig. 15.

Conclusion

This paper demonstrates the validity and benefits of using generic shape grammars for the design of ceramic tableware. Different tableware collections were encoded into elemental shape grammars and compared. This comparison between grammars revealed their similarities and differences, providing clues for the development of a generic shape grammar, through iterative processes of transformation, decomposition and generalization. Such actions provided a deeper understanding of the general grammar, down to its smallest elements, while at the same time allowing for a broader vision of its relationships. We could then fine-tune the rules, towards a more efficient design system, able to generate coherent languages of tableware collections. It is our belief that such control over the design system is necessary for providing the experience of manipulating shape grammars to the end user of a mass customization system.

Acknowledgments Eduardo Castro e Costa is funded by FCT (Fundação para a Ciência e a Tecnologia) with PhD grant SFRH/BD/88040/2012, and by CIAUD (Centro de Investigação em Arquitectura Urbanismo e Design) with financial support for the present publication and corresponding communication.

References

1. Toffler A (1971) Future shock. Bantam Books, New York
2. Toffler A (1984) The third wave. Bantam Books, New York
3. Pine BJ (1993) Mass customization: the new frontier in business competition. Harvard Business Press, Boston
4. Bernard A, Daaboul J, Laroche F, Cunha CD (2012) Mass customisation as a competitive factor for sustainability. In: ElMaraghy HA (ed) Enabling manufacturing competitiveness and economic sustainability. Springer, Berlin, pp 18–25
5. Piller FT (2004) Mass customization: reflections on the state of the concept. Int J Flex Manuf Syst 16:313–334
6. Duarte JP (2008) Synthesis lesson – mass customization: models and algorithms – aggregation exams. Agregação, Faculdade de Arquitectura, Universidade Técnica de Lisboa
7. e Costa C, Duarte JP (2013) Mass customization of ceramic tableware through digital technology. In: Bartolo H, Bartolo P (eds) Green design, materials and manufacturing processes. CRC Press, Lisboa, pp 467–471
8. Stiny G, Gips J (1972) Shape grammars and the generative specification of painting and sculpture. In: Freiman CV (ed) Information processing, vol 71. North Holland, Amsterdam, pp 1460–1465
9. Stiny G (1980) Introduction to shape and shape grammars. Environ Plan B: Plan Des 7:343–351
10. Beirão JN, Duarte JP, Stouffs R (2011) Creating specific grammars with generic grammars: towards flexible urban design. Nexus Netw J 13:73–111
11. Benrós D, Hanna S, Duarte JP (2012) A generic shape grammar for the Palladian Villa, Malagueira House and Prairie House. Design computation and cognition DCC'12
12. Li AI-K (2001) A shape grammar for teaching the architectural style of the Yingzao fashi. PhD, Massachusetts Institute of Technology
13. Mendes L, Beirão JN, Duarte JP, Celani G (2013) A bottom-up social housing system described with shape grammars. In: Stouffs R, Sariyildiz S (eds) Computation and performance – In: Proceedings of the 31st eCAADe conference. Faculty of Architecture, Delft University of Technology, Delft, p 705–714

14. Mendes L, Beirão JN, Celani G (2013) Meta-PREVI – Uma meta-gramática para a geração de habitação de interesse social. In: Bernal M, Gomez P (eds) Anais do XVII Congresso SIGRADI. Valparaiso, pp 217–221
15. McCormack JP, Cagan J, Vogel CM (2004) Speaking the Buick language: capturing, understanding, and exploring brand identity with shape grammars. Des Stud 25:1–29
16. Pugliese MJ, Cagan J (2002) Capturing a rebel: modeling the Harley-Davidson brand through a motorcycle shape grammar. Res Eng Des 13:139–156
17. Barros M, Duarte JP, Chaparro B (2011) Thonet chairs design grammar: a step towards the mass customization of furniture. In: Proceedings of the 14th international conference on computer aided architectural design futures. pp 181–200
18. Castro e Costa E, Duarte JP (2013) Tableware shape grammar. In: Stouffs R, Sariyildiz S (eds) Computation and performance – In: Proceedings of the 31st eCAADe conference. Faculty of Architecture, Delft University of Technology, Delft, pp 635–644
19. Pottmann H, Asperl A, Hofer M, Kilian A (2007) Architectural geometry, 1st edn. Bentley Institute Press, Exton
20. Knight TW (1994) Transformations in design: a formal approach to stylistic change and innovation in the visual arts. Cambridge University Press, New York
21. Gamma E, Helm R, Johnson R, Vlissides J (1994) Design patterns: elements of reusable object-oriented software. Pearson Education, Upper Saddle River
22. Krstic D (2012) Algebras of shapes revisited. Design computation and cognition DCC'12
23. Duarte JP (2011) Mass customization: models and algorithms. In: Keynote lecture at eCAADe 28th international conference of the education and research on computer aided architectural design in Europe
24. Beirão J, Duarte J, Stouffs R, Bekkering H (2012) Designing with urban induction patterns: a methodological approach. Environ Plan B: Plan Des 39:665–682

Part VII
Design Support

Using Text Mining Techniques to Extract Rationale from Existing Documentation

Benjamin Rogers, Yechen Qiao, James Gung, Tanmay Mathur, and Janet E. Burge

Abstract Software development and maintenance require making many decisions over the lifetime of the software. The decision problems, alternative solutions, and the arguments for and against these solutions comprise the system's rationale. This information is potentially valuable as a record of the developer and maintainers' intent. Unfortunately, this information is not explicitly captured in a structured form that can be easily analyzed. Still, while rationale is not explicitly captured, that does not mean that rationale is not captured at all—decisions are documented in many ways throughout the development process. This paper tackles the issue of extracting rationale from text by describing a mechanism for using two existing tools, GATE (General Architecture for Text Engineering) and WEKA (Waikato Environment for Knowledge Analysis) to build classification models for text mining of rationale. We used this mechanism to evaluate different combinations of text features and machine learning algorithms to extract rationale from Chrome bug reports. Our results are comparable in accuracy to those obtained by human annotators.

Introduction

Software design and development requires making many decisions throughout the software lifecycle. These decisions range from early decisions on what the requirements are to later ones on how to respond to change requests during maintenance. These decisions, the alternative solutions considered, and the justifications for choices made (or rejected) comprise the rationale for the system. This rationale, more commonly known as design rationale (referring to engineering disciplines

B. Rogers (✉) • T. Mathur • J.E. Burge
Miami University, Oxford, OH, USA
e-mail: rogersbc@miamioh.edu

Y. Qiao
University of Pittsburgh, Pittsburgh, PA, USA

J. Gung
University of Colorado, Boulder, CO, USA

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_26

where the decision-making process is more active prior to manufacturing) has many potential uses. Understanding the intent of the decision-maker can help to prevent issues further on during development and maintenance. Potential uses for rationale include documentation, traceability, impact assessment, and decision evaluation [1].

A major obstacle towards design rationale becoming a standard part of working practice has been "the capture problem"—the perceived high cost and effort involved in collecting and structuring rationale. The capture problem has been described as "the spectre haunting all design rationale efforts" [2]. One possible solution to this problem would be to extract rationale from existing documentation sources. While rationale is not explicitly captured, there are many sources of information about the design and implementation of a software system that may contain rationale that could be used together to capture its rationale.

There are three major research questions that need to be answered to successfully build a base of rationale from existing documents:

1. Which types of documents contain rationale?
2. How can the rationale be identified and extracted automatically?
3. What tools and support need to be provided to structure the rationale into a format that is useful to developers?

We have chosen to initially focus on the second question—identifying tools and techniques for automatic rationale extraction because this question must be answered to provide motivation for the first question and inputs to the third. We identified bug reports as a potential source of rationale (since they contain a description of a software defect along with a discussion of how it could be repaired) and have been working with a training/test corpus of Chrome web browser bug reports. This paper makes the following contributions:

1. Preliminary assessment of bug reports as a (not the) source of rationale by identifying the relative frequency of different types of rationale.
2. Comparison of the difficulty in learning rationale at varying levels of specificity.
3. Comparison of different learning algorithms in detecting rationale.
4. Comparison of different feature combinations in detecting rationale.
5. Creation of a process for combining two tools, GATE (General Architecture for Text Engineering) [3] and WEKA (Waikato Environment for Knowledge Analysis) [4], to build text classifiers. This process could be applied to any sentence-level text classification problem using any text data source, where only the training/test data annotation is application specific.

Section "Approach" of this paper describes our approach to training a rationale classifier, section "Results" presents our results, section "Related Work" describes related work, and section "Conclusions and Future Work" provides conclusions and future work.

Approach

Automated rationale extraction is a multi-step process that first requires training a machine-learning classifier to do the extraction. This requires integrating two tools, GATE and WEKA. GATE was used to manually annotate our training/test set as well as to automatically annotate linguistic features that may be useful in classification. We also developed additional annotation scripts for features not supported in GATE. After both manual and automated annotation, we used a Text Feature Extractor developed for this project to extract the features of interest for each sentence. Then we converted the sentence files into the format required by WEKA (ARFF – attribute-record file format). WEKA was then used to build and evaluate the classifier, using tenfold cross validation for evaluation (a technique for splitting data into test and training sets and averaging the results). WEKA produces a single classifier built from the training data as its final output. Figure 1 shows the steps involved in pre-processing and classifier training.

The classifier produced by WEKA would be used in a similar fashion to classify new instances. In that case, the pre-processing in GATE would be similar except without the manual annotation step and WEKA would use the model built earlier during training for classification. We have developed software to convert the classified bug reports into GATE-format XML (eXtensible Markup Language) so they can be imported into a Rationale Management System as future work.

The following sub-sections describe our methodology for pre-processing the training/test data, identifying candidate feature sets, and performing the classification.

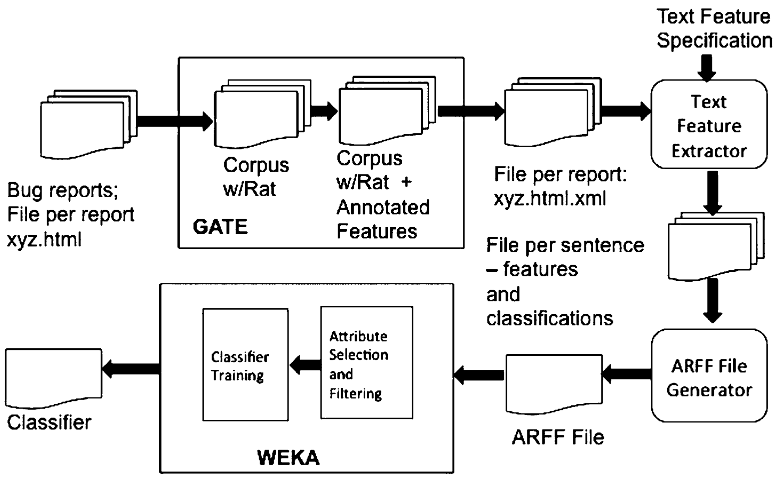

Text Feature Specification
Text Feature Extractor
Bug reports; File per report xyz.html
Corpus w/Rat
Corpus w/Rat + Annotated Features
GATE
File per report: xyz.html.xml
File per sentence – features and classifications
Attribute Selection and Filtering
Classifier Training
WEKA
Classifier
ARFF File Generator
ARFF File

Fig. 1 Classifier training

Annotation and Pre-processing

Data pre-processing has a significant effect on the success of classification. For this work, we had to first prepare a corpus to train the classifier and then pre-process the data to support training as well as classification. The following sub-sections describe the processing required prior to classifier training.

Data Annotation

The data source used for this paper was a set of 200 Chrome bug reports extracted randomly from the data provided for the Mining Software Repositories 2011 mining challenge (http://2011.msrconf.org/msr-challenge.html). While the selections were random, they were constrained to ensure that the bug reports selected had the potential to contain rationale. We restricted our training set to bug reports that were greater than 20 KB in size.

We initially annotated the bug reports looking for eight different types of rationale:

- *Requirements* – features that the software was supposed to provide.
- *Decisions* – statements of problems to be solved and choices that needed to be made.
- *Alternatives* – alternative solutions to problems/alternative design options.
- *Arguments* – positions taken for and against the alternatives.
- *Assumptions* – statements where the author/contributors are not sure that they are stating a definite fact, but instead are articulating a belief or conditional statement.
- *Questions* – questions that need to be answered before a solution can be determined or alternative selected.
- *Answers* – answers to questions posed by the document author.
- *Procedures* – descriptions of what needs to be done in order to answer questions posed in the document.

Each bug report had three researchers involved in the annotation process. Two of the researchers annotated the document independently, while the third researcher adjudicated the annotations. This ensured that each annotation was agreed upon by two of the three researchers. The bug reports were split into groups of ten so that the annotators and adjudicators could be varied. The annotation was done using GATE [3]. An annotation guide was developed to assist in consistent annotation of rationale.

Figure 2 shows an example of an annotated bug report segment. The types are denoted by their name in bold, with brackets identifying the text, shown in italics, that was given that type.

Decision: [*There is a compatibility difference in how Chromium and Firefox handle IPv6 in PAC scripts versus Internet Explorer*.] **Argument:** [*This can cause scripts that work fine in IE to behave differently in Chromium/Firefox, resulting in non-functional configuration. For example, the "myIpAddress()" PAC library function may return an IPv6 address in chromium/firefox. Whereas in in Internet Explorer it will only returns IPv4 addresses.*](Similarly for other functions with DNS resolving dependencies: dnsResolve(), isResolvble(), isInNet()). **Argument:** [*Since existing scripts written for IE don't expect IPv6 ad-dresses to be returned, when one is,*] **Assumption, Argument**:[*it may pass through the wrong codepath and things won't work.*] **Argument:** [*This is a known issue in Firefox (try searching for "ipv6 myIpAddress pac").]* **Argument:** [*Here is a thread where users describe hitting it:]* http://support.mozilla.com/tiki-view_forum_thread.php? lo-cale=tr& comments_parentId=259063 &forumId=1 **Alternative:** [*Internet Explorer has worked around the problems with the PAC standard by defining the existing functions to be IPv4-only, and adding extensions to support IPv6:*] http://blogs.msdn.com/wndp/archive/2006/07/18/IPV6-WPAD-for-WinHttp-and-W inInet.aspx **Alternative:** [*I think that we should similar y to IE and define the old PAC functions to be IPv4 only*]**Argument:** [*since there are learly users hitting this problem.]***Alternative:** [*Currently the only workaround for Chrome users is to use the--winhttp-proxy-resolver flag. (If we had a flag to disable IPv6 that would work too).*Comment 1 by dvy12...@gmail.com , Oct 14, 2009 **Alternative:** [*I suggest adding another IPv6 version myIpAddress() function, may be called myIpv6Address().*] **Argument:** [*auto-config script PAC file is very very useful stuff, I use it into Chromium in my every day web browsing.*]

Fig. 2 Example bug report rationale

Data Pre-processing in GATE

In addition to supporting manual data annotation, GATE allows documents to be annotated automatically using plug-ins provided with the tool-kit and through writing JAPE (Java Annotation Pattern Engine) transducers. We used plug-ins provided with GATE to perform tokenization, part-of-speech (POS) tagging, sentence splitting (with some customizations we provided to take advantage of the html tags within the data), verb group chunking, and stemming.

We performed additional pre-processing using our own JAPE transducers to do the following:

Contextual Information Annotation. Some sentences can only be classified as rationale when we also take into account their neighbors. For example, a part of an argument could be a factual sentence that does not contain cue words that signal it as an argument without the context of the sentences around it. A short answer next to a rhetorical question may be rationale but may not contain enough information by itself. A JAPE transducer was coded to consider features in a sentence's neighbors and weight them by their distance from the target sentence.

Sentence Length Annotation. Sentence length can be a good measure for determining whether a sentence is of a particular rationale type, and whether it is rationale at all. For example, it is difficult to form a Question in less than three words, and difficult to form a Decision in less than five words. Some sentences are extremely long (our dataset has a maximum sentence length of 1,656 words). Sentences that are very long are usually machine-generated reports that are missing full stops and newline characters. These sentences are highly unlikely to contain rationale and are far more likely to be noise, such as stack traces.

After this initial pre-processing, the result is a GATE document corpus with annotations denoting the different types of rationale, tokens, sentences, parts-of-speech, sentence length, relationship to neighboring sentences, and the lemma and affix for each token.

Feature Extractor

Before WEKA can process the document corpus, the GATE documents need to be converted to create an initial ARFF (Attribute-Relation File Format) file that contains the GATE attributes needed to build the classifier. We wrote a Feature Extractor that processes the corpus to create a text file for each sentence with features selected by the analyst. The text files contain tags for each requested GATE feature (for example, if POS-tags were selected then the sentences contain POS-tags for each token in the sentence). The Feature Extractor also supports *n*-gram tokenization (*n*-grams are groups of words where *n* refers to the number of words in the group). Token *n*-grams are intuitively a useful feature in rationale detection. Sentences that express rationale often have repeating grammatical patterns. While these cannot be represented in a typical bag-of-words approach to feature selection, sequences of words, POS-tags, or a combination of these can capture these patterns. Word, lemma, and POS *n*-grams are all potentially useful features in both rationale detection and classification. The analyst could specify both the type and length of the n-grams in the Feature Extractor. After performing Feature Extraction, we used another application we wrote to generate a WEKA-readable ARFF file.

Data Pre-processing in WEKA

Data pre-processing is required for feature selection, where the analyst specifies which features they wish to use to build the classifier used to detect rationale, and filtering, to reduce the number of features used in order to decrease the time required to run an experiment.

It is often the case that after the features have been picked, the number of features is too large to run an experiment in a short amount of time. For example, calculating all 5-g for all words on our dataset results in more than 250,000 features. In this kind of situation, a filter was used to statistically reduce the number of features so that only the most useful features remain.

By default, WEKA filters out all but the most 1,000 frequent features per class. While some experiments such as POS-only or Verbs-only may not have enough features for filtering to make a difference, feature sets such as 5-g far exceed the 1,000 feature limit. The frequency of a word appearing does not necessarily indicate that this word provides useful information. For example, conjunctions such as "on", "in", "at" appear more often than adjectives such as "stable", "secure", and yet do not contain nearly as much information as those adjectives. Some pre-processing schemes perform stop-word removal to remove words of this type but we chose not to remove them since they are often critical to grammatical patterns that indicate rationale.

Due to these reasons, we have pursued an alternative way of filtering features in the feature set. Information gain is a measure of the amount of uncertainty reduced regarding a target class when a particular feature is used. Features with the highest information gain will likely be the most useful features in application. Specifically, rather than relying on frequency, we use only the 1,000 attributes with highest information gain for training. To prevent the filter from peeking at the test data, the information gain attribute selection is done after the cross-validation test maker produces separate training and test datasets. WEKA's information gain filter selects attributes based on the entropy of the attributes in the training set.

Information gain attribute selection is a slow process. In order to make the filter efficient, we also applied a type of unsupervised filtering which looks at the data set as a whole and eliminates any features that appear less than three times in the corpus.

Classification Approach

There were three main factors we varied in our classification experiments. The first was our classification goal, the second was the learning algorithm, and the third was the combination of features used in training our classifier.

Classification Goal

While our ultimate goal is to identify rationale by type, we are taking an incremental approach towards that by identifying the following sets of rationale:

- *Rationale-binary*. Here we are interested if a sentence is rationale or not rationale.
- *Argumentation-subset-binary*. Some of the rationale in the bug reports, particularly the questions, answers, and procedures, are in a "boilerplate" section at the start where the person writing the report is asked how to reproduce it (this boilerplate text comes from the bug reporting system and is the same for all bug reports). We are more interested in identifying the other types of rationale (requirements, decisions, alternatives, arguments, and assumptions) since those

comprise the argumentation. We have been running experiments at identifying argumentation vs. non-argumentation.

- *Argumentation-types* {*Dec*, *Alt*, *Arg-all*}. One of the challenges in annotating the rationale was detecting requirements and assumptions. An end-user reporting that Chrome "must" behave a certain way is not necessarily stating something that a Chrome developer would consider a requirement. Assumptions are often relative to the person posing them (they may be uncertain but it could still be a factual statement). Since requirements and assumptions are forms of argument, we are combining them for classification purposes.

The incremental approach is useful because it points out cases, such as with the bug report boilerplate, where additional pre-processing may be necessary to improve results. We are also considering a two-stage classification process where we start with identifying sentences that are rationale and then perform classification by type on only those sentences.

Learning Algorithms

WEKA implements over 40 learning algorithms. We used a subset of WEKA's individual classifiers based on results from [5]. We also experimented with Ensemble Learning. All classification evaluation was done using tenfold cross validation rather than providing separate test and training data. This allowed and us to make the most efficient use of our data.

Three popular forms of Ensemble Learning are boosting, voting, and stacking. We used the AdaBoost M1 [6] classifier provided by WEKA. Classification using AdaBoost develops a classification model by iteratively generating and evaluating another classification model, where each iteration gives a larger weight to incorrect instances than correct ones. A major drawback to using AdaBoost is caused by large numbers of iterations leading to over-fitting the test data. Due to the fact that our corpus is relatively small, we attempted to minimize this problem by running AdaBoost for only five iterations We also used Stacking in WEKA. Stacking, or Stacked Generalization [7], combines multiple learners where each learner produces a prediction output. The different base learners are given different weights based on their performance. A meta-classifier for stacking decides how much weight a particular base learner should have (i.e., how much trust should meta-classifier put on a base learner). Ting and Witten [8] found that a simple linear model performed best as a stacking meta-classifier, so we chose NaiveBayes for our experiments.

Feature Combinations

When deciding which feature combinations to use, we started with relatively simple feature combinations and then extended them based on early results and the desire to investigate the impact of specific feature types. In particular, we were curious about differences between treating sentence length as a single numeric feature (a continuous attribute) or as individual features; comparing the use of POS tags vs. word roots; and comparing all word roots to lower dimension data-sets that only used particular parts of speech that had been observed as occurring frequently in rationale.

Results

As stated in the introduction, there are five contributions from this work: (1) A preliminary assessment of bug reports as a rationale source made by identifying the relative frequency of different types of rationale found during manual annotation; (2) A comparison of the difficulty of classifying rationale at different levels of granularity; (3) A comparison of different learning algorithms; (4) A comparison of different feature combinations; and (5) A process for combining GATE and WEKA to build text classifiers. Our results are given in the following sections.

Composition of Annotated Rationale

Table 1 shows the count of each type of rationale found in the set of 200 randomly selected bug reports. There are some sentences that were classified as being of multiple types. In particular, requirements and arguments often had overlap.

These counts show that arguments appear most frequently in the rationale, especially when all three categories are combined. Questions come next, even outnumbering decisions. Questions, procedures, and answers are all common elements, comprising nearly half the rationale elements. This suggests that if these elements are easier to classify than others (which is likely since many of

Table 1 Rationale types found

Element type	Sentences
Decision	424
Alternative	352
Argument	494
Requirement	76
Assumption	79
Argument-all {Arg, Req, Assumption}	634
Question	487
Procedure	169
Answer	397
Non rationale, total	17,410
Rationale, total	2,131

them come from "boilerplate" at the start of each bug report) then including them in classification results is likely to present results that are better than if classification only looks for argumentation. In total, only 10.9 % of the sentences contained rationale of any type, making this an unbalanced dataset.

Impact of Classification Goal

Our results, the best of which are shown in our Summary of Results, show that the more detailed the classification level, the poorer the results. When classifying all the rationale, including the questions, answers, and procedures, as either rationale or not rationale we were able to obtain a best F-1 measure of 0.677. Our best F-1 dropped to 0.569 when we were only looking at argumentation. When we tried to get more detailed classification, by specifically looking for decisions, alternatives, and arguments (all forms) we did well looking for decisions with a best F-1 score of 0.795. The results were much weaker for alternatives (0.36 for the best F-1) and arguments (0.373 as the best F-1). We suspect the higher score for decisions was because of two factors: the title of the bug report was always a decision and the bug report title was often repeated more than once.

The decreasing F-1 scores are not surprising. The difference between classifying rationale with categories included in the bug report specific boilerplate and only looking for the argumentation suggests that we may want to consider an additional pre-processing step that removes boilerplate prior to classification.

Impact of Learning Algorithm

In experiments using ensemble learning, stacking out-performed Ada Boosting. This is because it takes multiple learners into consideration, including one using Ada Boosting, and learns over the success of each of its constituent learners. For the best performing feature set (VADCC2_PSn: 2-closest sentences, verbs, adverbs, proper nouns, determiners, conjunctions, sentence length and POS tags), the F-1 measure was 7 % better than the closest individual classifier. Stacking was the slowest algorithm to run because it required combining results from multiple base learners.

Impact of Feature Selection

Many of our experiments were designed to learn the impact of including various features on our classification results. The following sections give our findings.

Impact of Feature Filtering Method. Compared with using the general and domain ontologies with stemming and WordNet expansion [5], using the best 1,000 information gain on the dataset from [5] yielded an average of 3.14 % higher F-measure using lemma of unigrams.

Using the filter with minimum term frequency of 3, we managed to drop the dimensionality of 5-g feature set from 254,120 to 23,869.

Impact of Sentence Length. We have attempted to incorporate sentence length in two ways: as separate features and as a single numerical feature. In the first method, we divide the sentence length into intervals of 5 (required to deal with the sparsity that would result if every possible sentence length was treated as a feature). In the second method, we include one feature, sentLength, which indicates the length of the sentence in its word count value in WEKA.

When we used the first method, we found out that the feature sent Length 0, which is a characteristic of sentences of length one to five words, has the highest information gain. There are many other sentence length intervals that were also in the top 1,000 when calculating information gain. However, we also found out that as sentence length increased, the number of sentences in each 5-word interval decreased, which resulted in underuse of the sentence length feature.

Our second method was an attempt to rectify this problem. We incorporated one single feature that describes the sentence length. This feature is ranked the highest in information gain and has twice as much information gain as the second best feature in our experiment with POS tags and keywords. When the rationale/non-rationale distribution was analyzed by WEKA, we could clearly observe that very long sentences rarely contained rationale. Although a slight decrease of 0.3 % in F-measure was observed for the base classifiers, stacking achieved 2.6 % increase in F-measure on numerical sentence length. This may be largely influenced by BayesNet, which had a 3.2 % higher F-measure when we used numerical values.

Impact of N-gram Length. Although using higher-order n-grams vastly increases the size of the feature set (from 8,495 for unigrams to 254,120 for 5-g), we hoped to capture more complex rationale-expressing structures. Also, the dimension problem could be mitigated using feature filtering to reduce the feature set size. We noticed consistent improvements over POS unigrams using 5-g, with an average F-measure increase of 8.68 %. However, 2-g used together with POS tags and sentence length decreased in F-measure for both BayesNet and SGD, but achieved a 3.33 % increase in F-measure for RandomForest.

Impact of N-gram Type. Using POS tags instead of word roots for *n*-grams significantly reduced the dimensions of the feature set, from 254,120 unfiltered features to 54,903. Furthermore, using POS tags did not significantly impact the classification accuracy (p-value = 0.7273 in a paired-sample *t*-test with 12° of freedom). Bayesian networks consistently produced the highest F-measures for these feature sets as individual learners in both the rationale-binary classification task and the argumentation-subset binary classification task, achieving 61.4 % (POS) and 60.4 % (roots), and 50.4 % (POS) and 50.2 % (roots) respectively.

Impact of POS and Verb Text. POS by itself did not perform well, with 46.9 % as its highest recorded F-measure for rationale-binary classification. This is likely due to it only having 46 attributes to learn over. Verb text generated somewhat better results than POS, with 60.5 % as its highest recorded F-measure for rationale-binary classification. This is likely due to rationale indication not always being directly associated with particular verbs. POS and verbs combined performed above average when compared to other methods, with 63.1 % as its highest recorded F-measure for rationale-binary classification. This likely indicates that both the text of verbs and the surrounding parts of speech complement each other when trying to indicate rationale.

Context (**neighbors**). By extracting verbs in surrounding sentences (2-Surrounding Sentences), we were able to achieve a higher F-measure than simply using verbs and POS tags on all experiments. However, the performance is significantly lower than the feature set with unigrams, POS, and sentence length. We have analyzed the annotations and it turns out that many rationale do not have verb phrases in them. In addition, GATE misclassified many verbs as pronouns when they appeared in the beginning of the sentence.

Our second attempt of using the contextual information by including adverbs, adjectives, conjunctions, determiners, verbs and pronouns gave us much better performance. We did not include all tokens in the surrounding sentences in order to reduce the number of attributes in the feature. In general, this feature set outperforms other feature sets we have constructed. Compared to the feature set that contains bag of words, POS, and sentence length only, the new feature set, including POS, sentences, and all words of the POS tags mentioned above achieved, on average, of 2.43 % higher F-measure.

In the experiment with POS tags, verbs surrounding sentences with maximum distance one in conjunction with sentence length, we achieved much better results than using verbs in the sentence and POS only. The new method yields as much as 15.9 % higher F-measure using BayesNet, 8.3 % higher F-measure using AdaBoost on SGD, and 2.2 % higher F-measure using RandomForest.

Summary of Results

We ran over 250 experiments during the work described in this paper. Here we summarize the best results obtained to date for each type of classification. These tables show the results with the best Recall (R), Precision (P), and F-1 measures from our experiments (with fewer rows if a feature set/learning algorithm was the best for two measures). All experiments shown here used information gain filtering. As mentioned earlier, we did not attempt to extract requirements, arguments, and assumptions as separate labels since there was too much subjectivity in their classification. Instead, we simply combined these annotations and treated them all as arguments.

Table 2 shows the best results we obtained for classifying the data as rationale/not rationale where the rationale included the questions, procedures, and answers as well as the argumentation.

Table 3 shows the best results we obtained for classifying the data as rationale/not rationale where the rationale consisted only of argumentation and did not include the questions, procedures, and answers. Tables 4, 5, and 6 show the best results for identifying decisions, alternatives, and arguments, where refers to arguments, requirements, and assumptions.

As noted earlier, since labels can overlap it would be necessary to build a separate classifier for each label when using single label learners as described above. Therefore, it would be possible to use a different learning algorithm for

Table 2 Best results for rationale-binary classification

Learning algorithm	Feature set	R	P	F-1
Stacking (PART, SGD, BayesNet)	VADCC2_PSn	**0.901**	0.509	0.65
Random forest	1WGPSn	0.442	**0.731**	0.553
Stacking (random forest, BayesNet, J48, SGD	VADCC2_PSn	0.878	0.551	**0.677**

Table 3 Best results for argumentation-subset binary classification

Learning algorithm	Feature set	R	P	F-1
Stacking (SGD, PART, BayesNet)	VADCC2_PSn	0.698	0.484	**0.569**
Naive bayes	VADCC2_PSn	**0.857**	0.209	0.335
Random forest	VADCC2_PSn	0.378	**0.882**	0.53

Table 4 Best results for decisions

Learning algorithm	Feature set	R	P	F-1
BayesNet	VADCC2_PSn	**0.842**	0.514	0.638
Random forest	VADCC2_PSn	0.719	**0.889**	**0.795**

Table 5 Best results for alternatives

Learning algorithm	Feature set	R	P	F-1
BayesNet	VADCC2_PSn	**0.81**	0.131	0.226
Random forest	VADCC2_PSn	0.142	**0.794**	0.241
SGD	VADCC2_PSn	0.429	0.31	**0.36**

Table 6 Best results for arguments-all

Learning algorithm	Feature set	R	P	F-1
BayesNet	VADCC2_PSn	**0.823**	0.208	0.332
Random forest	VADCC2_PSn	0.054	**0.607**	0.099
Stacking (BayesNet, SGD w/AdaBoost, Random forest)	V	0.273	0.537	**0.362**

each. Detailed results for questions, answers, and procedures are not listed because their presence in boilerplate suggests that if we are interested in obtaining that information it would be more effective to do so using methods other than machine learning. Experiments obtaining questions, answers and procedures were only run using VADCC2_PSn feature set. The best F-1 measure for questions was 0.668, for answers it was 0.549, and for procedures it was 0.386. These results all used J48 (a decision-tree approach) as the classifier.

Assessment of Results

The classification results can be assessed by three criteria—how well does automatic classification do compared to human annotators, how well do our techniques perform compared to those used by others, and are the results fit for purpose.

Since the annotation was performed in groups of ten bug reports with annotation tasks rotating among the four researchers, the inter-annotator agreement was computed by calculating the precision, recall, and F1-measure between pairs of annotators for each set of ten bug reports. These values were calculated leniently, considering any overlap between two annotations to be a match. Table 7 compares the best results from classification with the average F1-measures among the bug report sets for all three of our classification goals. Precision, recall, and F-1 measure were chosen for comparison rather than the Kappa measure because part of the annotation task included selecting the span of text, which could vary between annotators and some consider the Kappa to not be an appropriate measure in this case [9].

Table 7 shows that the system accuracy was very close to the agreement of human annotators for the binary classification task. In identification of arguments, system performance was only slightly worse than inter-annotator agreement. The greater difference between system performance and annotator agreement in locating alternatives suggests that identifying alternatives is a harder task than identifying arguments. The high system performance for identifying decisions is explained by the consistent and almost sole location of decisions as the title of each bug report. The comparatively low inter-annotator agreement for decisions resulted

Table 7 Best classifier and inter-annotator agreement

Classification goal	Best classified	Annotator average	Annotator median	Annotator StDev
All rationale	0.677	0.694	0.69	0.10
Argumentation subset	0.569	0.461	0.495	0.17
Decisions	0.795	0.32	0	0.407
Alternatives	0.36	0.516	0.515	0.151
Arguments (all)	0.362	0.376	0.375	0.185

from a missed instruction to classify the bug report titles as decisions. These errors were corrected in the adjudication phase of annotation.

The agreement values also set a relatively low upper bound for system performance compared to other information extraction tasks in more general domains. They suggest that annotation of rationale, at least in the bug reports domain, is not an easy task.

The related work section of this paper describes results obtained by others. It is not possible to do a direct comparison since the datasets, classification goals, and in some case the measurements made are not consistent.

The most critical question is if the rationale extracted is fit for purpose. The ultimate goal is to import the documents with their classified rationale into a Rationale Management System. We were able to improve on results from earlier work at classifying text as rationale or not rationale. The results for classifying rationale into argumentation elements (decision, alternative, and argument) are not as strong, with F-1 measures under 0.4 for alternatives and arguments. Some techniques do give good precision (around 0.8) but with low recall. It is possible, however, to argue that extracting some rationale is better than none. The extracted rationale would need some manual corrections to be usable. We plan to import the documents and rationale into an existing DR-Wiki system built on the Drupal content management platform. Future work will study the speed and accuracy of structuring rationale after it has been classified by machine versus when it is done without assistance.

Related Work

While there is significant prior work in the capture and use of rationale in fields such as Human Computer Interaction [10], Engineering Design [11], and Software Engineering [12, 13], there have been only a few attempts to automatically extract rationale from existing documentation. An exception to this is the work of Liang et al. [14]. Their work focuses on learning design rationale from patent documents. They use a three-layer model of rationale that captures issues, design solutions, and artifacts. The learning process follows three steps: first, they use PageRank [15] on frequently appearing words to identify artifacts. They then perform issue summarization by defining issue language patterns (phrases that contain motivational meanings) and use those as part of a manifold ranking process that assumes that sentences with issue language patterns have a higher score. The last step looks for solution and reason pairs by using reason language patterns to identify candidate reason sentences, which are then correlated with the remaining sentences to detect the reason-solution pairs. They achieved an 18.5 % F-value for artifact extraction, a 51.95 % ROUGE-1 (Recall-Oriented Understudy for Gisting Evaluation) F-value for issue summarization, and a 56.15 % ROUGE-1 F-value on solution and reason identification.

The TREx (Toeska Rationale Extraction) approach [16] used extraction tools built by domain experts to perform information extraction of “knowledge units,” some of which correspond to rationale. This tool was based on GATE and used manually created extraction rules to identify properties defined in architecture and rationale ontologies. The aggregated extraction results achieved an F-1 value of 50 % when analyzing 26 pages of architecture documents.

Mochales et al. [17] analyzed documents to detect arguments. Their approach used n-grams and keywords as well as linguistic features (specifically modal auxiliaries, adverbs, and verb tense). For legal texts, they reported an 80 % F1-Score for identifying arguments. They also applied their methods to a corpus of sentences where 50 % of the sentences contained arguments. For that corpus they reported a 73 % F1-Score. In comparison, less than 11 % of the sentences in our dataset contained rationale.

Rogers et al. [5] investigated techniques to extract rationale from bug reports. The work described in this paper differs from [5] in several ways. First, we annotated bug reports by rationale type, not by rationale/not rationale. The pre-processing described in [5] was much more limited than described here, with a very different set of classification features. Rogers et al. [5] performed two distinct sets of experiments, one set that used WEKA and one that only used GATE.

The WEKA experiments used two different ontologies to provide features—one that included generic arguments and one that used security terminology. The WEKA experiments determined that pre-processing, such as stemming and stop-word removal, improved performance and that the generic argument ontology performed better than the domain-specific one (with the best results coming from combining the two). Using WordNet to expand the list of ontology terms gave improved results. Even with the ontology pre-processing and expansion, the best F-1 measure achieved with WEKA was 59.7 %. This was for classifying text as rationale or not rationale for all the rationale types (this is lower than the 67.7 % presented in this paper).

The GATE experiments used LibSVM to identify rationale using three different linguistic features: modal auxiliaries, adverbial clauses, and projective clauses. Adding additional features increased recall but decreased precision. The best F-1 measure achieved with linguistic features was 33.6 %. This was also for classifying text as rationale or not rationale.

Conclusions and Future Work

Our eventual goal is to use our classification tools to identify rationale and to then import the rationale and its source documents into a Rationale Management System for use by future software developers and maintainers. We plan to investigate a number of research questions as we continue this work.

Is it easier to partially structure rationale rather than create it from scratch? We plan to use the DR-Wiki tool to compare the time and effort required to adjust the structure of partially identified rationale versus structuring as a completely manual process.

Are there features or feature combinations that result in better performance? We plan to continue experiments with different features and feature combinations to improve classification performance using our GATE-WEKA pipeline.

Can annotation guidelines be refined to create training/test data with less subjectivity? We will be refining our annotation guidelines and training. The annotation accuracy forms the upper bound for system performance. Using a third annotator to adjudicate these differences was intended to catch errors resulting from inconsistent interpretation of instructions but it is possible that with more detailed annotation instructions, annotator agreement, and thus system performance, would increase.

How do classification results compare when different data sets are used? **Can the same classifier be used for different input data or will a new model need to be created for each**? We also plan to work with additional datasets to compare our ability to classify rationale on documents with different authors and audiences. We anticipate that a higher density of rationale may lead to better classification results. Bug reports are only one possible source of rationale and further research is needed to determine what sources will be the most likely to yield rationale that will be useful to future software developers. No single source is likely to provide a complete picture of system rationale. We will be conducting experiments to determine if classifiers can be re-used on document types not used in training to determine when re-use is practical and when new classifiers must be built. We will do this using similar data (such as using classifiers created on Chrome bug reports to classify Firefox bug reports) and on data combing from different sources (such as design session transcripts).

Rationale extraction and structuring is a challenging and important problem and requires further work to determine if it is possible to achieve accuracy at a level that will make the rationale useful. The work described in this paper provides a mechanism for investigating the many research questions that must be tackled to address the rationale capture problem that has been the major obstacle in providing this valuable information for software developers and maintainers. The process followed for text mining by combining GATE and WEKA can also be applied to other text mining tasks.

Acknowledgements This work was supported by NSF CAREER Award CCF-0844638 (Burge). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

References

1. Burge J, Brown DC (2008) Software engineering using rationale. J Syst Softw 81(3):395–413
2. Buckingham Shum SJ, Selvin AM, Sierhuis M, Conklin J, Haley CB, Nuseibeh B (2006) Hypermedia support for argumentation-based rationale: 15 years on from gIBIS and QOC. In: Dutoit AH, McCall R, Mistrik I, Paech B (eds) Rationale management in software engineering. Springer, Heidelberg, pp 111–132

3. Cunningham H, Maynard D, Bontcheva K (2011) Text processing with GATE (Version 6). University of Sheffield Department of Computer Science. 15 April 2011. ISBN 0956599311
4. Hall M, Frank E, Holmes G, Pfahringer B, Reutmann P, Witten IH (2009) The WEKA data mining software: an update. SIGKDD Explor 11(1):10–18
5. Rogers B, Gung J, Qaio Y, Burge JE (2012) Exploring techniques for rationale extraction from existing documents, In: Proc. International Conference on Software Engineering, IEEE Press, pp 1313–1316
6. Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm, Proc international conference on machine learning. Morgan Kaufmann, San Francisco, pp 148–156
7. Wolpert DH (1992) Stacked generalization. Neural Netw 5:241–259
8. Ting KM, Witten IH (1997) Stacked generalization: when does it work?. In: Proc. Int. Joint Conference on Artificial Intelligence. pp 866–871
9. Hripsak G, Rothschild AS (2005) Agreement, the F-measure, and reliability in information retrieval. J Am Med Inform Assoc 12(3):296–298
10. Moran TP, Carroll JM (eds) (1996) Design rationale: concepts, techniques, and use. L. Erlbaum, Hillsdale
11. Lee J (1997) Design rationale systems: understanding the issues. IEEE Expert: Intell Syst Appl 12(3):78–85
12. Dutoit H, McCall R, Mistrik I, Paech B (eds) (2006) Rationale management in software engineering. Springer, New York
13. Burge JE, Carroll JM, McCall R, Mistrik I (2008) Rationale-based software engineering. Springer, Berlin
14. Liang Y, Liu Y, Kwong C, Lee W (2012) Learning the 'Whys': discovering design rationale using text mining – an algorithm perspective. Comput Aided Des 44(10):916–930
15. Brin S, Page L (1998) The anatomy of a large-scale hypertextual web search engine. Comput Netw ISDN Syst 30:107–117
16. López C, Codocedo V, Astudillo H, Cysneiros LM (2012) Bridging the gap between software architecture rationale formalisms and actual architecture documents: an ontology-driven approach. Sci Comput Program 77(1):66–80
17. Mochales P, Moens MF (2009) Argumentation mining: the detection, classification and structure of arguments in text, ICAIL-2009

Learning from Product Users, a Sentiment Rating Algorithm

Dilip Raghupathi, Bernard Yannou, Roain Farel, and Emilie Poirson

Abstract Social media gives new opportunities in customer survey and market survey for design inspiration with comments posted online by users spontaneously, in an oral-near language, and almost free of biases. This new source however has huge size and complexity of data needed to be processed. In this paper, we propose an automated way for processing these comments, using sentiment rating algorithm. Traps like negations, irony, smileys are considered in our algorithm. We validate it on the example of a commercial home theatre system, comparing our automated sentiment predictions with the one of a group of 15 test subjects, resulting in a satisfactory correlation.

Introduction

Product designers always welcome feedbacks for the sake of design improvement. Spontaneous comments on new products posted by users or customers in the internet are an incredible source of unbiased information. They are testimonies of individual experiences with product usage and/or satisfaction levels. Unbiased feedback has been proven to be unexpectedly hard to obtain. But, spontaneous customer comments on new products remain a valuable source for feedback on design. Resulted data from interviews, questionnaire, surveys and other similar methods suffer from the influence of the test situation [1]. With the rise of Social media, people express themselves without any influence of fear, pressure, intimidation or incentives while giving their opinion. These new media become the centre of attention for analytical purposes, both for industrial and academic research, design analytics for example [2].

D. Raghupathi (✉) • B. Yannou • R. Farel
Ecole Centrale Paris, Châtenay-Malabry, France
e-mail: dilip.raghupathi@student.ecp.fr

E. Poirson
Ecole Centrale de Nantes, Nantes, France

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_27

A lot of event specific sentiment analyses have been carried out like stock market trends [3]. Real-time geo-localized tweet analysis has shown to develop efficient and inexpensive applications. For example, they have been effectively used to adapt the emergency situations in the wake of natural disasters [4]. In the same way, an epidemic can be detected based on a certain tweet trend [5]. For a designer however, the use of the customer feedback extracted from the internet is still limited. The other particularity of online product reviews is that the product user motive is either to help others buy the product or make sure no one buys the product in future. So a major part of the review would talk about the salient features of a product linked to its method of usage. Analysing such micro blogs or product reviews carefully may provide a lot of details as to how people uses it, in which scenarios and whether he is satisfied and happy about its usage values and features.

In this paper, we bring our focus to a methodology for product review analysis using a sentiment rating algorithm. The following section reviews the body of the literature on the user data analysis and the Natural language processing (NLP) method. Section "Methodology" explains our proposed framework: the SENTiment Rating ALgorithm (SENTRAL) which is used to rate the user reviews, isolate the usage scenarios, sacrifices and sarcasm into individual entities. Section "Case Demonstration: Reviewing a Home Theatre" applies the proposed method on a case study, illustrating the use of SENTRAL on a commercial product. Section "Validation" goes through the validation procedure where the ratings obtained from our system are compared with those obtained from humans, before concluding in section "Conclusion".

Literature Review

Online Customers' Data Analysis

Understanding the customer is a crucial issue for product design. The difficulty of capturing the voice of the customer orally in person can now be compensated with the opinions that customers leave on internet. The analysis of opinions aims to provide professionals and developers with an overview of the customer experience and ideas that provide clues or evidence for designers to better interpret the voice of the customer [6]. The first interest is to enrich the customer database, very useful in Customer Relationship Management for example [7]. The first domain using online reviews is the marketing to find the strategic goals and identify the customers [8] and customer service [9]. Increasingly, the design sector employs the weblogs and product review to target relevant information for designer [10] and [11]. To analyze these online reviews, computer tools like the General Inquirer [12] are essential. Iker [13] proposes a method attempting to reduce the choice "a priori" word classes. After a phase of cutting and cleaning (determiners, prepositions . . .), the synonymous words are gathered. Sometimes when designers use search engines, they find themselves stuck with a lack of keywords to search. A tool called Tweetspiration

[14] was created to provide designers alternative search paths and recommendations from recent twitter trend. Occurrences of the remaining words are calculated and presented as a matrix of correlation between each other. These interactions help to keep the meaning of the text underlining the main topics. In linguistics, POS tagging (Parts-Of-Speech) is the process of marking up a word in a text as corresponding to a particular part of speech based on its definition and context using a software tool [15]. Syntactic analysis can then be used to determine the combinations of words. It may be noticed that in all cases, the structure is similar: (1) Data retrieval and preparation (2) Text processing (3) Analysis.

The freedom given to the online reviewers allows them to express some feelings and sentiments. Particularly on twitter, it is believed that sentiment in public media plays a big role in the decision making process of the end users [6] and [16], and hence collective sentiment in social media may influence consumer preferences and impact buying decision.

Natural Language Processing (NLP)

Textual information in the world can be broadly categorized into two main types: *facts* and *opinions*. Facts are objective expressions about entities, events and their properties. Opinions are usually subjective expressions that describe people's sentiments, appraisals or feelings toward entities, events and their properties [3]. Liu [6] created a model to classify data as subjective and objective. Sentiment analysis, the process of extracting the feelings expressed in a text, is considered as one of the methods of Natural Language Processing (NLP). This is an area of research that involves the use of computers to analyse and manipulate natural language with minimum human intervention for interpretation. In order to construct a program that understands human language, three main bases are required [17]. Thought Process, Linguistic representation, World Knowledge.

NLP is carried out in parts starting from word level to understand the Parts of Speech, then to sentence level in order to understand the word order and meaning of the sentence and then the entire text as whole to lift the underlying context.

Chowdary [18] explained that language is understood in seven interdependent levels by humans and must be integrated in computer programs to replicate it. They are: (1) Phonetic level (2) Morphological level (3) Lexical level (4) Syntactic level (5) Semantic level (6) Discourse level and Pragmatic level. Phonetics deals with the pronunciation, the smallest parts of a word like suffixes and prefixes are related to the morphology. Lexical level is the parts of speech and syntactic level deals with the structure of the sentence and the order of the words. Meanings of word and sentences are understood at the Semantic level where as knowledge exterior to the document is classified in the pragmatic level. Our system involves four of the seven levels; Morphological, lexical, syntactic and semantic level. Several works had to be studied in order to understand these methodologies.

Though tweets are used for diverse reasons and the context of each tweet is different, they can primarily be grouped into two categories. One category shares personal issues while the other spreads information and creates awareness among the online community [19]. A number of biases are possible while conducting an opinion survey. The most prominent of them all is called the Bradley effect in which the responders are unwilling to provide accurate answers, when they feel such answers may reflect unpopular attitudes or opinions [20]. To overcome this effect, automated polling approaches, known as opinion mining were introduced. These automated polling approaches overcome most of these biases naturally. It was extended to sentiment analysis by Bollen et al. [21] using POMS (Profile of Mood States) and Hu et al. [22] using POS (Parts of Speech).

Methodology

We developed a methodology to analyse the online user review on products, looking forward to deal with the following challenges:

1. Indicates features a customer is not pleased about
2. Indicates features a customer is pleased about
3. Outlines the overall satisfaction/dissatisfaction
4. Provides keywords of appreciation
5. Provides keywords of criticism
6. Evaluate the modes of usage as described by the customer
7. Detects possibility of sarcasm

The proposed methodology is depicted in Fig. 1. The first step is the extraction of data from website, in step 2, pre-processing, we carry out the reduction of the noise, classification of words with the aid of Perl script API and Stanford CoreNLP tokenizer. In the third step of Text processing, the noise free data is organised as a tree of dependency from the dependency list obtained with the aid of Stanford Parser and Probabilistic Context Free Grammar (PCFG). Now the text is prepared completely for extraction of sentiment: locally with the aid of DAL (Dictionary of Affect Language) and globally with our SENTRAL algorithm. Along with the

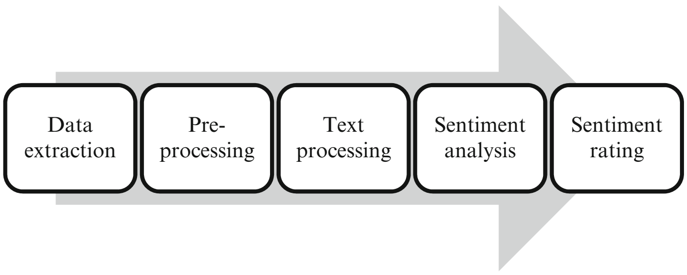


Fig. 1 Process flow chart

sentiments, using the tree of dependency, the modes of usage is also isolated using our algorithm. Each step is described in the following sections.

Extraction of Data from Website and Pre-processing

Data Crawling

Three websites are selected to obtain data: Twitter, Amazon and Flipkart. The main reason is the publicly of their data, available with Perl script API's. Basically two types of data are obtained: Tweets and User review data. A tweet is a microblog, as shown in Fig. 2, limited to 140 characters, containing normal text in addition to targets denoted with a "@" symbol, hash tags (#) to group words from different tweets and smileys (emoticons). Another place to express feelings is a product review on commercial websites without character constraint (example hereafter).

Since the maximum number of characters in a tweet is 140, they have a lot of constraints to deal with. This constraint becomes an advantage for textual analysis because the user has no place to ramble, thus expresses quickly and directly his feelings. A tweet consists of the combination of entities: the content, Hash tags, URLS, targets, acronyms and emoticons. Since there is a character constraint, users tends to use a lot of acronyms like "lol" which means "Laugh Out Loud", short forms like "bcoz" in place of "because" and alpha numeric short forms like "p6" instead of "physics" etc. Certain tweets are targeted at specific users and are denoted with the "@" symbol followed by their name. Hash tags are used to group data based on certain user defined topics. They are denoted by "#" followed by the word. URL are provided by certain users to mark references or proofs. Twitter automatically shortens these links to 20 character phrases to minimize character usage. We created thus an acronym/symbolic dictionary from an online resource that contains meaning for all these commonly used acronyms and action of the symbols used.

Unlike tweets, there is no restriction to the size of a product review. The data are extracted with Perl script API from amazon.com and flipkart.com. A user review consists of the following information: the date of the review, the number of stars or rating in a scale of 0–5, the location of the user, the content of the review and also a

@jcdave The iPhone 5 is a waste of money, you end up paying 200 grand more than any other phone with same features ☹ #apple #dis-appointed	The new sound box by #Bose is an absolute marvel. Crystal clear sound :D I am so happy I decided to invest in this system ☺

Fig. 2 Example of tweets that review a product

count of the number users agreeing with the review to eliminate plagiarism and misleading customers.

Data Pre-processing

As our objective is to find out the sentiments and usage objectives of the customer, there is a lot of noise in the data that is crawled and hence needs to be filtered before it is taken forward in the process. This step is a filtration of the text extracted: each word is categorized thanks to an original list of acronyms (Stanford CoreNLP tokenizer [23]). For example, NNP is a singular proper noun, VB is a verb on its basic form, PRP a personal pronoun, RB an adverb. All standard acronyms are expanded using this list and the ones not found in the dictionary are ignored and removed from the sentence. All URLs are removed as they do not help the performance of the system in any way.

The example below illustrates the data pre-processing for the sentence “This product is very good” where one can find a descriptive determiner (ND), a common name (NN), a verb VB2, an adverb RB and an adjective JJ.

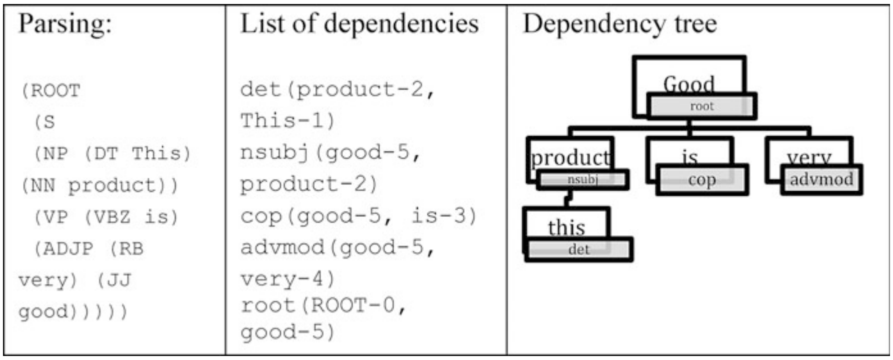
```
Before: This product is very good http://tinyurl.com/n2hboap
After: This/ND product/NN is/VB2 very/RB good/JJ
```

Text Processing

Parsing and Creation of Dependency Trees

Parsing is the process of breaking down the sentences to words and finding out the grammatical relations between these words. Probabilistic Context Free Grammar (PCFG) is based on the study of language gained from hand-parsed sentences to try to produce the most likely analysis of new sentences. A list of dependencies is obtained and a tree is created. This model proposes 55 kinds of possible grammatical dependencies between words in the English language. A standard dependency is written as: Relation (governor, dependent). For instance, for the sentence “This product is very good”, “This” associated to “product” is a nominal group (NP). “is” is the verbal group (VP) and “very” and “good” is a qualificative group (ADJP). We define grammatical relations defined in a hierarchy so as to arrive at the intended meaning. Using the dependency list and the hierarchy, we are able to create the dependency. The result of the parsing, dependencies and tree is given Fig. 3.

We want to focus on the feelings and the modes of usages expressed by the user. The full list of relations is reduced for us to acomp (adjective complement), advmod (adverbial modifier), amod (adjectival modifier), neg (negation modifier), aux (auxiliary) and mod (modifier). These are the ones that allow the expression of opinion and physical activities as demonstrated in the sections that follow.



Parsing:

```
(ROOT
 (S
 (NP (DT This)
(NN product))
 (VP (VBZ is)
 (ADJP (RB
very) (JJ
good)))))
```

List of dependencies

```
det(product-2,
This-1)
nsubj(good-5,
product-2)
cop(good-5, is-3)
advmod(good-5,
very-4)
root(ROOT-0,
good-5)
```

Fig. 3 The stages of text processing

Table 1 Example of the pleasantness rating of words in the dictionary of affect language

Word	DAL score
Money	0.8889
Phone	0.4375
Waste	0.0000
Marvel	1.0000
Happy	1.0000
Investment	0.7222

Extraction and Analysis of the Sentiments

Local Sentiment Analysis with DAL

In the dependency list, the relations are binary in nature. To carry out the process of finding the sentiment rating, we propose the SENTRAL algorithm that uses the Dictionary of Affect Language (DAL). The DAL [24] scores each of the 200,000 English words based on the pleasantness it evokes in the human mind. It is on a scale of 1–3 where 1 means the most unpleasant and 3 means the most pleasant. We normalize this score on a scale of 0–1 to suit out algorithm. Table 1 presents some words of tweet with their DAL score. For adjectives, the scores from the DAL can be directly assigned. The meaning of the adjective will change based on the presence of a modifier before or after it. For example, the word “good” and the word-cell “very good” evoke different levels of appreciation.

There are basically two types of emotions; good and bad. The emotional guidance system [25] of humans indicates that a person is happy and satisfied if he is in alignment with his requirements. After the dependency tree is created, the words with the tags of *advmod* and *amod* are assigned the pleasantness score by comparing it with the DAL.

Global Sentiment Rating with Our SENTRAL Algorithm

We finally choose a 0–5 scale to globally rate the sentiment of the reviews through our SENTRAL algorithm in order to further compare with customer reviews which are most of the time appraised on such a scale.

The SENTRAL algorithm uses the dependency tree, traversing from the last leaf till the root by progressively evaluating the grammatical relations encountered.

Each time a dependency relation is considered two words are compared: S_{governor} and $S_{\text{dependent}}$ which have their respective DAL scores. For the dependency relation advmod (adverbial modifier), we propose the specific sentiment rating algorithm of Fig. 4. We are influenced by the thresholds of segmentation of 0.55 and 0.4 obtained from the guidelines of DAL. An illustration of this algorithm is shown in Fig. 5.

The second step is to check if the ROOT word's POS tag is JJ (adjective) or adverb and the DAL scores are assigned directly. If no such tags are found, it means no sentiment has been expressed and the sentence is ignored.

After this process we have the separate scores of all the related words, sentences and the paragraph. The score of the jth sentence is given by Eq. 1.

$$Sentence_j = \frac{\sum Dependency\ \ tag_{ij}}{i} \tag{1}$$

where "dependency tag_{ij}" denotes the score of the i^{th} tag in sentence j.

```
If ($S_{governor} \geq 0.55$ and $S_{dependent} \geq 0.4$)
        { if $S_{governor} \leq S_{dependent}$
                $S_{tag} = \frac{S_{governor} + S_{dependent}}{2} * 5$
          else $S_{tag} = S_{governor} + S_{governor} * S_{dependent} * 5$ }

If ($S_{governor} \geq 0.55$ and $S_{dependent} \leq 0.4$)
{ $S_{tag} = S_{governor} - S_{governor} * S_{dependent} * 5$ }

If ($S_{governor} \leq 0.55$ and $S_{dependent} \geq 0.4$)
        { $S_{tag} = S_{governor} - S_{governor} * S_{dependent} * 5$ }

If ($S_{governor} \leq 0.55$ and $S_{dependent} \leq 0.4$)
        { if $S_{governor} \geq S_{dependent}$
                $S_{tag} = \frac{S_{governor} + S_{dependent}}{2} * 5$
          else $S_{tag} = [1 - S_{governor} - S_{governor} * S_{dependent}] * 5$ }
```

Fig. 4 Sentiment rating algorithm for adverbial modifier relation *relation (governor, dependent)*

```
nsubj(easy-4, It-1)
cop(easy-4, is-2)
```
`advmod(easy-4, very-3)`
```
root(ROOT-0, easy-4)
aux(use-6, to-5)
xcomp(easy-4, use-6)
det(control-9, the-7)
amod(control-9, remote-8)
dobj(use-6, control-9)
```

`advmod(easy-4, very-3)`
`(0.6665 , 0.4165)`

$S_{tag} = [S_{governor} + (S_{governor} * S_{dependent})] * 5$

$S_{tag} = [0.6665 + (0.6665*0.4165)]*5$

$S_{tag} = [0.6665 + 0.2775]*5 = 4.72$

Fig. 5 Example of sentiment rating for an adverbial modifier relation

Table 2 Sentiment score legend

Scores	Conclusion
$0 \leq S_{review} < 2$	Sad and unsatisfied
$2 \leq S_{review} < 3$	Indifferent, happy to use with sacrifices
$3 \leq S_{review} \leq 5$	Happy and satisfied

The score of the entire text is given by Eq. 2.

$$Sentiment\ \ score = \frac{\sum Sentence_j}{j} \tag{2}$$

The words that do not figure in the DAL are ignored since almost all words in the WordNet [26] dictionary are found in this and the probability of a common word missing is very weak. All nouns that have an adjective close to it are grouped together. Negations words like 'not' 'cannot' 'shouldn't' are dealt in such a way that the scores are inverted for the words. For the non English words, the list of words not found even in the WordNet dictionary is given, with a neutral value of 0.5.

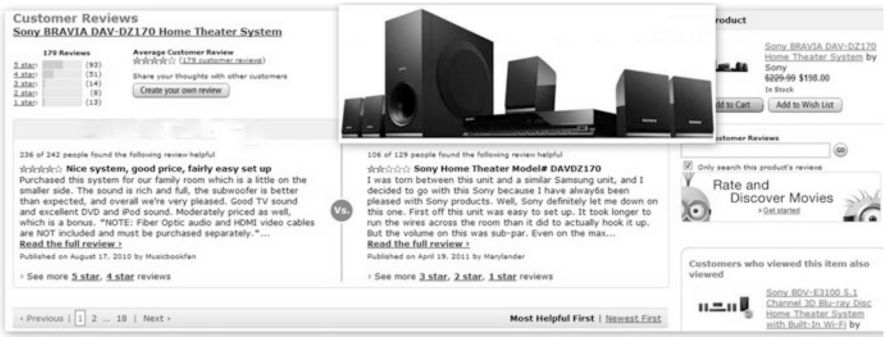
Finally, once the score of a sentence calculated, one can consider that the feeling of the customer is approximately given by Table 2.

Isolation of Modes of Usage from the Reviews

The model looks for the tags *"aux"* and *"auxpass"* to find out the usage functions as described by the user. The *"aux"* and *"auxpass"* are defined as an auxiliary of a clause is a non-main verb of the clause, e.g., a modal auxiliary, or a form of "be", "do" or "have" in a periphrastic tense [27]. Hence this tag is used by our algorithm to obtain all kinds of physical actions that are being expressed in a text.

```
Sentence: I have to turn the knob everytime

nsubj(have-2, I-1)
root(ROOT-0, have-2)
aux(turn-4, to-3) - - - > Mode of usage "to turn"
xcomp(have-2, turn-4)
det(everytime-7, the-5)
amod(everytime-7, knob-6)
dobj(turn-4, everytime-7)
```



Customer Reviews
Sony BRAVIA DAV-DZ170 Home Theater System

179 Reviews
5 star: (93)
4 star: (51)
3 star: (14)
2 star: (8)
1 star: (13)

Average Customer Review
(179 customer reviews)
Share your thoughts with other customers

236 of 242 people found the following review helpful
Nice system, good price, fairly easy set up
Purchased this system for our family room which is a little on the smaller side. The sound is rich and full, the subwoofer is better than expected, and overall we're very pleased. Good TV sound and excellent DVD and iPod sound. Moderately priced as well, which is a bonus. "NOTE: Fiber Optic audio and HDMI video cables are NOT included and must be purchased separately."...
Read the full review ›
Published on August 17, 2010 by Musicbookfan
› See more 5 star, 4 star reviews

106 of 129 people found the following review helpful
Sony Home Theater Model# DAVDZ170
I was torn between this unit and a similar Samsung unit, and I decided to go with this Sony because I have alway6s been pleased with Sony products. Well, Sony definitely let me down on this one. First off this unit was easy to set up. It took longer to run the wires across the room than it did to actually hook it up. But the volume on this was sub-par. Even on the max...
Read the full review ›
Published on April 19, 2011 by Marylander
› See more 3 star, 2 star, 1 star reviews

‹ Previous | 1 2 ... 18 | Next ›
Most Helpful First | Newest First

roduct
Sony BRAVIA DAV-DZ170 Home Theater System by Sony
~~$229.99~~ $198.00
In Stock

stomer Reviews
Only search this product's reviews

Customers who viewed this item also viewed
Sony BDV-E3100 5.1 Channel 3D Blu-ray Disc Home Theater System with Built-In Wi-Fi by

Fig. 6 Reviews of products on Amazon

Case Demonstration: Reviewing a Home Theatre

In this section we use the methodology proposed in the previous section to analyse the users review on a commercial home, shown in Fig. 6.

In order to demonstrate the SENTRAL sentiment rating algorithm, a general usage product has been selected from an online product provider with an active feedback forum, in form of text and an overall note from 0 to 5. The selected product is a home theatre system (see Fig. 6). Fifteen reviews (from different reviewers) are crawled from the feedback forum website (see for instance Fig. 7). Here is how the methodology is applied.

Step 1
Extraction of data from website and pre-processing. The 15 comments are extracted and sequenced by sentences. Let us take the example of: "It took longer to run the wires across the room than it did to actually hook it up".

Step 2
Text processing (organised as tree of dependency). The Stanford Parser is used to establish the dependencies network. For the line "It took longer to run the wires

Amazon Product Code: B003B8VBJ2
Product name: **Sony BRAVIA DAV-DZ170 Home Theatre System (Electronics)**

Review: Well, Sony definitely let me down on this one. First off this unit was easy to set up. It took longer to run the wires across the room than it did to actually hook it up. But the volume on this was sub-par. Even on the max level volume (35) it still wasn't that loud. The main problem was the amount of bass that it produces. The bass is so overpowering that you can barely even hear people talking in the movie, and there is no way to adjust the levels at all.

Fig. 7 Sampled review

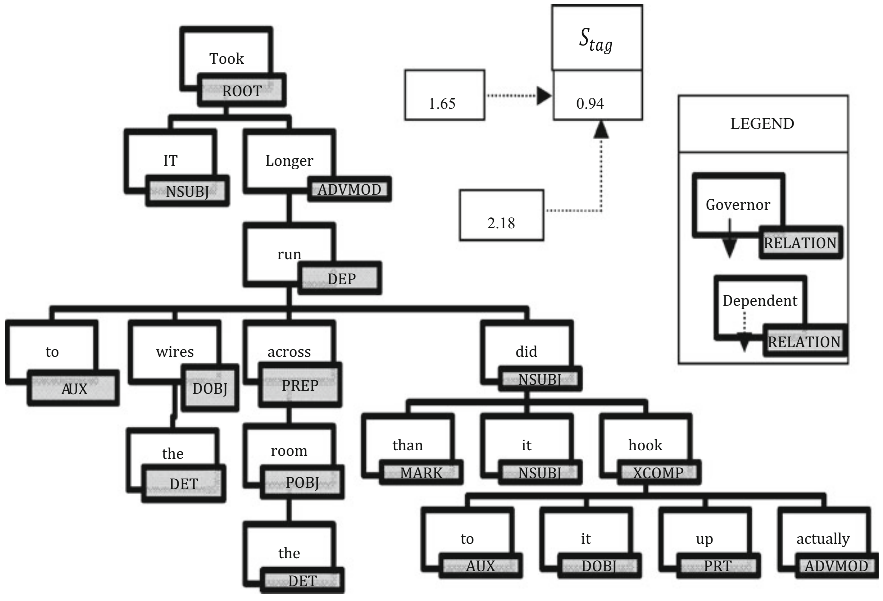


Fig. 8 Dependency tree of a sentence for a technical review on the home theatre system

across the room than it did to actually hook it up" It gives: "It/**PRP** took/**VBD** longer/**RB** to/**TO** run/**VB** the/**DT** wires/**NNS** across/**IN** the/**DT** room/**NN** than/**IN** it/**PRP** did/**VBD** to/**TO** actually/**RB** hook/**VB** it/**PRP** up/**RP** ./.". Using this, we obtain the dependency list from the parser again that arranges words in such a way that all grammatical relationships are established between the words. Following this step, we are able to create the dependency tree as shown in Fig. 8.

Step 3
Extraction and analysis of the sentiments in the message. SENTRAL identifies the following tags and assigns them the DAL score and calculates the score of the individual tags as Stag.

Table 3 Sentence-wise scores in the review

Sentence	Score
Well, Sony definitely let me down on this one	1.016
First off this unit was easy to set up	2.325
It took longer to run the wires across the room than it did to actually hook it up	1.39
But the volume on this was sub-par	N/A
Even on the max level volume (35) it still wasn't that loud	1.8052
The main problem was the amount of bass that it produces	N/A
The bass is so overpowering that you can barely even hear people talking in the movie, and there is no way to adjust the levels at all	1.0675

$$advmod(took-2,\ \ longer-3):\ \ advmod(0.33, 0.4375)\ \ S_{tag} = 0.94$$
$$advmod(hook-16,\ \ actually-15):\ \ advmod(0.55, 0.33)\ \ S_{tag} = 1.84$$

The same procedure is carried out for all sentences iteratively (see scores in Table 3) and the score is obtained for the review as whole using Eq. 3.

$$S_{review} = \frac{\sum S_{Sentences}}{Number\ \ of\ \ valid\ \ sentences} = \frac{1.016 + 2.325 + 1.39 + 1.8052 + 1.0675}{5} = 1.52074 \tag{3}$$

The total score of emotion found by our algorithm is then 1.52 on a scale of 5.

Step 4
Now to obtain the terms related to usage: SENTRAL localizes the tags "*aux*" and "*auxpass*" from this target text and isolates them. As a result, the following tags are obtained.

aux(run-5, to-4).
aux(hook-16, to-14)

Hence the two modes of usage expressed by the user in this particular line of the review are *"to run"* and *"to hook"*. Insights like these can help designers isolate the scenarios of usages and hence improve experience related to specific parts of the product.

Validation

The model that we propose basically replaces the human function of understanding and interpreting a text. We propose to validate our model by asking humans to do exactly the same task that our model performs rate reviews on a scale of 0–5. For

Table 4 Results from the online questionnaire

Rating/review	1	2	3	4	5
1	2	0	17	19	0
2	15	18	2	3	0
3	0	0	2	6	30
4	0	2	4	15	17
5	0	3	19	15	1
6	1	1	13	19	4
7	3	11	10	11	3
8	17	13	5	3	0
9	0	6	10	18	4
10	3	15	14	6	0
11	2	3	18	15	0
12	0	1	6	17	14
13	0	1	14	21	2
14	0	1	9	15	13
15	17	9	4	7	1

this, a poll was conducted online. A form containing all the 15 reviews was made public, people were asked to read all the reviews and rate them on this scale based on what their mind evokes about the satisfaction. The question was the following: "This questionnaire contains reviews about a Home Theatre system written by different users. After reading, please rate these reviews on a scale of 0–5 based on what you feel is the satisfaction level of each of these users. I request your kind patience and help me with my thesis. Thanks a lot in advance :)". The results obtained from the poll are summarized in Table 4.

In this table, each column denotes the number of persons who have voted for that particular rating, 1 being the least satisfied and 5 being the most satisfied based on their inference after reading the reviews. The scores being well divided (unimodal repartition), the mean is calculated and given in Table 5. The weighted average is then compared with the score obtained from our model in Table 5 to find out the error (difference).

This error is rather weak (see Tables 5 and 6, and Fig. 9) since the average of errors is 1.3 % (over five points) and the average of absolute error values is 6.42 %.

Human-computer interaction research often involves experiments with human participants to test one or more hypotheses. We use ANOVA (Table 7) to test the hypothesis of whether the difference between results obtained from SENTRAL and the online poll to rate the sentiments (Table 5 column 2 and 3) are significant (H1) or not (H0).

The ANOVA result is reported as an F-statistic and its associated degrees of freedom and p-value. The individual means for SENTRAL and Human rating were 3.29 and 3.22 respectively. The grand mean for both types of sentiment rating is 3.255. As evident from the means, the difference is only 1.92 %. The difference is statistically insignificant with ($F_{1,\ 28} = 0.034093$, $p > 0.005$). Hence the null hypothesis H0 was accepted and H1 was rejected, which by extension, validates our model.

Table 5 Weighted scores of the votes

Review #	Average	Model's score	Error	% error
1	3.39	3.21	0.181	3.6 %
2	1.81	1.07	0.748	15 %
3	4.73	4.21	0.523	10.5 %
4	4.24	4.05	0.187	3.7 %
5	3.37	3.33	0.038	0.8 %
6	3.63	3.48	0.154	3.1 %
7	3	3.46	−0.457	−9.1 %
8	1.84	1.88	−0.034	−0.7 %
9	3.53	2.86	0.671	13.4 %
10	2.61	2.43	0.172	3.5 %
11	3.21	3.47	−0.257	−5.1 %
12	4.16	3.95	0.212	4.2 %
13	3.63	4.65	−1.014	−20.3 %
14	4.05	4.21	−0.160	−3.2 %
15	2.11	2.10	0.003	0.1 %

Table 6 Student-t test for correlation

Correlation test (student t test)	
Correlation coefficient	0.896425516
tTab	0.063928134
tcal	7.292754614
Correlation	YES

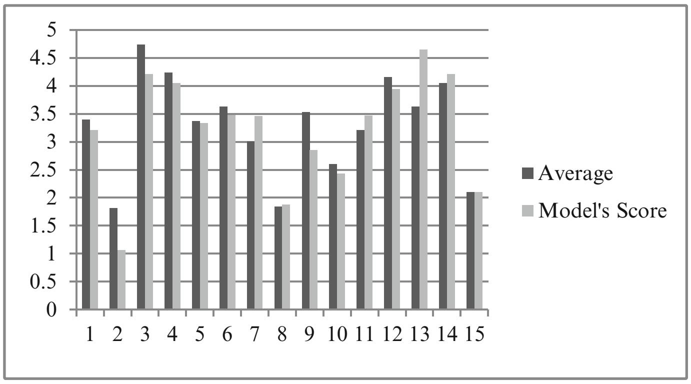


Fig. 9 Comparison of weighted values of votes and ratings obtained from SENTRAL

Conclusion

Analysing and obtaining structured feedback from online product evaluation by users provide an enormous value for the different services of a company, such as marketing, design, engineering, etc. Users' reviews are available online and almost

Table 7 ANOVA results

Anova: single factor	H0:	The difference between SENTRAL's score and human ratings is not significant			
	H1:	The difference is significant			
Summary					
Groups	Count	Sum	Average	Variance	
Weighted Average obtained from human ranking	15	49.31	3.287333	0.775278	
Model's score	15	48.36	3.224	0.989483	
ANOVA					
Source of variation	SS	df	MS	F	P-value
Between groups	0.03008333	1	0.030083	0.034093	0.85483920
Within groups	24.7066533	28	0.88238		
Total	24.7367366	29			

ANOVA result: **F crit = 4.195971819** > F (0.034093) Accept hypothesis H0

for free. However, the huge amount of data and the complexity limit their usability. This paper is a first step toward automatically analysis of user evaluation with focus on sentiment rating. The developed methodology is demonstrated on a case and was evaluated against a sample human rating.

Whether conversation in person or expressed in textual form online, subjectivity and sentiment add richness to information being shared. Captured electronically, customer sentiment can go beyond facts and rumours and convey unbiased mood, opinion and emotion. This carries immense business value. Listening for brand mentions, complaints and concerns is the first step in social engagement program for any company. Businesses that can listen, could potentially uncover sales opportunities, measure satisfaction, channel reactions to marketing campaigns, detect and respond to competitive threats.

An algorithm like SENTRAL will help save a lot of time by easily and quickly obtaining key information from data sources. Compared to other sentiment analysis models discussed earlier, SENTRAL provides lesser computing complication with a simple algorithm. Though the automation of this model requires a decent platform, it definitely does not necessitate very complex computing facilities and is capable of running on a reasonably powerful personal computer. This algorithm can be used to find out the global satisfaction of a particular product in the market by comparing the satisfaction scores of similar products. It can possibly be used to find out the trend of a product and to predict its performance in the future as well.

One aspect that might be a drawback in this model is quality of input data. Though our model uses a dictionary to expand all generally used acronyms, typos (typographical errors) and grammatical mistakes in the target data might pose a problem at the accuracy level of sentiment rating. This of course, is at the mercy of the customer's linguistic capabilities. The future improvements in SENTRAL could on certain extensions of the current model to isolate pleasant and unpleasant opinions on specific product features. This information can then be given to designers to improve future versions of the product and can also be used in analyzing the sentiments on a competitor's products.

We conclude that our algorithm SENTRAL is a potential contributor to intelligent automation that enables machines to understand and interpret the complex spectrum of signals present in the human world back to humans in quantitative way.

References

1. McGue M, Bouchard TJ (1998) Genetic and environmental influences on human behavorial differences. Annu Rev Neurosci 21:1–24
2. Lewis K, van Horn D (2013) Design analytics in consumer product design: a simulated study. In: ASME international design engineering technical conferences, Portland
3. Bollen J, Mao H, Zeng X-J (2011) Twitter mood predicts stock market. J Comput Sci 2(1):1–6
4. Caragea C, McNeese N, Jaiswal A, Traylor G, Kim HW, Mitra P, Wu D, Tapia AH, Giles L, Jansen BJ (2011) Classifying text messages for the haiti earthquake. In: Proceedings of the 8th international conference on in- formation systems for crisis response and management (ISCRAM2011)
5. Culotta A (2010) Towards detecting influenza epidemics by analyzing Twitter messages. In: Proceedings of the first workshop on social media analytics (SOMA '10). ACM, New York, pp 115–122
6. Liu B (2010) Sentiment analysis and subjectivity. In: FJN Indurkhya (ed) Handbook of natural language processing, Chicago
7. Buttle F (2003) Customer relationship management. Butterworth-Heinemann, Oxford
8. Berry MJ, Linoff G (1997) Data mining techniques: for marketing, sales, and customer support. Wiley, New York
9. Bennekom FCV (2002) Customer surveying: a guidebook for service managers. Customer Service Press, Bolton
10. Kushal D, Lawrence S, Pennock D (2003) Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In: WWW2003, 20–24 May 2003, Budapest
11. Tucker C, Kim H (2011) Predicting emerging product design trend by mining publicly available customer review data. In: Proceedings of the 18th international conference on engineering design (ICED11), 6, pp 43–52
12. Stone PJ, Dunphy DC, Smith MS, Ogilvie DM (1966) The general inquirer: a computer approach to content analysis. MIT Press, Cambridge, MA
13. Iker HP (1974) SELECT: a computer program to identify associationally rich words for content analysis. I. Statistical results. Comput Humanit 8:313–319
14. Herring SR, Poon CM, Balasi GA, Bailey BP (2011) TweetSpiration: leveraging social media for design inspiration. CHI Extended Abstracts, pp 2311–2316. ACM, 2011
15. Nazarenko A, Habert B, Reynaud C (1995) "Open response" surveys: from tagging to syntactic and semantic analysis. In: Proceedings of JADT (3rd international conference on statistical analysis of textual data), Vol. II, pp 29–36, Rome
16. OConnor B, Balasubramanyan R, Routledge BR, Smith NA (2010) From tweets to polls: linking text sentiment to public opinion time series. In: Proceedings of the international AAAI conference on weblogs and social media, pp 122–129
17. Pak A, Paroubek P (2010) Twitter as corpus for sentiment analysis and opinion mining. LREC conference, pp 24–37
18. Chowdary G (2003) Natural language processing. Annu Rev Inf Sci Technol 37:51–89
19. Liddy E (1998) Enhanced text retrieval using natural language processing. Bull Am Soc Inf Sci 24:14–16

20. Naman M, Boase J, Lai C-H (2010) Is it really about me? Message content in social awareness streams. In: Proceedings of the 2010 ACM conference on Computer supported cooperative work, pp 189–192
21. Bollen J, Mao H, Pepe A (2011) Modelling public mood and sentiment: twitter sentiment and socio-economic phenomena. AAAI conference on weblogs and Media. Michigan, pp 450–453
22. Hu M, Liu B (2004) Mining and summarizing customer reviews. SIGKDD. pp 168–177
23. Manning Klein D, & D, C (2003) Accurate unlexicalized parsing. 41st Meeting of the Association for Computational Linguistics, pp 423–430
24. Whissel C (1989) The dictionary of affect in language. Academic, London
25. Bryne R (Director) (2006) The secret [motion picture]
26. Miller GA (1995) WordNet: a lexical database for english. Commun ACM 38(11):39–41, ACM New-York
27. de Marneffe M-C, Manning CD (2008). The Stanford typed dependencies representation, CrossParser '08 Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation. Association for Computational Linguistics Stroudsburg, pp 1–8

Collaborative Evolution of 3D Models

Juan C. Quiroz, Amit Banerjee, Sushil J. Louis, and Sergiu M. Dascalu

Abstract We present a computational model of creative design based on collaborative interactive genetic algorithms. In our model, designers individually guide interactive genetic algorithms (IGAs) to generate and explore potential design solutions quickly. Collaboration is supported by allowing designers to share solutions amongst each other while using IGAs, with the sharing of solutions adding variables to the search space. We present experiments on 3D modeling as a case study, with designers creating model transformations individually and collaboratively. The transformations were evaluated by participants in surveys and results show that individual and collaborative models were considered equally creative. However, the use of our collaborative IGAs model materially changes resulting designs compared to individual IGAs.

Introduction

Design is a goal-oriented, constrained decision-making activity, involving learning about emerging features [1]. It is usually characterized by four phases – conceptual design, detailed design, evaluation, and iterative redesign [2]. Within this context, creativity has the potential to occur when a designer purposely shifts the focus of the search space [3]. Specifically, the ability to perform goal-oriented shifts while brainstorming and exploring potential solutions is crucial to creativity in the design process. This is accomplished implicitly by the designer's understanding of the problem changing over time, or explicitly by considering additional traits which may yield interesting solutions [3].

J.C. Quiroz (✉)
Independent contractor, Reno, NV, USA
e-mail: juancq@gmail.com

A. Banerjee
Pennsylvania State University, Middletown, PA, USA

S.J. Louis • S.M. Dascalu
University of Nevada, Reno, NV, USA

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_28

We present a computational model of creative design based on collaborative interactive genetic algorithms (IGAs) which supports goal-oriented shifts by adding variables to the search space. In our model, designers individually guide IGAs to generate and explore design solutions quickly [2]. The model allows designers to share solutions amongst each other by presenting designers with a sample of solutions generated by their peers. When a designer selects a solution from a peer, the solution is injected into his/her IGA population, with this injection adding new variables to the search space.

In our previous work [4, 5], we introduced our model of creative design and presented a pretest of the model for user guided design of floorplans, but without expanding the search space with additional variables. The pretest results showed that floorplans created collaboratively were considered to be more original than floorplans created individually. Following the pretest, we conducted a user study where we addressed the question of whether collaboration alone – without expanding the search space – introduced the potential to generate creative solutions [5]. Results showed that floorplans created collaboratively were considered to be more revolutionary and original than floorplans created individually.

In this paper, we present experiments where the search space is expanded by adding variables during the evolutionary search. We use 3D modeling as the case study for the experiments. However, rather than creating 3D models from scratch, we explore transformations of 3D models with vertex programs.

The rest of the paper is organized as follows. We start by presenting a discussion of the computational model and the 3D modeling case study. Next, the experimental setup is presented in detail, followed by user study results and discussions.

Computational Model of Creative Design

Models of creative design presented by the research design community manipulate the search space through the use of techniques, including combination, analogies, transformation, emergence, and first principles [6]. Genetic algorithms provide a means for exploring a search space consisting of potential solutions that meet a given set of requirements [7]. However, there are times when it is difficult, if not impossible to define a suitable fitness function, especially when dealing with problems require subjective evaluation. An interactive genetic algorithm empowers the user to drive evolution by replacing the fitness evaluation [8], and enables users to guide evolution based on their sense of aesthetics, intuition, and domain expertise.

Our computational model of creative design leverages the exploration power of the GA, the visualization and subjective feedback integration of the IGA, and collaboration in order to allow designers to shift the focus of the search space during an evolutionary run. Our model is unique in (1) using IGAs to guide the subjective exploration of changing search spaces, and (2) using collaboration to change the search space by adding variables. By using IGAs in our model, users can incorporate their personal preference, sense of aesthetics, intuition, and expertise into the search process. In addition, each user decides when to take solutions from peers, meaning that the user always remains in control of his/her own IGA.

Collaboration allows designers to share expertise, to be exposed to traits they may not have considered, and to complement each other in the task of exploring solutions which meet a given set of requirements. In our model, designers start with different variable sets. The designers are exposed to solutions being explored by their peers during collaboration and consequently to the different effects resulting from different variable sets. Taking solutions from peers allows designers to expand their search space by automatically incorporating the variables being explored by their peers.

Figure 1 illustrates our computational model of creative design. The figure illustrates three users collaborating with each other. Each user interacts with a GA by acting as the subjective evaluation. Evaluation may consist of subjective evaluation only, or a combination of subjective and objective evaluations. The arrows between the IGAs represent the communication that takes place between the peers. If a user likes a design solution from one of his/her peers, then the user has the option to inject that solution into his/her population, thus introducing a search bias. For implementation details of the collaborative IGA model, see [4] and [5].

Our collaborative IGA computational model is a special case of a case injected genetic algorithm (CIGAR) [9], where (1) each user serves as a case base to peers, and (2) each user determines when and how many individuals to inject into his/her population, instead of injection being done in an algorithmic fashion. When a user chooses to inject a solution from one of his/her peers, the introduced bias will not only become apparent in the user's own population, but it will also be visible to

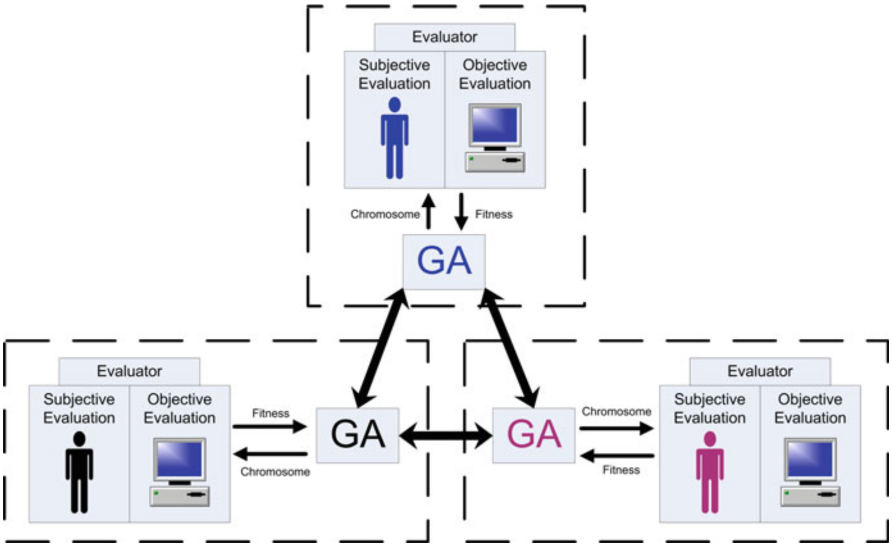


Fig. 1 Computational model of creative design

his/her peers, since users can always see a subset of each others' solutions. For example, as "user A" interacts with the IGA, the changes in the population of user A will be reflected on the screens of the peers of user A. Thus, each user participating in collaboration serves as a dynamic case base to his/her peers.

Typically in CIGARs a case base of solutions to previously solved problems is maintained. Based on problem similarity, individuals similar to the best individuals in the current population are periodically injected from the case base, replacing the worst individuals in the population [9]. In our computational model, the designer plays the role of determining how many, when, and which individuals to inject at any step during the collaborative evolutionary process. If the injected individuals make a positive contribution to the overall population, then they will continue to reproduce and live on, while injected individuals that do not improve the population performance will eventually die off. Hence, the user is not penalized for injecting subpar individuals. We use fitness biasing (linear scaling) to ensure that injected individuals survive long enough to leave a mark on the host population.

Creative Potential of Model

Boden describes two types of creativity in design, P-creativity and H-creativity [10]. P-creativity (personal or psychological creativity) occurs when the design is creative to the designer. H-creativity occurs when the design is creative when compared to all that has been created and produced historically by all of humanity. S-creativity (situated creativity), a third type which has also been presented, occurs when the resulting design is novel to that particular situation, but not necessarily be creative to the individual or creative historically [11]. In our model, at the individual level designers guide P-creative processes while interacting with the IGA. During collaboration, the sharing of design solutions allows the designers as a group to guide the S-creative process. Specifically, users can begin exploration of distinct search spaces (defined by different variable sets), and through collaboration, explore search spaces defined by combinations of their variable sets.

3D Modeling Representation

We use 3D modeling as the case study for our experiments. Rather than creating 3D models from scratch, we perform modifications to existing and well-formed 3D models by evolving vertex programs. The vertex programs allow for an operation to be applied on a per vertex basis for every vertex on a 3D model. For the experiments in this paper we used the OGRE 3D rendering engine and Cg as the GPU programming language for the vertex programs.

We evolve vertex programs with genetic programming (GP) [12]. GP is an evolutionary computation technique where each individual in the population is a computer program. The computer program is represented using a tree structure (GP tree) and the operations of the GP tree are typically mathematical operations.

Figure 2 illustrates our representation of the vertex programs in the IGA as a bit encoded binary tree. A binary tree is a tree data structure in which each node has at most two child nodes. For example, a node with index *i* would have its children at indices $2i + 1$ and $2i + 2$. In our representation, all leaves are at the same depth and every parent node has two child nodes. From the perspective of GA encoding, storing a binary tree in an array has the advantage of being readily mapped to a bit string.

In our implementation, parent nodes consist of binary operations on the children nodes, with these operations including addition, subtraction, multiplication, and division. It was seen in preliminary experiments that unary operators, such as the trigonometric functions, also created a wide range of interesting transformations of 3D models, and therefore a second array is used to store unary operators applied to each child node in the tree, as shown in Fig. 2. We use the trigonometric functions of sine, cosine, and tangent. The leaf nodes in the tree consist of variables including

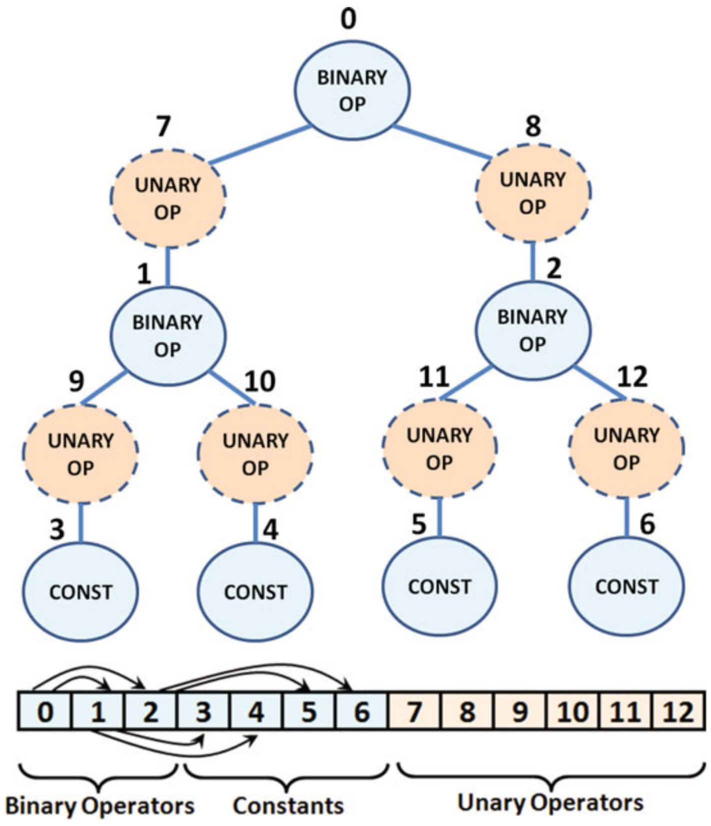


Fig. 2 Binary string representation of vertex programs

the coordinates of the current vertex (x, y, or z), the current simulation time looping from 0 to 2π, and random numbers between −10 and 10. The binary operations, the set of constants, and the unary operators are combined into a single bit string array manipulated by the IGA as illustrated in Fig. 2.

The decoded equation of each individual in the population is written to a Cg file. The decoded equation modifies all of the x, y, and z coordinates with the same equation, or only one of the x, y, or z coordinates. For example, Eq. 1 modifies the x, y, and z coordinates of each vertex with the same equation:

$$p.xyz = p.xyz + (2.2 - (p.x/11)) + (7 * \cos(p.y)); \tag{1}$$

where p.xyz represents the x, y, and z coordinates of the current vertex, p.x is the x coordinate of the current vertex, and p.y is the y coordinate of the current vertex. The value of the equation on the right is added to each of the x, y, and z coordinates of the current vertex. Thus, Eq. 1 is equivalent to the following:

$$\begin{aligned} p.x &= p.x + (2.2 - (p.x/11)) + (7 * \cos(p.y)); \\ p.y &= p.y + (2.2 - (p.x/11)) + (7 * \cos(p.y)); \\ p.z &= p.z + (2.2 - (p.x/11)) + (7 * \cos(p.y)); \end{aligned}$$

The vertex program representation enables us to have a first set of users evolve programs that modify the x coordinate, and a second set of users evolve programs that modify the y coordinate, as illustrated in Fig. 3. Through collaboration, the first set of users can inject solutions from the second set of users, resulting in their respective search spaces expanding from exploring equations that only modify the x and y coordinate, or only the y and z coordinate, to equations that modify all the coordinates of the 3D model.

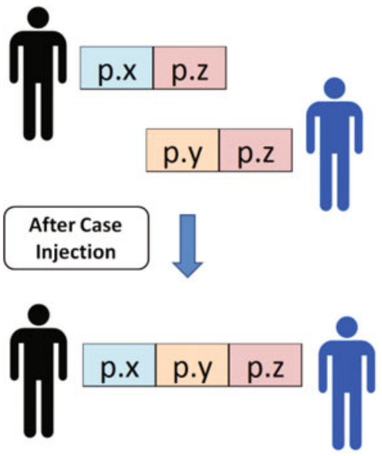


Fig. 3 Expanding design variable space through collaboration

Experimental Setup

The experiments presented in this paper were conducted in an environment built with the OGRE rendering engine. The goal of the experiments is to show that the use of our collaborative computational model results in solutions that are more creative compared to solutions generated using a simple IGA. To this end, the experiment consisted of three phases: (1) of designs, (2) first evaluation of the designs, and (3) online evaluation of the designs. These are described in detail in the next subsections.

Design Creation

A group of 20 students from the Computer Science Department at the University of Nevada, Reno, ("the design participants") used simple IGAs and our collaborative model to create transformations of the 3D models individually and collaboratively. During the creation of the designs, participants were split into pairs. When using our collaborative model, participants were only allowed to share solutions within each pair. During collaboration between users, the search space was expanded as shown in Fig. 3.

We used an ABA experimental design to test the hypothesis that our collaborative model results in more creative solutions when compared to a simple IGA. In the experimental design, the baseline condition (A) was participants creating solutions individually with a simple IGA, and the experimental condition (B) was participants creating solutions collaboratively with our computational model of creative design. The experiment consisted of each design participant conducting an ABA session (individual session, collaborative session, individual session) followed by a BAB session (collaborative session, individual session, collaborative session). The goal of this design is to show that the use of our computational model, rather than time, is the controlling variable if there is a change in behavior between baseline and experimental conditions [13]. Our hypothesis is that the resulting scores (from a 7-point Likert scale) would resemble a zig-zag pattern as illustrated in Fig. 4, with

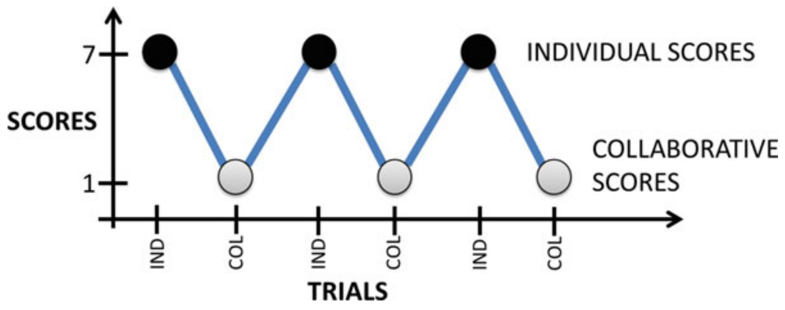


Fig. 4 Hypothesized comparison of scores between individual and collaboratively created models

the average scores of models generated during individual trials being close to 7, and the average scores of models generated during collaborative trials being close to 1.

Figure 5 shows the three models used in the user study: (1) a futuristic female model in a blue suit, (2) a green ninja, and (3) a white robot. Half of the design participants were assigned to a first design group, while the other half where assigned to a second design group. The first design group created transformations for the models in this order: (1) **individual** – female, (2) **collaborative** – ninja, (3) **individual** – robot, (ABA) and (4) **collaborative** – robot, (5) **individual** – ninja, and (6) **collaborative –** female (BAB). The second design group created transformations similarly to the first design group, except that in trials 4–6 the second design group begins with the female model instead of the robot model: (4) **collaborative – female**, (5) individual – ninja, and (6) **collaborative robot** (BAB).

Each design participant created at least two transformations for each of the models during the collaborative and the individual sessions. The design participants were then asked to pick the solutions they considered the most creative from each set of models, for a total of six final solutions from each design participant: (1) female – individual, (2) female – collaborative, (3) ninja – individual, (4) ninja – collaborative, (5) robot – individual, and (6) robot – collaborative. This set of final solutions was evaluated as described in the next subsection.

Evaluation

Our evaluation is based on the work of Thang et al. [14], which consists of criteria derived from the Creative Product Semantic Scale [14, 15, 16]. The participants were asked to rate model transformations on a 7-point Likert scale on the following

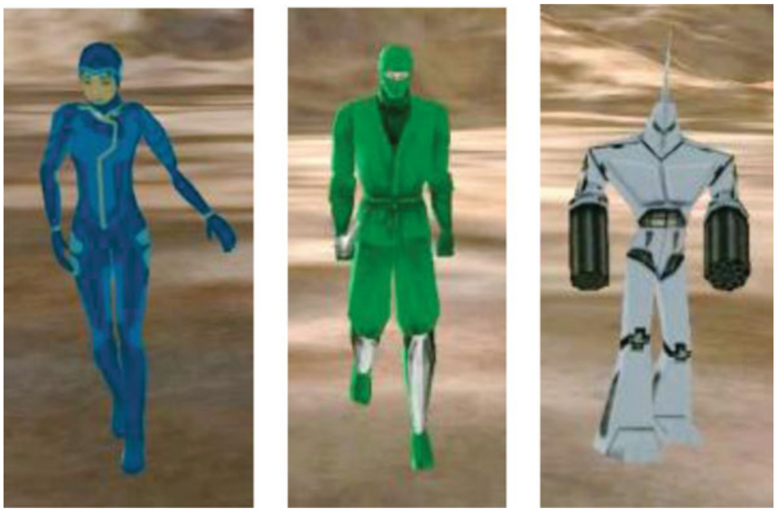


Fig. 5 Original 3D models used during experiments

criteria: creativity, novelty, surprising, workability, relevance, and thoroughness. However, instead of simply using the terms from this rating scale, we presented the users with five statements, and participants specified the degree to which they agreed or disagreed with the statements. The evaluation participants were only informed that a group of students had created a set of transformations of models, representing special effects for a video game.

The first evaluation statement, related to creativity, was: “The transformation is creative.” Many definitions of creativity exist, and the definition largely depends on the context and problem domain. Therefore, we provided the evaluation participants with the following definition to evaluate the created designs: “A creative transformation is a transformation that is new, unexpected, and valuable.” The statement could be answered by selecting between “Extremely Creative” (coded as 1) versus “Not Creative At All” (coded as 7), or with a number in between 1 and 7.

The rest of the evaluation statements did not use the terms workability, relevance, and thoroughness, to avoid ambiguity regarding the meaning of these terms. Instead, the evaluation asked whether the transformation with or without tweaks could be used in a video game, which addressed workability, relevance, and thoroughness. The other four statements in the evaluation were: (1) The transformation can be used in a video game; (2) The transformation with minor tweaks can be used in a video game; (3) The transformation is novel; and (4) The transformation is surprising. Users could answer these questions on a 7-point Likert scale as “Very True” (coded as 1) versus “Not True At All” (coded as 7).

The first group of evaluation participants evaluated the final solution sets created and selected by the design participants. Each evaluation participant evaluated two final solution sets; hence, each evaluation participant evaluated a total of 12 models.

After the first evaluation, we conducted a second online evaluation. A total of 16 adult volunteers completed the online evaluation. We selected the best six individually created models and the best six collaboratively created models for online evaluation. We made a video of these 12 models, posted the video online, and collected data via an online survey. The online survey used the same evaluation criteria as the first evaluation phase, except that we removed the statement asking whether the transformation with minor tweaks could be used in a video game to make the survey shorter.

Results and Discussion

Below we present the results of the first evaluation and the online evaluation of the designs created by the design participants. In addition, we provide examples of models created individually and collaboratively, and a discussion of how our collaborative model changes design.

First Evaluation

Figure 6 illustrates the evaluation scores received from the first evaluation participants for the statement "The transformation is creative." The average scores and boxplots are for the models created by the first design group and the second design group during individual and collaborative trials. The difference in these design groups is the order in which the models were evolved, as shown in the top axis of the plot. The boxplots compare the distributions of scores between individual and collaborative trials. The average scores between individually and collaborative trials were compared using a Student's *t*-test to verify statistical significance. We did not account for sample size in the statistical analysis.

For the first design group, trials 1–3 exhibit a zig-zag pattern that is the opposite of our hypothesis. We expected the individually created models to receive scores closer to 7 and the collaboratively created models closer to 1, yet the average scores and boxplots show the opposite for trials 1–3. Thus, when collaboration was introduced, the resulting models were considered less creative. For trials 4–6 of the first design group, we see scores supporting our hypothesis. That is, when users created models collaboratively, the resulting models were more creative. For the second design group, only trials 4–6 seem to support our hypothesis that models created collaboratively are more creative.

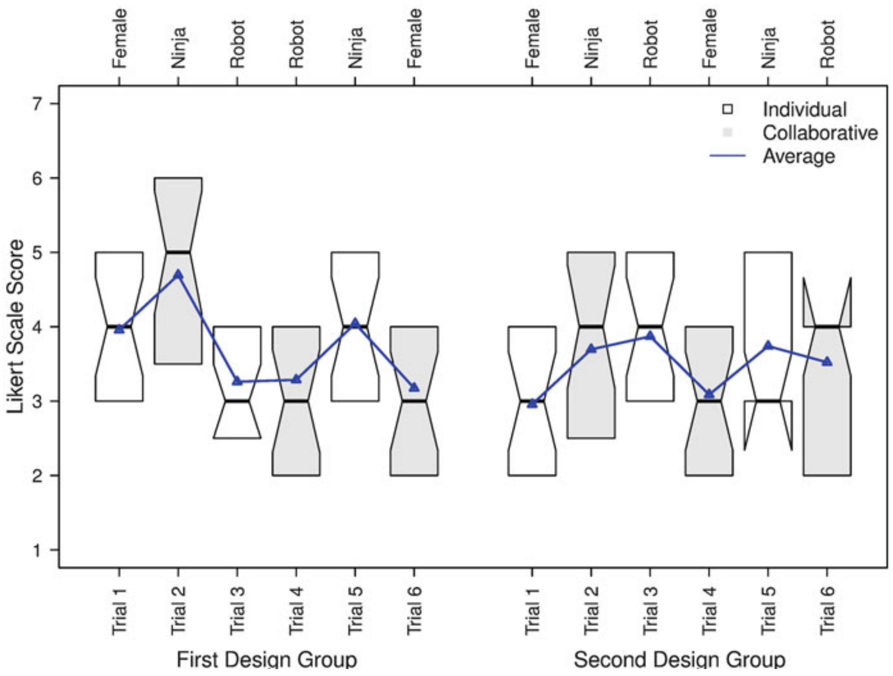


Fig. 6 Scores for "The transformation is creative" statement (*1* Extremely creative, *7* Not creative at all)

Using the Student's *t*-test, we compared the average scores between successive trials to determine whether the changes between the average scores of collaborative and individually created models were statistically significant. For the first design group, the change from trial 2 (collaborative) to trial 3 (individual) was statistically significant ($p < 0.05$). This was unexpected because it shows that when users created models individually, after having created models collaboratively, the resulting models were scored as more creative. This is also clear from the lack of overlap between the confidence intervals of the medians (denoted by the notches of the boxplots) of the second and third trial. The change in average scores from trial 5 (individual) to trial 6 (collaborative) was the only change in average score that was statistically significant ($p = 0.05$) that supported our hypothesis. For the second design group, the average scores are closer to our hypothesized scores, especially after the second trial. Yet, none of the changes in average scores were statistically significant ($p < 0.05$).

The boxplots from Fig. 6 show that the ninja model was the least popular of the three models, which can be especially appreciated in the first design group. In the first design group, the individually created robot model and the collaboratively created female model were considered the most creative. About 75 % of the answers (as indicated by the top of the box) for these two models were concentrated below the median score of 4. In the second design group, the individually and collaboratively created female models were considered the most creative. Finally, the ninja model was considered the least creative in both the first and the second design group.

Figure 7 illustrates the scores for the statement of whether the transformation could be used in a video game. The white robot in both the individual and collaborative sessions of both design groups was found to be the most suitable to be used in a video game as shown by the lowest average scores. For trials 1–3 in the first and second design groups, the results are the opposite of our hypothesis, while for trials 4–6 the results are closer to our hypothesis. For the first design group, the change in average from trial 2 (collaborative) to trial 3 (individual) was statistically significant ($p < 0.05$), and the change in average from trial 4 to trial 5 was also statistically significant ($p < 0.05$), with the latter change supporting our hypothesis.

None of the other changes in average scores were statistically significant in the first design group. In the second design group, none of the changes were statistically significant.

Figure 8 shows the evaluation scores for the question asking whether the model with minor tweaks could be used in a video game. The model found to be most suitable after minor tweaks was the white robot, which is consistent with the results from Fig. 7. In the first design group, the individually created robot received the best scores, whereas in the second design group the collaboratively created robot received the best scores. None of the changes in average scores were statistically significant ($p < 0.05$).

Figure 9 illustrates the evaluation scores for the novelty criterion. For the first and second design groups, the average scores and boxplots for trials 4–6 show the

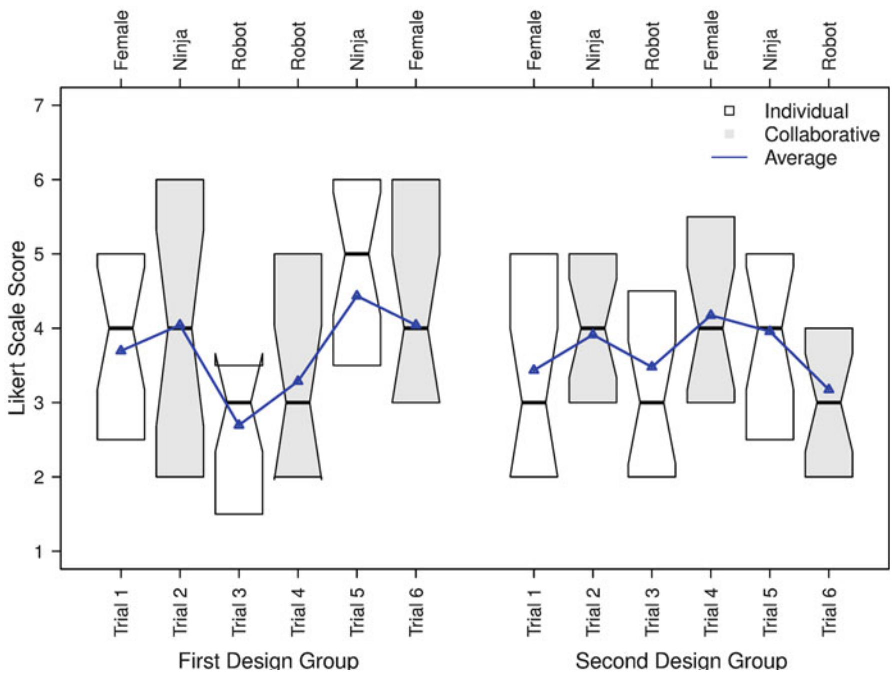


Fig. 7 The transformation can be used in a video game (*1* – very true, *7* not true at all)

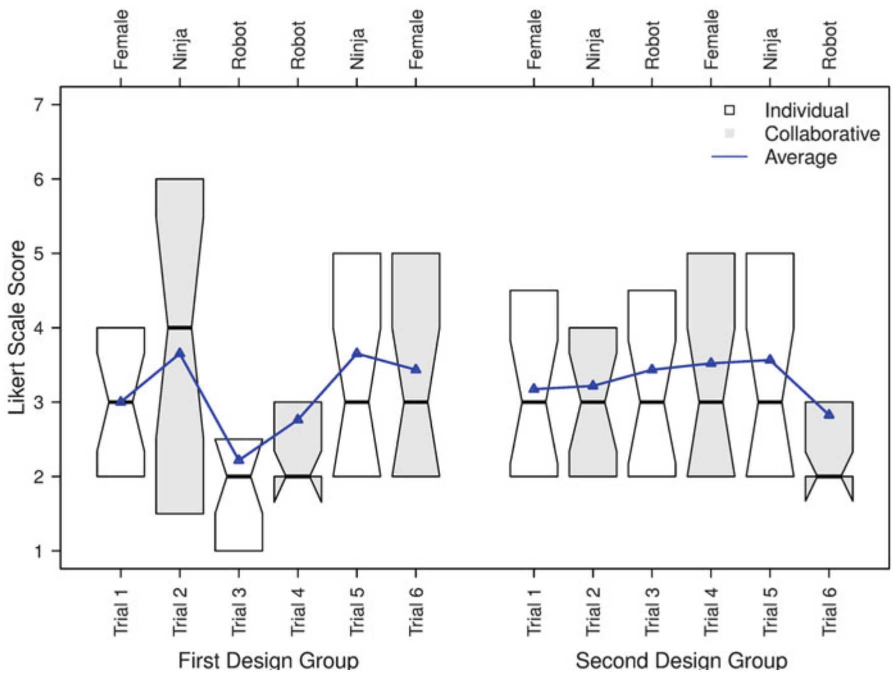


Fig. 8 The transformation with minor tweaks can be used in a video game (*1* very true, *7* not true at all)

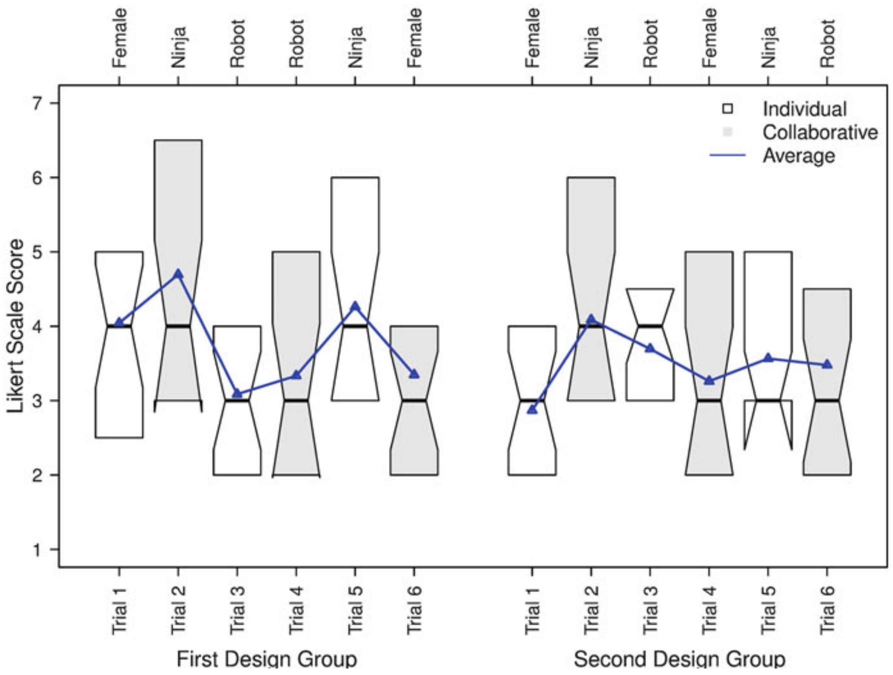


Fig. 9 The transformation is novel (*1* very true, *7* not true at all)

desired zig-zag pattern. However, for the first design group only the change in average score from trial 2 (collaborative) to trial 3 (individual) was statistically significant ($p < 0.05$). For the second design group, only the change from the trial 1 (individual) to trial 2 (collaborative) was statistically significant ($p < 0.05$). Both of these results are the opposite of our hypothesis.

Figure 10 illustrates the scores for the surprising criterion. In the first design group, trials 4–6 reflect the desired score pattern. Furthermore, the collaboratively created blue female model received the best scores with at least 50 % of scores being in the range 1–3. In the second design group, both the individually and collaboratively created solutions received similar scores. However, the collaboratively created blue female model received at least 25 % of scores in the range 1–2.

For the second design group, none of the changes in average scores were statistically significant. For the first design group, the change from trial 2 to trial 3 was statistically significant ($p < 0.05$), which does not support our hypothesis. Yet, the change from trial 4 to trial 5, and from trial 5 to trial 6 were statistically significant ($p < 0.05$), while also supporting our hypothesis. This latter result is encouraging as it suggests that the 3D models were considered more surprising when our collaborative model was used to generate model transformations.

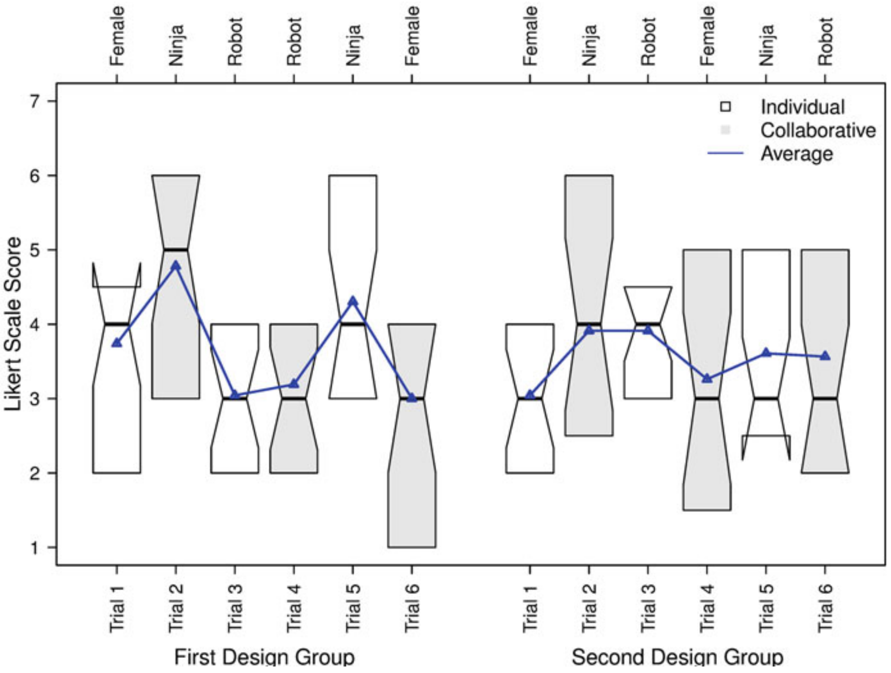


Fig. 10 The transformation is surprising (*1* very true, *7* not true at all)

Online Evaluation

The first evaluation phase showed that models created individually received similar scores to models created collaboratively. We believed that it would be difficult to prove that every solution generated collaboratively would be more creative than a solution created individually. In view of the results from the first evaluation, we created an online evaluation in which users viewed a video and provided scores for the models in the video via an online survey.

The video first presented the three original models, without any transformations, so that participants could appreciate the differences that the transformations made. The video then showed a first row of individually created transformations, followed by a second row of collaboratively created transformations. The online survey used the same evaluation criteria from the first evaluation phase, with one exception. We removed the question asking whether the transformation with minor tweaks could be used in a video game to make the survey shorter. The survey first asked participants to evaluate the row of individually created models, followed by evaluation of the row of collaboratively created models. After both rows of models had been evaluated, participants were asked which of the two rows they liked the most and which of the two rows was the most creative.

We had a total of 16 completed online evaluations. We aggregated the scores for all of the individual models per criterion and similarly for the collaborative models. We compared the averages using the independent samples *t*-test and we found no statistically significant differences in the results. With regard to which row participants liked the most, eight participants picked the row of individually created models, seven participants picked the row of collaboratively created models, and one participant did not answer. With regard to which row was the most creative, three participants picked the row of individually created models and 13 participants picked the row of collaboratively created models.

Discussion

Figure 11 shows examples of two models generated by the design participants. All of the effects on the models involved animation. Thus, it was difficult to capture the resulting effects in images. Our observations were that while both individual and collaborative solutions were interesting, the collaborative solutions had more dramatic effects. In addition, the collaborative solutions tended to be more chaotic, and thus had a less polished look.

Figure 11a shows a blue female model created individually. The female model starts by standing straight, and the evolved vertex program made the body of the female model curve from left to right in the shape of an "S." Even though the model is curved, all parts of the original model can be identified, and overall the model has a smooth and aesthetically pleasing look. Figure 11b shows a blue female model created collaboratively. The evolved vertex program made the female model expand upwards, making the model look like the tail of a comet. The interesting part of the model is that in the midst of the chaotic animation, the face of the model

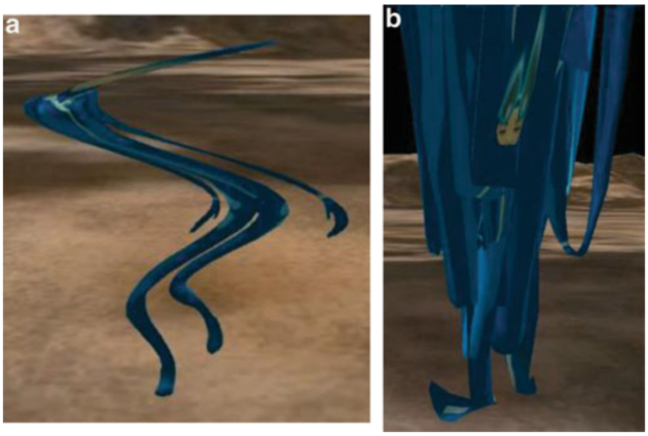


Fig. 11 Individual (**a**) and collaboratively (**b**) generated transformations of the female model

remains visible, along with some of the extremities, such as both legs and part of her arms. Yet, this model, while exciting, does give the impression of being a work in progress.

From the results from the first evaluation phase we can deduce that models created individually were equally creative as models created collaboratively. There are observations which we believe help explain some of the results obtained. First, we found that transformations on the blue female models resulted in the most interesting and aesthetically pleasing effects. The geometry and the skeleton, which dictates how a model moves when animated, of the female model resulted in transformations that looked polished and well-done compared to the transformations of the other two models. Transformations on the female model tended to have smooth transitions, and many times the original model was deformed in a curved fashion, resulting in soft edges. On the other hand, transformations on the robot and ninja models tended to result in sharp and jagged edges. In fact, we found that the same transformation applied to different models resulted in different effects due to the model geometry and the skeleton. Overall, the ninja models resulted in the less interesting effects. This was particularly identified while we were building the sets of best individual models and best collaborative models for the online evaluation.

Regarding the online evaluation results, the fact that the majority of participants identified the row of collaborative models as the most creative, yet the average evaluation scores of collaborative and individual models were not statistically different, suggests a couple of points. First, the structure of the survey may have created a learning confound. In the online evaluation, the video was not included in the online survey web page. Participants had to follow a link to watch the video on an external web page. Therefore, participants would have had to switch back and forth between the survey web page and the video web page, instead of being able to score the models with the video always in view. A further nuance was that the video focused on each model for an average of 10–15 s. We had to impose this time limit in order to keep the video short. If participants wanted to watch one of the models in further detail, then they would have had to rewind the video to the right spot. Finally, we do not know whether participants watched the entire video once and then filled out the online survey, or whether they scored each model in tandem with the video. All of these nuances may have resulted in a learning confound that negatively biased the results.

Conclusions

In this paper we presented and evaluated the results of design evaluation on evolving a set of transformations on 3D models used in video games. The evolution is based on a collaborative model of creative design that uses interactive genetic algorithm and collaboration via solution injection among peers. The study is a major extension of our previous work in evolving 2D architectural floorplans. In the present study we look at issue of expanding the design space by introducing new variables, which has the potential to lead to uncovering solutions (creative or otherwise) that would not have been possible otherwise. While the evaluation scores did not fully support our hypothesis that our computational model of creative design increases creative content of solutions, the use of our collaborative model materially changed the resulting designs due to the expansion of the search space via collaboration. Finally, our work demonstrates that our collaborative IGA computational model matched with designer collaboration offers a valuable mixed-initiative approach to the use of evolutionary systems in design.

As part of the experimental design, we limited how the search space could be expanded during the course of evolution. This is different than how creativity occurs in design practice, where a designer expands the search space as a result of a better understanding of the problem and solution, leading to a creative leap. Therefore, the model needs to be further validated by testing the use of the model on an actual design task. We foresee designers encoding different requirements to explore with evolution, letting designers explore solutions collaboratively with IGAs, and use this as a basis for discussion of potential design solutions. The work presented here can be thought of as the general framework of incorporating IGAs in design with peer-to-peer collaboration with the objective of evolving "more creative" solutions than what is possible in a non-collaborative environment.

References

1. Gero JS, Kazakov V (2000) Adaptive enlargement of state spaces in evolutionary designing. Artif Intell Eng Anal Manuf 14:31–38
2. Bentley PJ, Wakefield JP (1997) Conceptual evolutionary design by genetic algorithms. Eng Des Autom J 3:119–131
3. Gero JS, Schnier T (1995) Evolving representations of design cases and their use in creative design. In: Computational models of creative design, Key Center of Design Computing, Sydney, pp 343–368
4. Banerjee A, Quiroz JC, Louis SJ (2008) A model of creative design using collaborative interactive genetic algorithms, In: Proc 3rd Conf Des Compt Cognit, Springer, pp 397–416
5. Quiroz JC, Louis SJ, Banerjee A, Dascalu SM (2009) Towards creative design using collaborative interactive genetic algorithms. In: Proc IEEE Cong Evol Comput, pp 1849–1856
6. Gero JS (2002) Computational models of creative designing based on situated cognition. In: Proc 4th Conf Creativ Cognit, ACM Press, Loughborough, pp 3–10
7. Golderberg DE (1989) Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, Boston
8. Takagi H (2001) Interactive evolutionary computation: fusion of the capacities of EC optimization and human evaluation. Proc IEEE 89:1275–1296
9. Louis SJ, Miles C (2005) Playing to learn: case-injected genetic algorithms for learning to play computer games. IEEE Trans Evol Comput 9:669–681
10. Boden M (2003) The creative mind: myths and mechanisms, 2nd edn. Routledge, London
11. Suwa M, Gero J, Purcell T (2000) Unexpected discoveries and S-invention of design requirements: important vehicles for a design process. Des Stud 21:539–567
12. Koza JR (1992) Genetic programming: on the programming of computers by means of natural selection. MIT Press, Cambridge, USA

13. Marwell G, Schmitt DR, Boyesen B (1973) Pacifist strategy and cooperation under interpersonal risk. J Pers Soc Psychol 28:12–20
14. Thang B, Sluis-Thiescheffer W, Bekker T, Eggen B, Vermeeren A, de Ridder H (2008) Comparing the creativity of children's design solutions based on expert assessment. ACM Press, New York, pp 266–273, In Proc 7th Int Conf Interaction Des and Children
15. Besemer S, O'Quin K (1986) Analyzing creative products: refinement and test of a judging instrument. J Creat Behav 20:115–126
16. Silvia PJ, Martin C, Nusbaum EC (2009) A snapshot of creativity: evaluating a quick and simple method for assessing divergent thinking. Think Ski Creat 4:79–85

Custom Digital Workflows with User-Defined Data Transformations Via Property Graphs

Patrick Janssen, Rudi Stouffs, Andre Chaszar, Stefan Boeykens, and Bianca Toth

Abstract Custom digital workflows aim to allow diverse, non-integrated design and analysis applications to be custom linked in digital workflows, created by a variety of users, including those who are not expert programmers. With the intention of introducing this in practice, education and research, this paper focuses on critical aspects of overcoming interoperability hurdles, illustrating the use of property graphs for mapping data between AEC software tools that are not connected by common data formats and/or other interoperability measures. A brief exemplar design scenario is presented to illustrate the concepts and methods proposed, and conclusions are then drawn regarding the feasibility of this approach and directions for further research.

Introduction

The persistent lack of integration in building design, analysis and construction calls for new approaches to information exchange. We argue that bottom-up, user-controlled and process-oriented approaches to linking design and analysis tools, as envisaged by pioneers of CAD [1, 2], are indispensable as they provide degrees of flexibility not supported by current top-down, standards-based and model-oriented approaches.

P. Janssen (✉)
National University of Singapore, Singapore, Singapore
e-mail: patrick.janssen@nus.edu

R. Stouffs
National University of Singapore, Singapore, Singapore

Delft University of Technology, Delft, Netherlands

A. Chaszar
Delft University of Technology, Delft, Netherlands

S. Boeykens
KU Leuven, Leuven, Belgium

B. Toth
Queensland University of Technology, Brisbane, Australia

DOI 10.1007/978-3-319-14956-1_29

We propose a platform to support a design method where designers can compose and execute automated workflows that link computational design tools into complex process networks [3, 4]. By allowing designers to effectively link a wide variety of existing design analysis and simulation tools, such custom digital workflows support the exploration of complex trade-offs between multiple conflicting performance criteria. For such explorations, these trade-offs often present a further set of questions rather than a final set of answers, so the method is envisaged as a cyclical process of adaptive workflow creation followed by iterative design exploration.

The adaptive-iterative design method requires a platform for designers to effectively and efficiently create and execute workflows. The remainder of this paper first gives a general overview of the proposed platform and then focuses on critical aspects of overcoming interoperability hurdles, specifically the creation of mapping procedures that map data between tools with incompatible data representations. We explore the feasibility of a data mapping approach that allows end-users to define their own customized mappers, applying it to an example scenario focusing on a digital design-analysis workflow linking parametric design tools to performance analysis tools.

The research method consists of building a test workflow comprising a geometric design model and analysis tools to evaluate lighting and thermal performance, and applying customized data mappings between these applications via property graphs. The data collected from this experiment includes observations of the types of data mappings required, the complexity of the mappings, and the modifiability of the mappings when editing of the workflow is needed. We conclude that linking design tools via customized data mappers is a feasible approach that can complement other existing mapping approaches, and we discuss future research directions.

Adaptive-Iterative Design Platform

In order to support the adaptive-iterative design method, a platform for creating and executing workflows is proposed. This platform is based on existing scientific workflow systems that enable the composition and execution of complex task sequences on distributed computing resources [5].

Scientific workflow systems exhibit a common reference architecture that consists of a graphical user interface (GUI) for authoring workflows, along with a workflow engine that handles invocation of the applications required to run the solution [6, 7]. Nearly all workflow systems are visual programming tools in that they allow processes to be described graphically as networks of nodes and wires that can be configured and reconfigured by users as required [8]. Nodes perform some useful function; wires support the flow of data, linking an output of one node to an input of another node.

Each workflow system acts to accelerate and streamline the workflow process, but each system also varies greatly in specific capabilities [9]. We aim to identify the functionality needed to develop a flexible, open and intuitive system for design-analysis integration, building on the capabilities exhibited by scientific workflow systems, and further capitalizing on recent advances in cloud computing.

Actor Model

The proposed platform is a type of scientific workflow system using an actor model of computation [10]. Nodes are *actors*, and the data that flows between nodes is encapsulated in distinct data sets, referred to as *data objects*. The actor model allows for a clear separation between actor communication (dataflow) and overall workflow coordination (orchestration).

We consider three types of actors: process actors, data actors and control actors. *Process actors* define procedures that perform some type of simulation, analysis, or data transformation. They have a number of input and output ports for receiving and transmitting data objects, as well as meta-parameters that can be set by the user to guide task execution. *Data actors* define data sources and data sinks within the workflow, including the data inputs and data output for the workflow as a whole. *Control actors* provide functionality related to workflow initiation, execution and completion. We focus here on the role of process actors in workflows, and the development of an approach to support custom data mapping procedures.

Process actors can be further classified into tool actors and mapping actors. Tool actors define procedures that wrap existing applications to make their functionality and data accessible to the workflow; while mapping actors define procedures that transform data sets in order to map the output from one tool actor to the input for another.

Figure 1 shows a conceptual diagram of an example network in which a parametric CAD system actor is connected to three evaluation actors: Revit for cost estimating; EnergyPlus for heating/cooling load simulation; and Radiance for daylighting simulation. The CAD system actor encapsulates a procedure that starts the CAD system, loads a specified input model and a set of parameter values, and then generates two output models. One of the models is a generated as an IFC file, which is directly consumed by the Revit actor (using the Geometry Gym Revit IFC plugin [11]), while the other model is generated as a text file. This latter model then undergoes two separate transformations that map it into both EnergyPlus and Radiance compatible formats. The simulation actors then read in this transformed data, run their respective simulations, and generate output data consisting of simulation results.

The output results may be merged and displayed to the user within a graphical dashboard in an integrated way in order to support decision making. Furthermore, such a dashboard may also allow users to manipulate the input parameters for the workflow. Each time such inputs are changed, the execution of the network will be

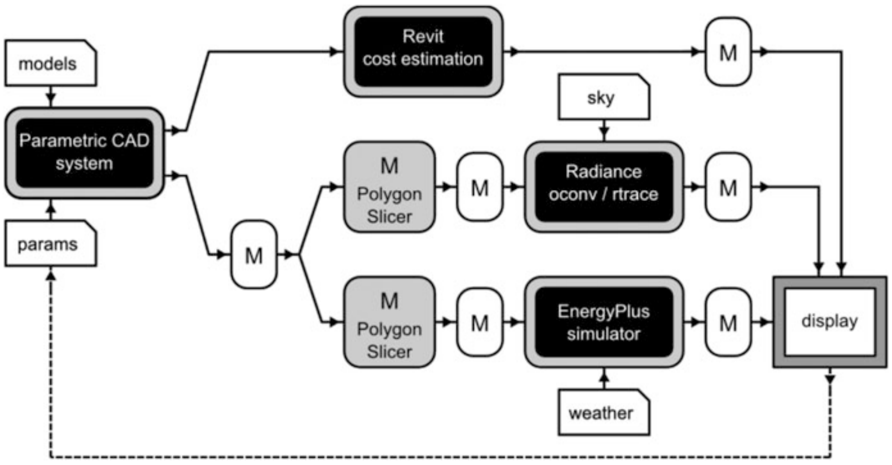


Fig. 1 Example network of actors. A parametric CAD system is linked to Revit, Radiance and EnergyPlus via various mappers (M). End users contribute *white* components, while actor developers build *grey* components and wrap *black* components representing existing tools

triggered, resulting in a new set of results. It is envisaged that more advanced data analytics tools could be developed in which multiple sets of results from the adaptive-iterative process could be analyzed and compared.

Platform Architecture

In order to support both the collaborative creation of workflows and their parallel execution, the proposed platform can be implemented as a web application deployed on a cloud or grid based infrastructure. Workflows (such as the one shown in Fig. 1) can be interactively developed by users within the web browser interface. When the user runs a workflow, it is uploaded to a workflow server as a workflow *job* consisting of a set of computational *tasks*, where it is executed in a distributed manner.

For scalability, both the procedures and the associated data objects are stored in an online distributed key-value database. Key-value databases are highly optimized for speed and scalability when storing and retrieving many documents. Although such documents may use a standardized format (e.g. JSON), they are typically used as containers for storing other data, with no restrictions on data models and data schemas. For example, a document may contain any number of serialized objects, text files or binary blobs. Both data objects and actor procedures are stored in documents in the key-value database. No restrictions are imposed on the types of data objects and actor procedures that can be stored.

Data Models for Data Mappers

Data mappings aim to overcome interoperability problems between tools that cannot be linked using existing approaches. We examine two common approaches for creating such mappers, and propose a complementary approach.

The various approaches to overcoming interoperability problems rely on three distinct levels of data representation: the *data model*, the *data schema*, and the *data file*. A data model organizes data using generic constructs that are domain independent. Due to this generic nature, the range of data that can be described is very broad. It offers a way of defining a data structure that is very flexible but relies on human interpretation of semantic meaning. For example, Tsichritzis and Lochovsky [12] distinguish seven types: relational, network, hierarchical, entity-relationship, binary, semantic network, and infological. Data models will often be coupled with highly generic languages for querying and manipulating data, variously called *data query languages* and *data manipulation languages*.

A data schema represents domain-specific information using semantic constructs related to a particular domain. Due to the highly specific nature of the constructs, the type of information that can be described tends to be relatively narrow. However, this manner of representing information supports automated interpretation of semantic meaning. The data schema is often built on top of a data model, by formally defining constraints that describe a set of allowable semantic entities and semantic relationships specific to a particular domain. This data schema is defined using a specialized language, variously called a *data definition language*, a *data description language*, or a *data modeling language*. Note that the data schema itself is distinct from the language used for describing the schema.

A data file uses syntactic constructs to describe how data is stored, either in a file or any other form of persistent storage. Going from the data file to the data schema is referred to as *parsing*, while the reverse is known as *serializing*. For any given data schema, there may be many different types of data files.

For example, consider the Industry Foundation Classes (IFC) representation for the exchange of Building Information Modeling data in the AEC industry [13]. In this case, the data model is an entity-relationship model, the data schema is an IFC schema defined in the EXPRESS modeling language, and the two main data files use a STEP or XML based file format. Note the use of XML for one of the file formats does not mean that an XML-based data model is being used for data manipulation. The data file is only used as a convenient way of storing the data.

Tool Interoperability

To link two tools with incompatible data representations, a formalized data mapping needs to take place [14]. The mapping is defined as a one-way transformation process, where the data representation of the source tool is mapped to the data representation of the target tool. In existing scientific workflow systems, this mapping process is called ‘shimming’ [15].

In most cases, the incompatibility exists at all levels: at the data model level, the data schema level, and the data file level. One approach in overcoming this type of incompatibility is to create direct file translators for each pair of tools to be linked. An alternative approach is to create indirect file translators that read and write to an intermediate representation. Within the AEC industry, these are the two main interoperability approaches being pursued, which we call *direct file translation* and *indirect file translation*.

The direct file translation approach has two key weaknesses. First, since separate data file translators need to be created for each pair of tools, the number of translators required increases rapidly as the number of tools increases. Second, the required generality of the translator means that no assumptions can be made regarding the sets of data to be processed, so these translators are complex to develop and maintain (as file formats are continuously updated and changes are often undocumented), and tend to be plagued by bugs and limitations.

With the indirect file translation approach, a range of different levels exist depending on the type of data schema. At the low-level end of the spectrum, data schemas may be limited to describing only geometric entities and relationships, while at the more high-level end of the spectrum, schemas may also describe complex real-world entities and relationships. The AEC industry has been using low-level geometric translators for many decades, based on file formats such as DXF, IGES and SAT. Such low-level geometry based file translation approaches are clearly limited, since they do not allow for the transfer of non-geometric semantic data.

At the more high-level end of the spectrum, the AEC research community has, since the 1970s, been working on ontological schemas that combine geometric information with higher level semantic constructs that describe entities and relationships in the AEC domain. Until recently, this approach was often conceptualized as a single centralized model containing all project data and shared by all project participants. However, this is now increasingly seen as being impractical for a variety of reasons, and as a result such high-level ontologies are now being promoted as a ‘smart’ form of file translation [16]. The latest incarnation of these efforts is the STEP-based IFC standard for BIM data storage and exchange.

Open Interoperability

A number of platforms are being developed that enable users to apply both direct and indirect file translators, focusing in particular on BIM and IFC. For example, the SimModel aims to allow BIM and energy simulation tools to be linked via a new XML based model aligned with IFC [17]; D-BIM workbench aims to allow a variety of tools for analyzing differing performance aspects to be tightly integrated with BIM tools [18].

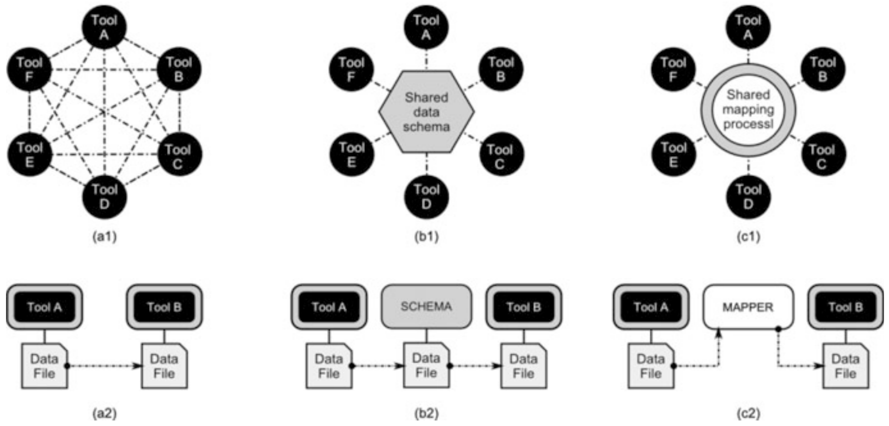


Fig. 2 Interoperability approaches. Linking individual tools via file translation (*left*); via a shared data schema (*middle*); or via a shared mapping process (*right*)

However, there are many sets of tools for which such translators have either not yet been developed or are no longer up-to-date. We therefore propose a more flexible and open type of platform that also supports linking tools for which direct and indirect file translators are not available. Such tools are linked using an approach that allows users to define their own customized data mappers tailored to the data exchange scenarios they require. Figure 2 shows the three approaches to linking tools. Note that this figure omits other approaches that do not require the neutral file format approach, e.g., agent-based interoperability [19]. The proposed platform will allow users to create automated workflows that mix-and-match all three tool-linking approaches, applying the most expedient approach for each pair of tools. For example, in cases where IFC-based mappers are already available, the user may prefer to simply apply such mappers, while in cases where no translators are available, customized data mappers can be developed.

End users will need to be provided with various tools and technologies in order to support them in this endeavor of developing mappers. This research focuses on communities of users with limited programming skills, with the goal to develop a mapping approach that allows such non-programmers to easily create and share custom mappings in a collaborative manner. Since users are focused on one specific data exchange scenario, these mappers can potentially be restricted to only a subset of the data schemas of the design tools being linked, thereby resulting in mappers that are orders of magnitude smaller and simpler when compared to general-purpose file translators.

To achieve this goal, the mapping approach must be both flexible and user-friendly. It must be flexible so that users can apply the same mapping strategies to a wide range of data exchange scenarios. It must also be user-friendly so that it supports users with limited programming skills in the process of creating and debugging mappers.

The complexity of creating such mapping is to a significant extent related to the levels of data incompatibility between the source tool and target tool. So far, the assumption has been that for most tools, data incompatibility will exist at all three levels: data model, data schema, and data file. This creates many difficulties for end-users attempting to create mappers. In particular, the user is required to deal with two different data models, each of which will have its own languages for data query, data manipulation, and schema definition. However, a simpler data exchange scenario can be imagined, where the input and output data for the mapping both use the same data model. In such a case, the user would be able to develop a mapping within one coherent data representation framework, including the option of using a single model based language for querying and manipulating data. In addition, given a single data model, it then becomes feasible to provide visual tools for developing such mappers, as will be discussed in more detail in the next section.

The simpler data exchange scenario with a common data model is clearly more desirable. The proposed approach is therefore to transform data exchange scenarios with incompatible data models into the simpler type of data exchange scenario where both input and output data sets have a common data model. With this approach, input data sets are created that closely reflect the data in the source data file, and output data sets are created that closely reflect the data in the target data file. The mapping task is then reduced to the simpler task of mapping from input data sets to output data sets using a common data model. Figure 3 shows the relationship between the mapping procedure and the source and target data files.

These input and output data sets may have data schemas, either informally defined in help documentation or formally defined using a schema definition language. In the latter case, the schema may be used to support visual mapping tools. In most cases, these schemas will be ‘reduced’ schemas in that they will only cover a sub-set of the data that might be represented in the source and target data files. For example, while the source and target tools may each have large and complex schema, a user may decide that for the task at hand, they will only be using a small sub-set of the entities and relationships defined in those schemas. As a result, these schemas may typically be small and highly focused on the task at hand.

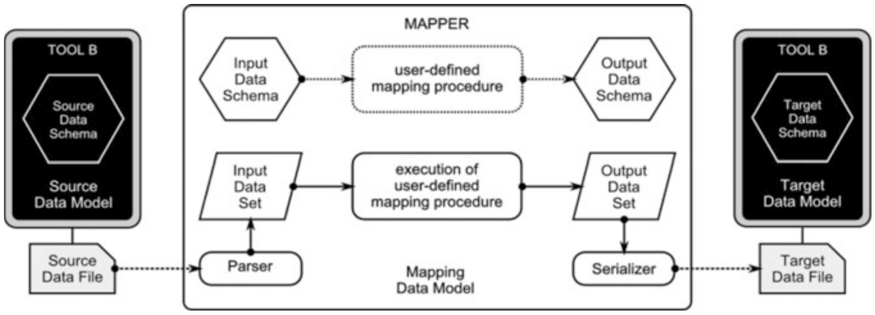


Fig. 3 The mapping of a source data file to a target data file involves three stages of data transformation: parsing, mapping, and serializing

The input and output data sets can be generated in various ways. One approach is to use a parser to convert the source data file into the input data set and a serializer to convert the output data set into the target data file. In both reading and writing the data files, the data is converted with the minimum amount of change, and neither the parser nor the serializer performs any actual data mapping. As long as a generic and flexible mapping data model is selected, the conversions between data file and data model should be relatively straightforward. In some cases, it may even be possible to automatically generate parsers and serializers. We focus here on the data mapping procedure, which requires more in-depth interpretation.

User Defined Data Mappings

Given a pair of automated parsing and serializing procedures, the task for the user is then to define the data mapping. This mapping procedure needs to be able to process any data set that adheres to the input data schema, and produce a data set that adheres to the output data schema. For creating such mapping procedures, a number of techniques can be identified, differing in complexity and power. For the simplest type of mappings, which we refer to as *declarative equivalency mapping*, user-friendly visual tools already exist, while for more complex types of mappings, which we refer to as *scripted mappings*, the user is forced to write code. In this research, we propose a powerful intermediate approach using flexible and highly generic visual programming tools. This intermediate level we refer to as *procedural query mapping*.

With declarative equivalency mappings, the input and output data schemas are formally defined, and the user defines a list of semantically equivalent entities between these schemas. Based on this user-defined information, a mapping procedure can then be automatically generated that will transform the input data set to the output data set. In some cases, it may be possible to define such mappings using visual tools [20]. One key problem with this approach is that only fairly simple mappings can be created using direct semantic mappings. More complex mappings may require a number of input entities to be processed in some way in order to be able to generate another set of target entities.

With procedural query mappings, the user creates data mapping rules using visual programming languages specialized for particular types of data models. These specialized languages include data query languages and data manipulation languages. The former are used for retrieving data from a data set, and the latter for inserting and deleting data in a data set. In many cases, the same language can be used for both querying and manipulation. A popular example is the Structured Query Language (SQL), which is used for both retrieving and manipulating data in relational databases. Other generic languages for retrieving and manipulating data include XQuery/XPath for data stored in XML data models, SPARQL for data stored in Resource Description Framework (RDF) data models, and jaql for data stored in JSON data models. Although such languages are specialized for certain data models, the languages themselves are semantically still highly generic.

With scripted mappings, the user develops mapping scripts using a programming language. Such scripted mapping may be appropriate in cases where complex mappings need to be defined. Consider the example in Fig. 1. The output from the CAD system cannot be easily mapped to either the input for EnergyPlus or input for Radiance. Instead, an additional step is required that performs Boolean operations on the polygons. For EnergyPlus, surface polygons need to be sliced where there are changes in boundary conditions (as each surface can only have one boundary condition attribute), and then infer what these boundary conditions are, i.e. internal, external or ground contact. For Radiance, surface polygons need to have holes cut where there are windows. These additional steps may have to be performed by a scripted mapper, the PolygonSlicer (Fig. 1).

For creating user-defined mappings within workflows, either declarative equivalency mappings or procedural query mappings approach are seen as being more appropriate, since these approaches do not require advanced programming skills. However, if more complex types of mappings are required, then scripted mappers can be created. Ideally, such scripted mappers should be developed to apply to a wide variety of situations and contexts, so as to be easily reusable.

For declarative equivalency mappings, a number of tools already exist, and for scripted mappers, query and manipulation languages abound. However, visual tools for procedural query mapping are rare. In addition, this approach is seen as being particularly crucial, since it is not subject to the limitations of the simpler declarative equivalency mapping approach, while at the same time it does not require the more advanced programming skills for the more complex scripted mapping approach. This research therefore specifically focuses on the development of a set of visual tools for developing procedural query mappings.

Data Models for Mapping

It is envisaged that these various tools, parsers, serializers, and mappers could be developed and shared through an online user community. Users could download diverse sets of actors developed by different groups from a shared online repository, and then string these together into customized workflows (e.g., Fig. 1). This process would ideally emerge in a bottom-up manner with minimal restrictions being placed on developers of actors. It is therefore important that no specific data model is imposed, but instead that actor developers and other users are able to choose preferred data models. For a particular pair of tool actors, various parser-serializer pairs may be developed allowing users to choose to generate mappings based on alternative data models. For example, one parser-serializer pair might use a hierarchical XML data model, allowing users to create mappings with their preferred declarative equivalency mapping tool, while another might use a relational data model, allowing users to create a mapping by writing SQL scripts. Ideally, a diverse ecosystem of actors would emerge.

For procedural query mapping, various data models and query languages were considered from the point of view of applicability and ease of use. Below, an example scenario is described in which the property graph data model was used as the mapping data model. A property graph is a directed graph data structure where edges are assigned a direction and a type, and both vertices and edges can have attributes called properties. This allows property graphs to represent complex data structures with many types of relationships between vertices. In graph theoretic language, a property graph is known as a directed, attributed, multi-relational graph. The query language used for querying and manipulating data in the property graphs is called Gremlin [21].

Example Scenario

In order to demonstrate the feasibility of the proposed approach, we have implemented part of the example scenario shown in Fig. 1 using Kepler [22], an open-source workflow system based on an actor-oriented software framework called Ptolemy II [23]. Kepler workflows can be nested, allowing complex workflows to be composed from simpler components, and enabling workflow designers to build reusable, modular sub-workflows that can be saved and used for many different applications.

Kepler is used to create a workflow connecting various tools, including SideFX Houdini as a parametric CAD system to generate a building model and EnergyPlus and Radiance as energy analysis simulation program and lighting analysis program, respectively, to evaluate building performance. Any other (parametric) CAD software and simulation tools could also be considered for this purpose.

A simplified design is used for testing the workflow, consisting of only two spaces stacked on top of each other, each with a window in one wall. The Kepler workflow is shown in Fig. 4; here the actors created wrap the Houdini application, the EnergyPlus program, and the two Radiance programs (using Python as the programming language). In total there are 14 polygons, and each polygon is

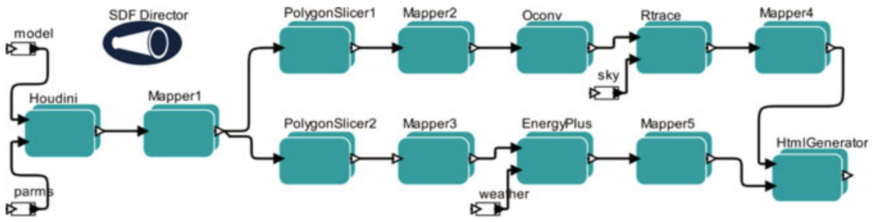
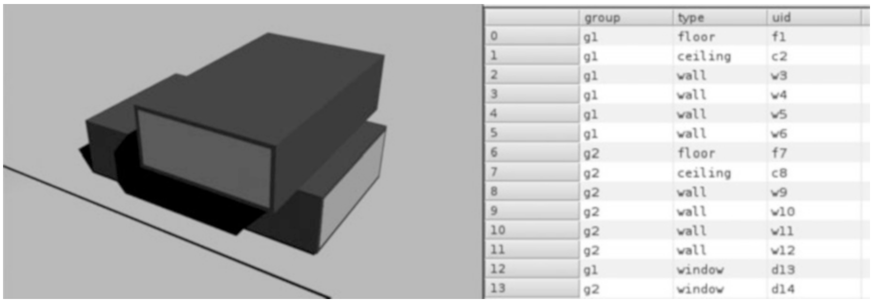


Fig. 4 The Kepler workflow. See Fig. 8 for the contents of the Mapper3 actor



	group	type	uid
0	g1	floor	f1
1	g1	ceiling	c2
2	g1	wall	w3
3	g1	wall	w4
4	g1	wall	w5
5	g1	wall	w6
6	g2	floor	f7
7	g2	ceiling	c8
8	g2	wall	w9
9	g2	wall	w10
10	g2	wall	w11
11	g2	wall	w12
12	g1	window	d13
13	g2	window	d14

Fig. 5 CAD model of 14 polygons, each with three attributes (‘uid’, ‘type’ and ‘group’)

assigned a set of attributes that are used for generating the property graphs. The model is shown in Fig. 5, together with key attributes defined for each polygon.

Workflow Mappers

For all the mappers, property graphs are used as the mapping data model. For Houdini, EnergyPlus, and Radiance, parsers and serializers are created for stepping between the native file formats and the property graphs.

The workflow contains two scripted mappers and five composite mappers containing sub-networks of mapping actors. The two scripted mappers are actually two instances of the PolygonSlicer mapper, the scripted mapper performing the polygon slicing using Boolean operations. This has been implemented in a generalized way to be used for various slicing operations by users who do not have the necessary programming skills to implement such a mapper. In this case, Radiance and EnergyPlus require the polygons to be sliced in different ways. The PolygonSlicer mapper has a set of parameters that allows users to define the polygons to be sliced, and the Boolean operation to be performed. It also allows users to specify certain other constraints, such as the maximum number of points per polygon. Finally, it also identifies adjacency relationships between polygons, for example if two polygons are a mirror image of one another. The input and output files for this mapper use a standard JSON property graph file format, and the parser and serializer using the existing GraphSON library.

The five composite mappers each contain sub-networks that perform a procedural query type of mapping operation. These sub-networks are defined using Kepler mapping actors that provide a set of basic functions for mapping graph vertices and graph edges. Users are able to build complex customized mapping networks by wiring together these basic mapping actors. Each mapping actor has various parameters that allow the mapping function to be customized. When these actors are executed, Gremlin mapping scripts are automatically generated based on these parameter settings.

Three highly generic mapping actors are defined that can be used to create a wide variety of mappings. The Merge actor is used for merging input graphs into a single output graph; the Spawn actor is used for adding new vertices to the input graph; and the Iterate actor is used for iteration over vertices and edges in the input graph while at the same time generating vertices and edges in the output graph. This last actor is powerful and very flexible, as it allows for copying, modifying, or deleting vertices and edges from the input to the output. A parameter called ‘select’ allows users to specify a Gremlin selection filter on the input graph. For each entity (i.e., vertex or edge) in the filtered input graph, a particular action is triggered, which could be the creation or modification of vertices or edges.

A Mapping Example

In order to understand the mapping process, the mapping from Houdini to EnergyPlus will be described in more detail. The first step is for the user to create the parametric model of the design together with a set of parameter values. The Houdini wrapper will trigger Houdini to generate a model instance and will save it as a geometry data file (using Houdini’s .geo format, though other formats could be used too, e.g., .obj).

Two mappers are then applied. The first mapper maps the output from Houdini to the input of the PolygonSlicer, and the second mapper maps the output of the PolygonSlicer to the input of EnergyPlus. For the first mapper, a parser is provided for stepping from the Houdini geometry file to the property graph, and a serializer is provided for stepping from the property graph to the JSON graph file. For the second mapper, a parser is provided for stepping from the JSON graph file to the property graph, and a serializer is provided for stepping from the property graph to an EnergyPlus input file (using EnergyPlus’ .idf format). As already mentioned above, these parsers and serializers just mirror between the data file and the data model, and as a result they can be implemented in a way that is highly generic. Although implementing these parsers and serializers will require someone with programming skills, it needs to be done only once, after which end-users can simply select the required parsers and serializers from a library. Implementing the parser and serializer for the JSON graph files is trivial since a library already exists. For Houdini and EnergyPlus, the ASCII data files have a clear and simple structure, resulting in a straightforward implementation for the parser and serializer.

Given a library of parsers and serializers, the task for the end-user is then reduced to the transformation of the input property graph into the output property graph using the three Kepler mapping actors. In anticipation of this mapping process, the user can define additional attributes in the geometry model that can be used during the mapping. In this scenario, the user knows that in order to map to EnergyPlus, surfaces will need to be assigned different types and will also need to be grouped into zones. In this case, the polygons in the parametric model are each assigned three attributes: a ‘uid’ attribute is used to define a unique name for each

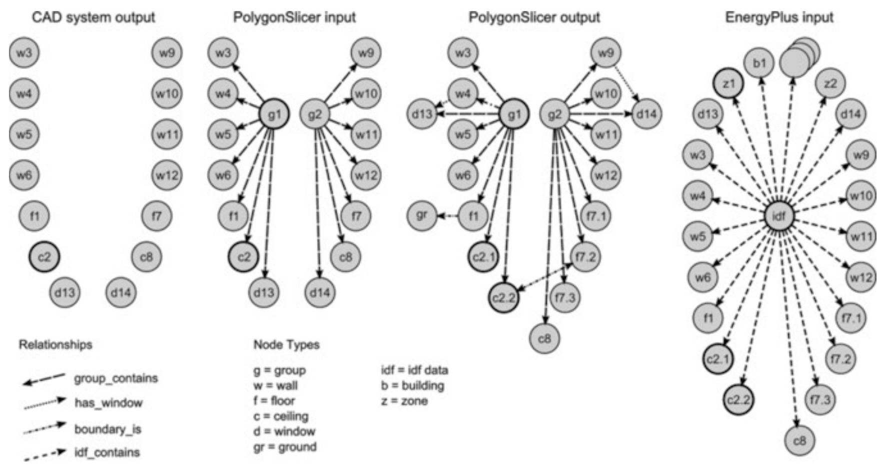


Fig. 6 Simplified diagrammatic representation of the property graphs for (Point data is not shown in order to reduce the complexity of the diagrams. Before the PolygonSlicer there are 24 points, while afterwards there are 28 points)

polygon, a ‘type’ attribute is used to define the type for each polygon, and a ‘group’ attribute is used to define the group to which the surface belongs, with groups corresponding to zones (see Fig. 5). When the parser reads the geometry data file, it will convert these attributes into properties, so that a polygon with attributes in the geometry file will become a graph vertex with properties in the property graph.

The user then needs to create the graph mappers using the graph mapping nodes. Figure 6 shows the overall structure of these input and output property graphs, and Fig. 7 shows the properties associated with three of the vertices in each property graph. The PolygonSlicer and the EnergyPlus actors both have input graph schemas that specify the required structure of the graph and the required properties of the vertices. The task for the user is therefore to create mappings that generate graphs that adhere to these schema constraints.

In the first mapping, where the output of Houdini is mapped to the input of the PolygonSlicer, two new vertices are added for the two groups, and edges are added from the new group vertices to the polygon vertices. These vertices and edges are created using an Iterate actor. The PolygonSlicer then transforms its input graph by dividing the surfaces for the ceiling of the lower zone (‘c2’) and the floor of the upper zone (‘f7’) so as to ensure that each surface has a homogeneous boundary condition. The PolygonSlicer also detects the relationships between the floors and ceilings, between the floors and the ground, and between windows and walls.

In the second mapping, where the output of the PolygonSlicer is mapped to the input of the EnergyPlus simulator, additional properties are added to the existing vertices in the input graph, and a number of additional vertices are also added to define a set of other objects required in the EnergyPlus input file. The Kepler network for this mapper is shown in Fig. 8. The Spawn actor is used to create the additional vertices, and Iterate actors are used to copy and modify existing vertices. The groups

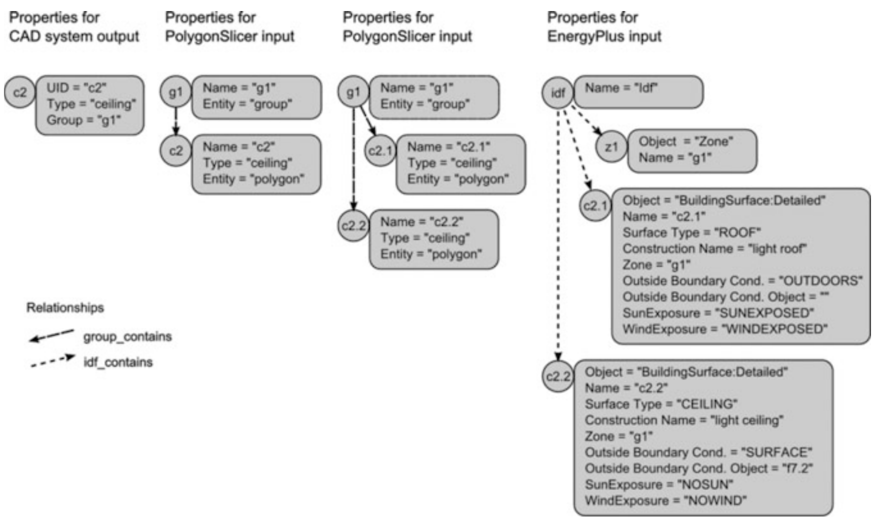


Fig. 7 An example of the property data for a few of the vertices in the property graphs. Typically, the property graphs will undergo a process of information expansion, where data is gradually added to the model as needed

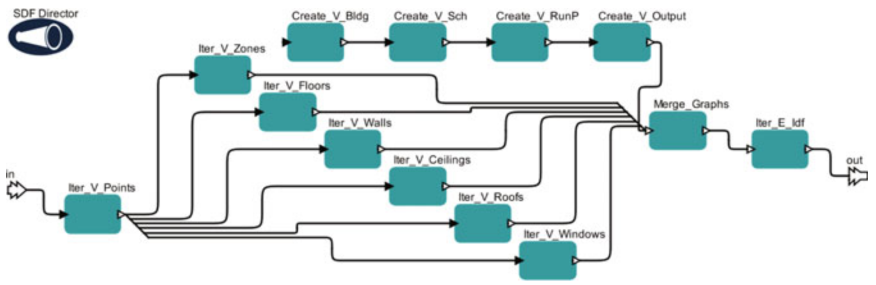


Fig. 8 The Kepler mapper that maps the output of the PolygonSlicer actor to the input of the EnergyPlus actor. See the 'Mapper3' actor in Fig. 5

are mapped to EnergyPlus zones, and the polygons to EnergyPlus surfaces. In the process of mapping, the Iterate actor also transforms the edges that existed in the input graph into properties in the output graph. The output graph becomes a simple list of vertices under the 'idf' root vertex. For example, in the input graph the window is connected to the wall with an edge, while in the output graph the window is no longer connected but instead has a property that specifies the wall name.

For each different surface type, a separate Iterate actor is defined. For example, consider the 'Iter_V_Ceilings' actor in Fig. 8. This node generates the ceilings of the two zones. Table 1 shows the two main parameters for the actor. The 'Select' parameter filters the input graph so that the remaining vertices all have an 'Entity' property with a value of 'polygon' and a 'Type' property with a value of 'ceiling', and in addition have an outgoing 'boundary_is' edge that points to another polygon

Table 1 The parameter names and values for the Iter_V_Ceilings actor. Gremlin code is shown in italics, and makes use of two predefined local variables: ‘g’ refers to the input graph, and ‘x’ refers to the entity being iterated over (which in this case is a vertex)

Parameter	Parameter value
Select	*g.V.has('Entity','polygon').has('Type','ceiling')*
	.as('result').out('boundary_is')
	.has('Entity','polygon').back('result')
Vertex properties	`Object : 'BuildingSurface:Detailed'`
	`Name :` *x.Name*
	`Surface_Type : 'CEILING'`
	`Construction_Name : 'light ceiling'`
	`Zone :` *x.in('group_contains').Name*
	`Outside_Boundary_Cond : 'SURFACE'`
	`Outside_Boundary_Cond_Object :` *x.out('boundary_is').Name*
	`Sun_Exposure : 'NOSUN'`
	`Wind_Exposure : 'NOWIND'`
	`Points :` *x.Points*

(i.e., the floor above). The ‘Vertex properties’ parameter then defines a set of name-value property pairs. For each polygon in the filtered input graph, the iterator actor will generate a vertex in the output graph with the specified properties.

Note that when the user is specifying the property values, they can insert Gremlin commands to extract these values from the input graph, thus ensuring that the values can be dynamically generated. Figure 6 shows changes for a number of vertices in the property graph as data is mapped and transformed. When the ‘Iter_V_Ceilings’ actor iterates over the ‘c2.2’ polygon in the input graph, it generates the ‘c2.2’ EnergyPlus surface.

Adaptive-Iterative Exploration

Once a workflow has been developed, it can be used to iteratively explore design variants. The example workflow can be used without modification to evaluate any design variants that consist of a set of stacked zones. If other spatial configurations of zones need to be evaluated not limited to stacking, then the workflow may need to be adjusted in minor ways. For example, if zones are configurations so that they are adjacent to one another, then the ‘Iter_V_Walls’ actor in Fig. 8 would be modified to allow common boundary walls between zones.

Conclusions

In order to support a bottom-up, user-controlled and process-oriented approach to linking design and analysis tools, a data mapping approach is required that allows designers to create and share custom mappings. To achieve this goal, the data mapping approach should be both flexible in that it can be applied to a wide variety of tools, and user-friendly in that it supports non-programmers in the process of easily creating and debugging mappers. The use of common data models simplifies the process for end-users to develop customized mappings. The example scenario demonstrated how designers with minimal scripting skills would be able to set up complex digital workflows that enable the fluid and interactive exploration of design possibilities in response to custom performance metrics.

The next stage of this research will explore the scalability of the user-defined graph mapping approach when working with larger data sets and more complex data schemas (such as the IFC schema). In the current demonstration, the data sets and data schemas are small, and as a result the graph mappers are relatively simple. However, if data sets grow and the number of entity and relationship types is very large, then the graph mappers could potentially become more difficult to construct. In order to deal with this increased complexity, we foresee that the user will require additional data management and schema management tools. The data management tools could enable users to visualize, interrogate and debug property graph data during the mapping process [24]. Schema management tools could let actor developers define formal graph schemas for input and output data for their actors. This could then let end-users identify and isolate subsets of large schemas relevant to their particular design scenario.

Acknowledgments The approach proposed in this research emerged out of discussions at the "Open Systems and Methods for Collaborative BEM (Building Environment Modeling)" workshops held at the CAAD Futures Conference in July 2011 and at the eCAADe Conference in 2013. We would like to thank the participants of these workshops for their contributions. Additionally, we would like to thank the reviewers for their valuable comments.

References

1. Coons SA (1963) An outline of the requirements for a computer-aided design system. In: Proceedings of the AFIPS, ACM, pp 299–304
2. Sutherland I (1963) SketchPad: a man-made graphical communication system. In: Proceedings of the Spring Joint Computer Conference, Detroit, pp 329–346
3. Janssen P, Stouffs R, Chaszar A, Boeykens S, Toth B (2012) Data transformations in custom digital workflows: property graphs as a data model for user defined mappings. In: Proceedings of the Intelligent Computing in Engineering Conference ICE2012, Munich
4. Toth B, Janssen P, Stouffs R, Boeykens S, Chaszar A (2013) Custom digital workflows: a new framework for design analysis integration. Int J Archit Comput 10(4):481–500
5. Deelman E, Gannon D, Shields M, Taylor I (2008) Workflows and e-science: an overview of workflow system features and capabilities. Futur Gener Comput Syst 25:528–540
6. Yu J, Buyya R (2005) A taxonomy of scientific workflow systems for grid computing. SIGMOD Rec 34:44–49
7. Curcin V, Ghanem M (2008) Scientific workflow systems – can one size fit all? In: CIBEC. Cairo, pp 1–9
8. McPhillips T, Bowers S, Zinn D, Ludaescher B (2009) Scientific workflow design for mere mortals. Futur Gener Comput Syst 25:541–551

9. Janssen P, Chen KW (2011) Visual dataflow modelling: a comparison of three systems. In: Proceedings of the 14th International Conference on Computer Aided Architectural Design Futures, Liege, Belgium, pp 801–816
10. Bowers S, Ludäscher B (2005) Actor-oriented design of scientific workflows. In: Proceedings of the International Conference on Conceptual Modeling ER, pp 369–384
11. Mirtschin J (2011). Engaging generative BIM workflows. In: Proceedings of LSAA 2011 Conference Collaborative Design of Lightweight Structures, Sydney
12. Tsichritzis DC, Lochovsky FH (1982) Data models. Prentice-Hall, Englewood Cliffs
13. Eastman CM, Teichholz P, Sacks R, Liston K (2011) BIM handbook – a guide to building information modeling for owners, managers, designers, engineers, and contractors, 2nd edn. Wiley, Hoboken
14. Kilian A (2011) Design innovation through constraint modeling. Int J Archit Comput 4:87–105
15. Altintas I (2011) Collaborative provenance for workflow-driven science and engineering. PhD thesis, University of Amsterdam, Amsterdam
16. Bos P (2012) Collaborative engineering with IFC: new insights and technology. In: Gudnason G, Scherer R (eds) eWork and eBusiness in architecture, engineering and construction, 9th ECPPM Conference Proceedings, pp 811–818
17. O'Donnell J, See R, Rose C, Maile T, Bazjanac V, Haves P (2011) SimModel: a domain data model for whole building energy simulation. In: Proceedings of the 12th Conference of International Building Performance Simulation Association, Sydney
18. Srinivasan, R, Kibert C, Thakur S, Ahmed I, Fishwick P, Ezzell Z, Lakshmanan J (2012) Preliminary research in dynamic-BIM (D-BIM) workbench development. In: Proceedings of the 2012 Winter Simulation Conference (WSC), pp 1–12
19. Kannengiesser U (2007) Agent-based interoperability without product model standards. Comput Aided Civ Infrastruct Eng 22(2):96–114
20. Bellahsene Z, Bonifati A, Duchateau F, Velegrakis Y (2011) On evaluating schema matching and mapping. In: Bellahsene Z, Bonifati A, Rahm E (eds) Schema matching and mapping. Springer, Berlin, pp 253–291
21. Gremlin (2014) Retrieved 1 Apr 2014 from https://github.com/tinkerpop/gremlin/wiki
22. Ludascher B, Altintas I, Berkley C, Higgins D, Jaeger E, Jones M, Lee A, Tao J, Zhao Y (2006) Scientific workflow management and the Kepler system. Concurr Comput Pract Experience 18 (10):1039–1065
23. Eker J, Janneck J, Lee EA, Liu J, Liu X, Ludvig J, Sachs S, Xiong Y (2003) Taming heterogeneity: the Ptolemy approach. Proc IEEE 91(1):127–144
24. Stouffs R (2001) Visualizing information structures and its impact on project teams: an information architecture for the virtual AEC company. Build Res Inf 29(3):218–232

Part VIII
Design Processes – 2

Application of Function-Behaviour-Structure Variables for Layout Design

Prasad Bokil

Abstract The field of graphic design, to a large extent, lacks an objective articulation and proper knowledge representation. This paper articulates the process of layout design. It represents the knowledge of layout designing with the function-behaviour-structure path. The variables of layout design process are identified within the same framework. These variables are then implemented to analyse the use of grids in graphic design with various degrees of innovation and creativity. The paper demonstrates how the knowledge of these variables empowers the designer to create a variety of interesting layouts. The work presented here aims to understand and articulate the layout design process with better objectivity. It contributes to bridge the gap between design practice and design as a knowledge domain.

Introduction

Layout design is the process of arrangement of graphical units within a two/three dimensional space. Layout design is an inseparable aspect of any graphic design project. It is also crucial for many other domains like exhibition design, interface design, interior design, etc. Layout design has to deal simultaneously with aesthetics and usability. In the field of graphic design it is mostly considered as an intuitive process and is rarely studied with a scientific approach. Yet, there is definitely a systematization of layout design process through the use of grids [1–3]. Grid is a well-known tool in graphic design. It can be defined as '**an understructure used to discretize any continuum**'.

In the previous work [4] the process of layout design was articulated and represented by FBS framework. The layout design was analysed as a two-level process where the designer was treated as user. This is consistent with Gero and Kannengiesser's approach for representational affordances in design [5]. Based on the Function-Behaviour-Structure ontology developed to represent designed objects [6–8], the process of layout design with grid was schematically represented

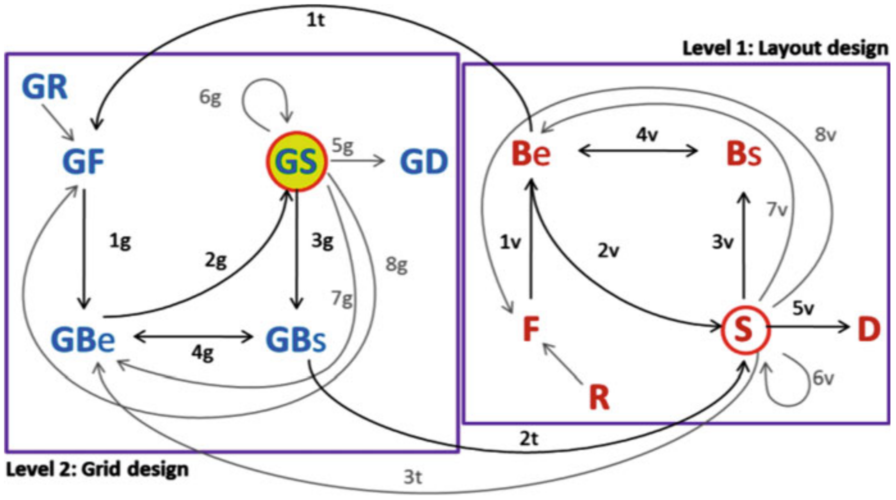
P. Bokil (✉)
Indian Institute of Technology, Guwahati, India
e-mail: prasad.bokil@iitg.ernet.in

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_30

by FBS path of grid nested in FBS path of layout design as shown in Fig. 1. The paper was concluded with the discussion on the possible advantages of knowledge representation of grids in graphic design. It was claimed that this articulation will help in understanding the practice and exploring new ways of using grids.

This paper will try to answer the next research question, that is, 'How will the knowledge representation support the understanding of application of grids and its creative use in visual design?' This work will demonstrate a way in which this knowledge representation can contribute to understanding of visual design. The hypothesis is – 'Function-Behaviour-Structure ontology is sufficient to articulate the process of graphic design and it is effective to understand the design decisions made by the designer'. This study focuses on the process to achieve flexibility in application of grids.

Grids are blamed for making the design look boxy and boring. It is unanimously accepted in the design community that the grid, if not used creatively, may lead to uninteresting and ineffective visuals. It is also claimed that the grid is extremely difficult to use in the service of invention [9, p. 160]. However, the author believes that this problem is rooted in the lack of proper knowledge representation. Without profound representation, it is difficult to provide the means for 'creative use' of grid. Design research has separated creative and innovative design from routine one



Where,
Be = expected behavior, **Bs** = behavior derived from structure, **D** = design description, **F** = function, **S** = structure, **R** = design requirements, **GBe** = expected behaviour of grid, **GBs** = grid behavior derived from structure, **GD** = grid description, **GF** = function of grid, **GS** = grid structure, **GR** = grid requirements
There are eight processes involved in each FBS path, which are formulation (1), synthesis (2), analysis (3), evaluation (4), documentation (5) and three reformulation processes (6, 7, 8). There are three transition processes across the FBS paths (1t, 2t & 3t).

Fig. 1 Nested FBS path for layout design with grids

[10]. Applying the same principles for grid, if it is defined with appropriate variables, the change in the value of such variables will result in the innovative use of grid; and change in the variables will give creative use of grid. There are a few techniques that designers generally prescribe to make grids interesting. This paper will investigate such techniques and try to articulate those in accordance to the previously published discourse.

Identification of FBS Variables of Grid

The representation of any concept by its variables enables the use of those variables to analyse the variations of the concept in real and hypothetical situations. In layout design process, visual qualities and usability aspects are discussed by designers; but their correlation with grid is hardly given an importance. Here the variables of grids are discussed to make them useful for analysis of grids and the visuals created using those grids. These variables are categorized into three groups – function variables, structure variables and behaviour variables. Different variables under these three categories are identified based on the influential work by Qian and Gero [11] on FBS variables.

Structure Variables of Grid

The grid structure can be defined by three finite sets of elements, attributes and relationships. These three sets are enough to create the structure of grid. These three sets as represented by Eq. 1 are sufficient to articulate the description of grid structure.

$$\mathrm{GS} = \{\mathrm{E}, \mathrm{A}, \mathrm{R}\} \tag{1}$$

Where,

E = a finite set of element variables (points, lines, curves and shapes);
A = a finite set of attribute variables (coordinates, curve equations, etc.);
R = a finite set of relation variables.

Set R gives the relationship among the elements, for example, the distance among two points, the angles between two lines, linearity, intersection etc. Many times, set R is the emergent set of E and A. Thus the relationship among the elements can be derived from their attributes and there is no separate need to define R.

The role of grid ends once the visual is designed and it is usually removed. It becomes difficult to use structure variables of grid for the analysis of the visual if the grid is not available.

Function Variables of Grid

The grid is created and used by designers. Grid assists designer by creating a finite set of values to choose from, for structural variables of visual design. It helps the designer to create a visual with an expected effect which is the result of structural attributes and behaviour of the visual. As per the FBS model (Fig. 1) developed for the grids in layout design, the function of the grid is derived from the expected behaviour of the visual to be designed. Hence the function descriptions of grid can be obtained from the expected effect in designed visual.

The effect of grid on visual behaviour contributes to improve the visual quality and communication. The variables dealing with visual quality are order, consistency, harmony, etc.; and the variables dealing with communication are cognitive load, quantity of information, etc. Function variables are the means by which the behaviour of visual can be connected to grid behaviour and ultimately to the grid structure.

The function of grid as well as visual is subjective to designers' capability and viewers' perception. Use of the same grid by different designers in a similar context may result in very different functions. It is difficult to articulate the function in terms of quantifiable variables. It is difficult to rely on function variables of grid for objective visual analysis.

Behaviour Variables of Grid

The function is difficult to articulate objectively and structure can have infinite variations. But the behaviour of grid has few standard variables which are able to define the grid. Behaviour is useful in design process for problem formulation, synthesis, analysis, evaluation and reformulation [7]. The way behaviour is defined in FBS framework, it is not dependent on any action of structure but derived from just the existence of the structure. The behaviour of grid is static spatial behaviour for static visuals. As shown in the FBS path the behaviour of grid contributes to the structure of visual hence the behaviour variables of grid play very important role in the design process.

Behaviour variables of grid are of two types. Structural type **direct behaviour variables** (BV_d) are derived from the structure itself without any external effect. After analysing various visuals [12], five basic structural type behaviour variables are identified. These are- (i) Position, (ii) Orientation, (iii) Area, (iv) Proportion and (v) Division. Given a structure of a grid, some or all of these behaviour variables attain a finite set of values, which help the designer to take appropriate design decisions. It should be noted that in design many of these variables are fixed simultaneously by the designer while creating a layout. When applied with the content of design, each of these direct behaviour variables, alone or in combination, gives an **exogenous** (indirect) **behaviour variable**. Exogenous behaviour variable emerges when external object (in this case 'the content') is applied to the structure of grid. The exogenous type variables thus derived are- stamp (semantic unit), direction, plan, perspective and sequence. The detailed discussion on these variables with illustrative examples can be found elsewhere [12, 13].

With this set of direct and exogenous behaviour variables, the behaviour space of grid can be defined as-

$$B = \{X, \ \Theta, \ \Omega, \ P, \ \Delta, \ BV_e\}$$

Where,

B = Behaviour space of grid
X = set of values of position variable
Θ = set of values of orientation variable
Ω = set of values of area variable (combination of width and height)
P = set of values of proportion variable
Δ = set of values of division variable
BV_e = set of values of exogenous behaviour variables (stamp, direction, plan, perspective and sequence)

Although the fear of getting trapped in the grid is expressed by many designers, the suggested solution is always in the form of subjective and vague description. The creative use of grid is treated like an art which can be observed and needs to be mastered through the practice. With the help of FBS path and FBS variables it is possible to develop a better representation of the current state of art in graphic design. It will be possible to use this representation for efficient knowledge transfer and improvement of the design process. The application of these defined variables with FBS path for the flexibility in layout design is discussed in the rest of the paper.

Use of FBS Path and FBS Variables to Understand Flexibility

The FBS variables can be used in both descriptive study and prescriptive study. They can be used to articulate the existing process of design and to analyse existing visuals. With the understanding built on the descriptive study, these variables can later be used to propose the method to employ grids in creative design and innovative designs.

To understand the entrapment experienced by the designers and the rigidity induced by the grid in design process, the FBS path and FBS variables can be used as means for analysis. With the understanding of FBS variables of grid and visual the basic concepts are defined as follows:

Entrapment of grid: Entrapment of grid is the fixation of multiple visual elements or series of visual structures to constant values of behaviour variables of the same grid in successive occurrences.

Flexibility of grid: Flexibility of grid is the potential of grid to provide multiple values of behaviour variables to choose from for deciding the structural values of visual. 'Making the grid flexible' means empowering the grid with a set of multiple values for behaviour variables and keeping the scope to add new variables.

Breaking the grid: Breaking the grid means allowing any structure variable of visual to attain a value other than the values provided by behaviour variables of grid. Breaking the grid means introducing a new variable directly to the visual level.

Creative Use of Grid and Flexibility Matrix

The classification of design into routine, innovative and creative is well accepted in design research [7, 14, 15]. If the design variables and the ranges of values they can take remain fixed during design processing, the design is routine; if the design variables remain fixed but the ranges of values change, the design is innovative; and if the design variables and the range of values both change, the design is said to be creative. Creative use of grid can be explained as- 'The use of grid can be called as creative when a new structure variable/s of grid is/are introduced or a dormant behaviour variable is activated during the design process. Sometimes, breaking the grid also helps.' The key to the flexibility in design process lies in the understanding the FBS path. There are two important facts to be noticed while discussing the creative use of grids. One is the non-mandatory position of grid FBS path to reach to a final visual structure. A visual structure can be derived independent of grid from expected behaviour emerged from the function of visual. The grid is an additional help to induce an order and consistency to reduce the working time. By definition it is the nature of the grid to put restrictions on the values of visual structure variables. Grid controls the amount of layout information. The knowledge of variables can certainly be used to understand the methods to bring flexibility in design.

Another aspect is the subjectivity involved through designers' interventions in between the grid FBS path and visual FBS path. The flexibility in grid may or may not result in creative/innovative layouts. Making the grid flexible, innovative or creative does not ensure the same result in visual layout. The designer who is using the grid to create visuals is an important and subjective agent in the process. With distinction of routine, innovative and creative design based on variables, a matrix can be generated as shown in Table 1. This matrix represents the categorization of all possible scenarios.

Entrapment of grid is the rigidity in design resulted due to limited values provided by grid. The knowledge of FBS variables can help to understand this entrapment created by grid. Grid provides the values for structure variables of visual design. The design becomes rigid when each structure variable of the visual

Table 1 Flexibility matrix of visual design based on grids

➦	Routine design: structure variables of visual are fixed to constant values	Innovative design: str. variables of visual can take multiple values	Creative design: new str. variables are introduced during design
Routine grids: all grid variables are fixed to constant values	GSV = C	GSV = C	GSV = C
	GBV = C	GBV = C (but uneven/asymmetric)	GBV = C (may be uneven/asymmetric)
	GBV$\rightarrow$VSV	GBV$\rightarrow$VSV	GBV$\rightarrow$VSV
	VSV = C	VSV $\neq$ C	New structure elements are introduced by breaking the grid VSV $\neq$ C
Innovative grids: keeping the structure same the grid behaviour variables choose from set of multiple values	GSV = C	GSV = C	GSV = C
	GBV $\neq$ C	GBV $\neq$ C. May have set of multiple values	GBV$\neq$ C. May have set of multiple values
	GBV$\rightarrow$VSV	GBV$\rightarrow$VSV	GBV$\rightarrow$VSV
	VSV = C due to repeated selection of same values of GBV	VSV $\neq$ C. Selection of various combination of values within and across the visual structures	VSV $\neq$ C. Selection of different combination of values within and across the visual structures
			Also, new structure elements are introduced by breaking the grid
Creative grids: new grid variables are introduced during design process	GSV $\neq$ C	GSV $\neq$ C	GSV $\neq$ C new structural elements of grid are introduced
	GBV $\neq$ C	GBV $\neq$ C	GBV $\neq$ C. May have set of multiple values
	But VSV = C	GSVs may take new values and so affecting grid behaviour variables	GBV$\rightarrow$VSV
	Visual is not making any use of newly introduced grid variables	GBVs may have set of multiple values	VSV $\neq$ C. Selection of different combination of values within and across the visual structures
		GBV$\rightarrow$VSV	Also, new structure elements are introduced by breaking the grid
		VSV $\neq$ C. Selection of different combination of values within and across the visual structures	

Where, *GSV* grid structure variables, *GBV* grid behaviour variables, *VSV* visual structure variables, *C* predefined constant value is restricted by a single value of grid behaviour variable. There can be two reasons for this rigidity. Either the grid designed by the designer does not provide variety of values for grid behaviour or the designer repeatedly chooses the same value from the range of values provided by the grid.

The key to all the problems mentioned above lies in the control over the grid to make it more flexible. The only solution presented is applying it creatively or breaking the grid. Both methods are assumed to be intuitive and have never been questioned for the proper process. To avoid the entrapment of grid and to create interesting layouts, the grids should be used with more flexibility. Here, a new method to bring flexibility is not claimed but few systematic methods to achieve the flexibility are articulated. In general practice, a grid like modular grid, which gives multiple options, is used to bring flexibility in grid [2, 16]. Expert designers sometimes use multiple grids overlapping each other for further variety in layouts [1, 17]. If it is still not sufficient to break away from the monotonous structure it is suggested to break the grid occasionally to get interesting results [18]. A systematic way is suggested here to understand this making and breaking of grid.

In terms of variables, it can be stated that- ‘Following grid as it is’ is a rigid, routine, non-interesting way. If behaviour variables attain multiple values during design then it is an innovative use of grid and if new behaviour variables are activated or new structure variables are introduced during design then it is a creative use of grids in design. These concepts of innovative and creative use of grids are elaborated below.

For creative layouts, the behaviour variables are activated and altered during the process of design. When these variables change their values during the design process it results in interesting layouts. For a design application like book design, sometimes no flexibility is required and a routine design with fixed values of area and position variables is commonly used. But in applications like magazine cover or poster, it is necessary to use grids creatively for creating attractive design.

The cover pages and the posters are needed to be created with uneven or surprising values of behaviour variables. Many visual elements either break the grid or have contrast of values with respect to remaining layout. The real challenge for using grid is in the layout with lots of information like newspapers or magazine spreads. The need for grid is severe in such applications, Fig. 2. Newspapers have lots of content and very little time to design hence the most common variables which are varied are proportion and area. The sports pages and cultural pages show lots of other variables in play. The layouts on these pages sometimes even break the grid.

Innovative Use of Grids Through Multiple Values of Grid Variables

Creating innovative layouts is an everyday battle for graphic designers. The discussion on FBS path of layout design mentioned the dependency of structure variables of visual on behaviour variables of grid. Although grid restricts the values attainable by visual structure, it has the potential to provide a set of values to choose from. The matrix shown in Table 1 categorises these potential possibilities to induce the flexibility by changing the values of variables or by introducing new variables. In simple cases like a novel, the only grid present is the text boundary. The font size, leading and all other typographic elements are already set to the fixed values. The structure remains the same throughout the novel. Here grid provides only one fixed value. It gives maximum rigidity to the page layout, and is an example of a routine design created with a routine grid.

Many times, the grid presents inbuilt potential to bring flexibility in visual structure by providing multiple values for each structure variable. Such potential should be used effectively to create more interest through the layouts. This potential to attain multiple values to choose from gives an opportunity to the designer to extend the design to the innovative space from routine space. Figure 3 shows variety of ways an image and the caption can be arranged on 5 × 4 modular grid. Here, area and proportion are the two behaviour variables of grid which had given

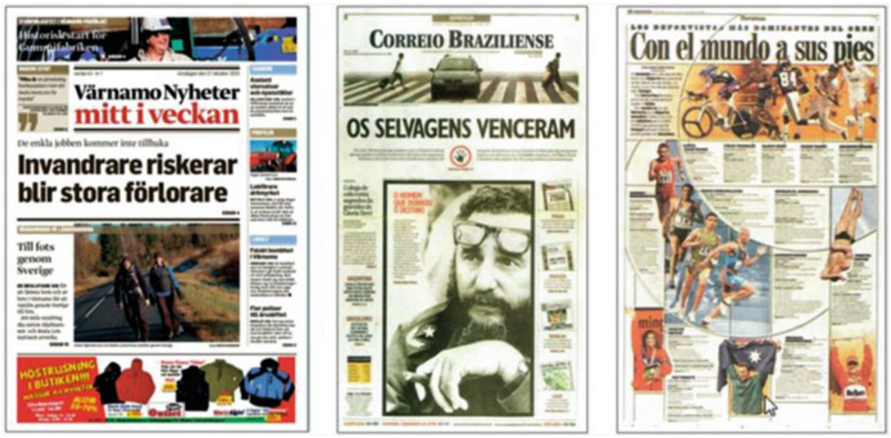


Fig. 2 Variations in grid by changing values of behaviour variables

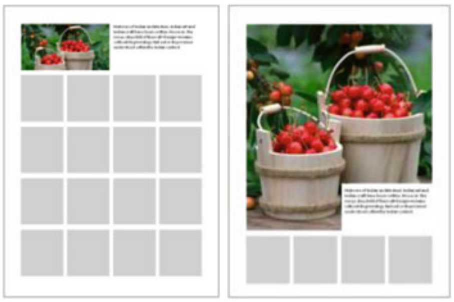


Fig. 3 Flexibility through variable values provided by grid

different values to the image. The area of the photograph can be represented in each case as follows-

$A = A_{11} + A_{12} + GS$
$A = A_{11} + A_{12} + A_{13} + A_{21} + A_{22} + A_{23} + GS$
$A = A_{11} + A_{12} + A_{13} + A_{14} + A_{21} + A_{22} + A_{23} + A_{24} + A_{31} + A_{32} + A_{33} + A_{34} + GS$
…

Therefore,

$$A = \Sigma A_{ij} + GS$$

Where, i and j are the number of columns and rows occupied by an image and GS is the gutter space under image (which can be calculated).

Therefore, the structural variable (size of image) can take any value from a set provided by behaviour variable of grid (grid area). The variables can take a negative value as well (last image in Fig. 3) which allows the overlapping or cut out of visual elements.

What we are discussing here is the variation in visual structure based on the same grid. The grid is already designed and fixed. The examination of the FBS path can throw more light on this innovative design with multiple choices. Derived from the original FBS path this process acquires the new path as shown in Fig. 4. In every design event based on grid GS, a variation of visual structure S_n can be generated using different values provided by GBs. More the number of values provided by GBs greater the flexibility of S_n. The number 'n' of different possible structures can be created out of GS by processes of reformulation 1t, 3t and 2t. With the information (1t) of expected behaviour of visual (Be) and feedback (3t) from

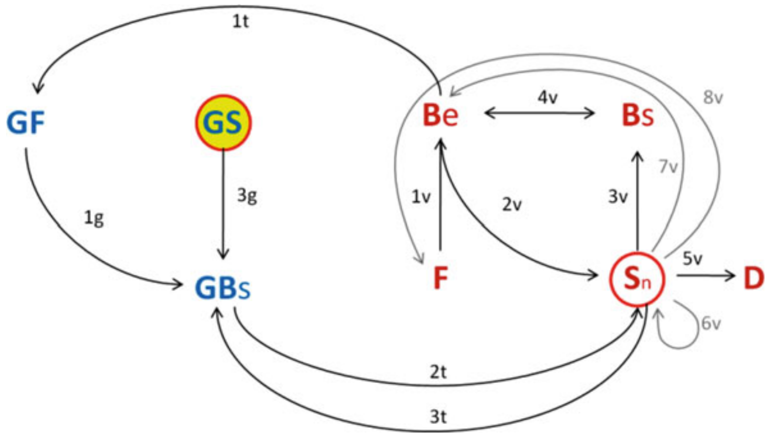


Fig. 4 FBS path to generate various visual structures based on same grid (n = 1,2,3,…)

already tested structures up to n – 1 the values of grid behaviour variables are chosen. And with these values a new structure S_n is created.

In the grid shown in Fig. 5, there are 72 (3 × 6 × 4) positions created by the grid to place the visual content. There are two orientations and 13 different proportions possible. And there are 126 (6! × 3!) combinations of areas which can be created. As Fig. 5 has shown few variations in proportion, any one variable or many variables in combination can change their values to give variety of layouts from the same grid. One can find many such demonstrations like [1, 19] in design literature.

The grid (GS) itself gives the scope to attain a range of values for behaviour variables without changing the structure of grid. Within this scope **the flexibility of layout can be increased by having uneven values for variables of grid**. The grid in Fig. 5 has an equi-modular structure. All the spaces created are of constant area and proportion. This gives more consistency to the visual but it reduces the information and so the flexibility. Making the grid with uneven values increases the visual information. The grid variables can attain different values either keeping the symmetry or breaking it (as shown in Fig. 6). Introducing the asymmetry makes the grid less obvious and thus more interesting.

As the number of structure elements of a grid increases the number of values attainable by each behaviour variable increases; thus increases the flexibility. But increase in the number of structural elements makes the grid complicated. Also, a dense grid gives more predictable patterns. A well-known method used by graphic designers to overcome these problems is overlapping grids on top of each other [2, 17]. In this method more than one grid are designed and placed in the same space.

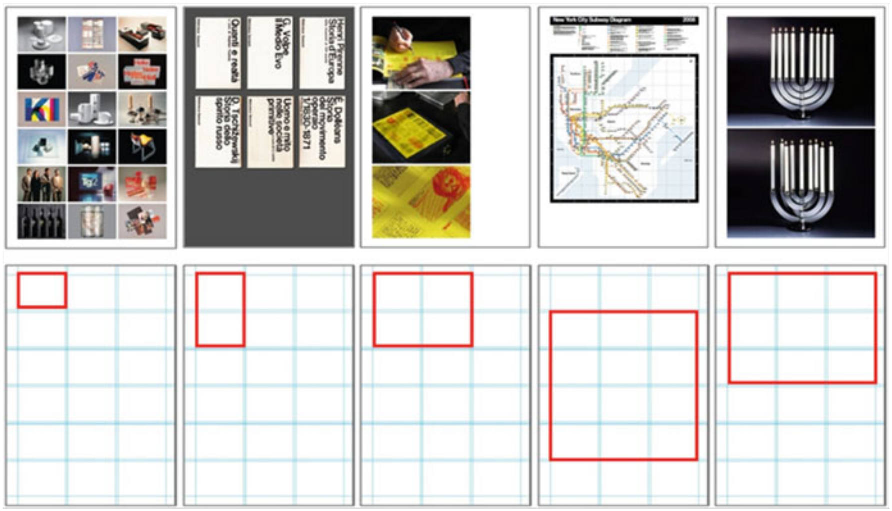


Fig. 5 Proportion variable attaining various values (1 × 1, 1 × 2, 2 × 2, 3 × 4, 3 × 3)

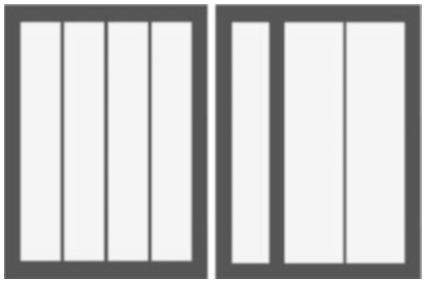


Fig. 6 Column grids with various values of column width in symmetrical and asymmetrical arrangements

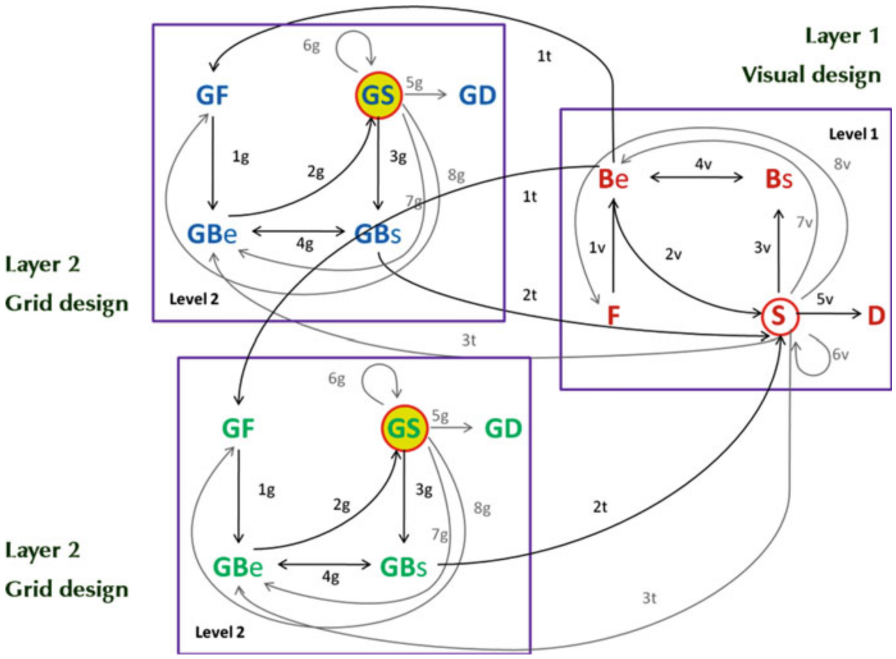


Fig. 7 FBS paths of two grids nested simultaneously

The choice between these grids is driven by aesthetic judgement of designer. Each grid offers the set of values for behaviour variables which enables a designer to choose a value from a wider range with different patterns. Sometimes, the choice between these grids is based on the type of visual elements. For example, in a layout design with text and images, all images can be set according to one grid and the text can be set on another grid. In this case, multiple Grid FBS paths are nested in FBS path of visual. This multiple nesting is shown in Fig. 7.

Creative Use of Grids by Activating New Behaviour Variables of Grid

The flexibility achieved by selecting various values provided by the grid is discussed above. As the grid has active behaviour variables with multiple values there are also certain dormant variables which retain constant values during the design. Sometimes, these dormant variables can be activated to bring flexibility. The values of variables can be changed by changing the grid either before or after creating the layout. In most of the publication design, as the demand of readability, the orientation variable becomes dormant. The only values permitted are horizontal text flow and vertical columns. This behaviour variable if activated can create creative possibilities to make interesting layouts. Either the orientation of designed grid is changed to new value or a visual is created and the complete visual is changed to new value of orientation.

For example, two posters and a magazine cover page in Fig. 8 are designed as per the grid and then the orientation variable of complete grid with the layout is changed to a new value. The value of orientation is changed from its routine vertical-horizontal position. This change has created a little imbalance in the layout and makes it interesting. The change in the orientation can be clockwise or anticlockwise with any value decided by the designer during the design.

There is another way to activate the behaviour variable during the design. As discussed in the last example the complete grid is changed according to the new value of an activated variable. Instead of changing the complete grid a part of grid can be subjected to the transformation by activating any one or more behaviour variables. It is demonstrated in Fig. 9. The original 3 × 4 modular grid is subjected to position, proportion and orientation transformation by activating the behaviour variable.

Creative Use of Grids by Inserting New Set of Variables

The predictability of grid can be broken and it can be made more interesting by adding new variables independent of the existing basic grid. The structure elements

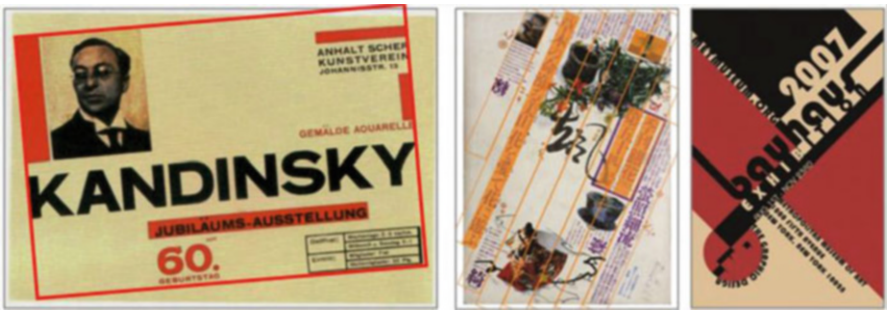


Fig. 8 Visual designed with grid and then the orientation is changed to new value

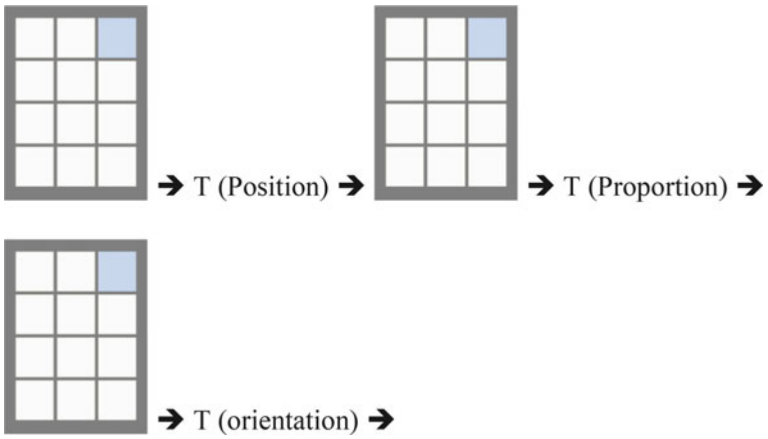


Fig. 9 Original grid→ behaviour transformations→ applied to part of the grid; applied to complete grid

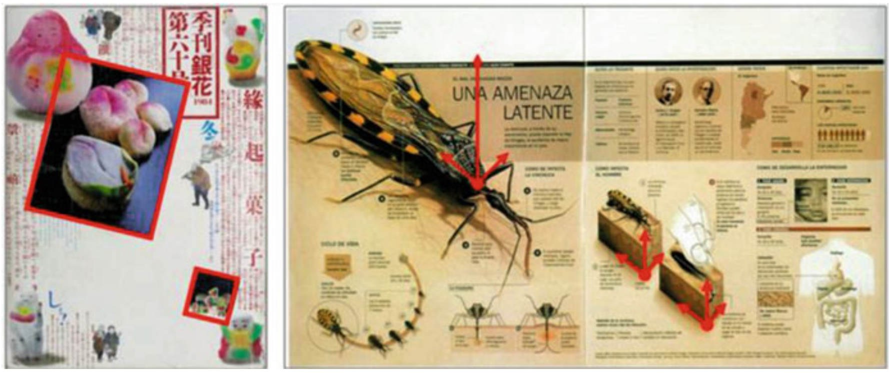


Fig. 10 Introducing new variables of grid during design

of visual are designed and placed according to the values provided by behaviour variables of grid. During the design process, few structure elements of visual can be set to a new set of behaviour values by introducing secondary grid. This secondary grid is temporary and can be tacit or explicit.

First image in the Fig. 10 shows a magazine cover page where only two visual elements are set to a new grid with titled orientation. This grid is only in the form of alignment and might not be laid physically by the designer. It breaks the vertical flow and makes layout more interesting. A magazine spread on the right shows a detailed info-graphics. To increase the interest level, only few elements (insects and two parallelepipeds) are given a perspective and all other elements are kept in two dimensional grid. The perspective effect is achieved by separate grid which gives a sense of third dimension to these elements. This made these visual elements stand out from rest of the page.

Another way to introduce a new variable is to introduce a new external structural element. This element might be a part of the final visual and hence visible or might be removed after the layout is done. The structure of the visual is guided by this guest element. The visual shown in the left of Fig. 11 is a layout from the sports section of a daily newspaper. Here, two circular arcs are added to the layout and the arrangement of visual elements is influenced by it. The designer has used them to create an illusion of a running track. The behaviour exhibited by these arcs has created a dynamic circular channel which would not have been possible with a static column grid. The circular arcs are visible in this design whereas the middle layout also shows the structure guided by the behaviour of a semi-circular arc which is removed after the design. The circular path created by this invisible extra element was followed by the position and alignment of text as well as the arrangement of the images.

Sometimes, a visual element from the content or the part of it acts as a new variable. The image on the right in Fig. 11 shows the outline of trophy cup acting as a text box. The outline traced from this object offers the value for area variable which was used to put the text. This technique is very much content specific, contextual and many times spontaneously used by designer. It is a common tool practiced by designer to generate a creative impact.

In Fig. 12, there are three images out of which two are posters and last one is a page from a newspaper. First two designs are not based on any regular grid. The first poster on the left has a visual as a key element. The placement of the digital screen is influential in the orientation of the text. The value for the orientation variable is given by the orientation of the screen. In the second image, the text is a typographic compilation of answers to various questions. They are arranged in the three dimensional form of a question mark. In both these examples, the structure or part of structure of proposed visual governs the behaviour of grid. It is worth to discuss this scenario further with the FBS framework (refer Fig. 1).

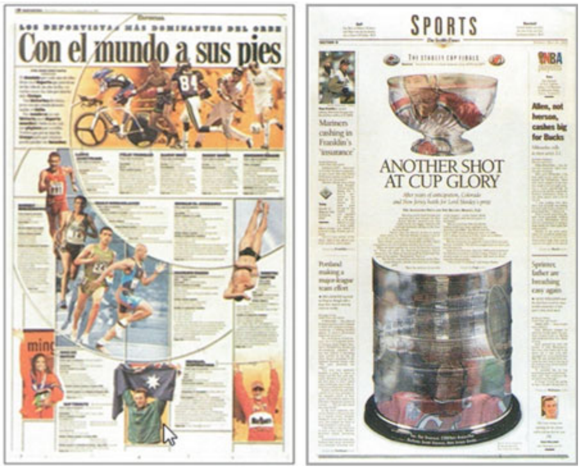


Fig. 11 Creative use of grids by inserting a new variable during design

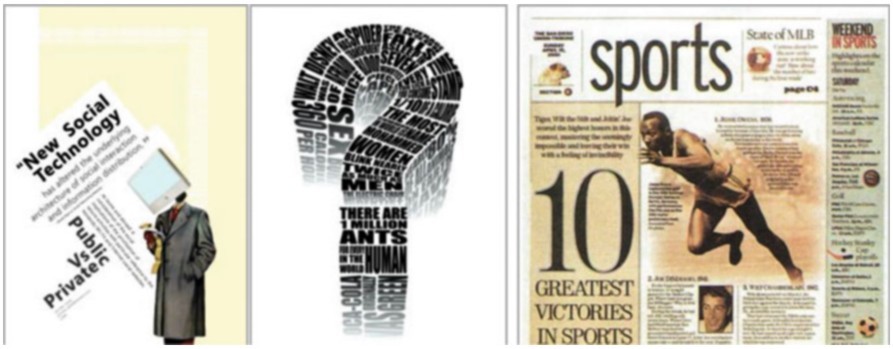


Fig. 12 Grid behaviour reacting to the visual structure element

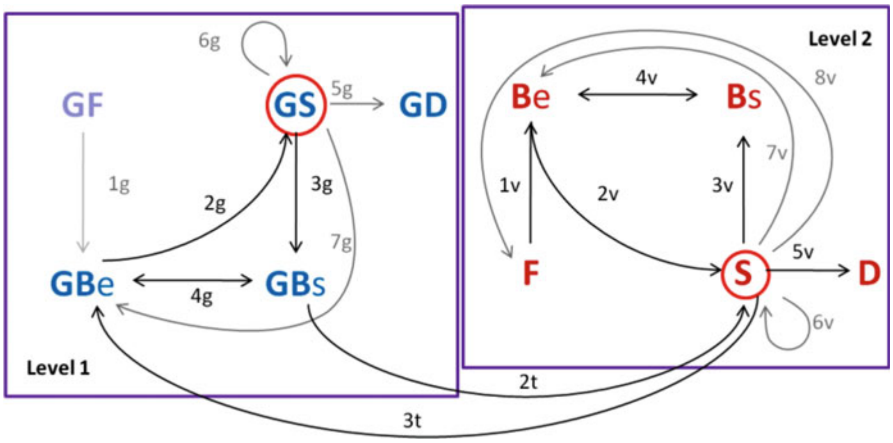


Fig. 13 Special case FBS path- grid behaviour is derived from visual structure

In case of the two images discussed above the total content was very less and both were single instances of design and not a part of a design series. The need for order and consistency through grid is not a major concern of the design here. The digital screen in the first image and the idea of putting words in the form of question mark are evolved during visual FBS path at level 2 (refer Fig. 13). These entities then defined some of the structure variables of visual. It is these values which transforms into the expected behaviour variables of the grid. Digital screen has defined the orientation variable and question mark has defined all the structure variables of the grid. Hence in the FBS path, GB_e got defined by S instead of GF.

In the third example in Fig. 12 is a newspaper page from sports section. As mentioned earlier, a newspaper is the application where grids are required the most. The text flowing horizontally set in vertical columns gives a consistent orderly look. To break the static tabular structure, the photograph of athlete Jesse Owens is intelligently placed in such a way that the rectangular borders of the photo are merged with the page and Jesse is coming out by breaking the grid. The angular posture along with this perturbation creates a dynamic effect. Now this dynamic figure as a structural element of visual starts affecting the rest of the structure. It got transformed (3t) to grid structure through grid behaviour. It has thus started acting as a part of grid and the text is arranged enveloping around the edge of Jesse.

Breaking the Grid

Breaking the grid means allowing any structure variable of the visual to attain a value other than the values provided by behaviour variables of grid. When a structural element of a visual acquires a random value outside the set of values provided by the grid, then by definition it creates an innovative option for the layout design. To avoid the boxy look or monotony of grid it is advisable to break the grid occasionally. The most important aspect is to understand where to break it and to what extent. It should be skilfully done so that the expected order should be maintained and enough interest should be brought into the visual.

The breaking of grid can be categorized as per the degree of perturbation. The different ways in which a grid can be broken are listed below:

1. By perturbation of single attribute of at least one structural element
2. By perturbation of multiple attribute of at least one structural element of visual simultaneously
3. By perturbation of single attribute of many structural elements of visual simultaneously
4. By perturbation of multiple attribute of many structural elements of visual simultaneously
5. By adding new structural element/s which is/are not in accordance with the grid behaviour variables
6. By using multiple grids disturbing each other's behaviour variables

The attributes of structural elements can be perturbed randomly and spontaneously by the designer, but after defining the variables a systematic method can be proposed using behaviour transformation rules. From the five direct behaviour variables (position, orientation, area, proportion and sequence), one or more variables can be used for transformation. For example, within a layout only the position and orientation attributes of a structural element can be perturbed to break the grid. It is essential to discuss each of the cases (listed above) for breaking the grid with ample examples; but it is not feasible to cover it in this paper within the space constraints. Here is a brief discussion on two visuals breaking the grid.

In Fig. 14, there are two examples of magazine pages which demonstrate the breaking of grid. The first example has an image at the top left corner which breaks the boundaries of area provided by the grid. The transformation rule is applied to area variable. It does not disturb the grid on the rest of the page but enhances the interest level. In second example, the image at the background breaks the column pattern of the text. The text follows the discipline of grid whereas the background

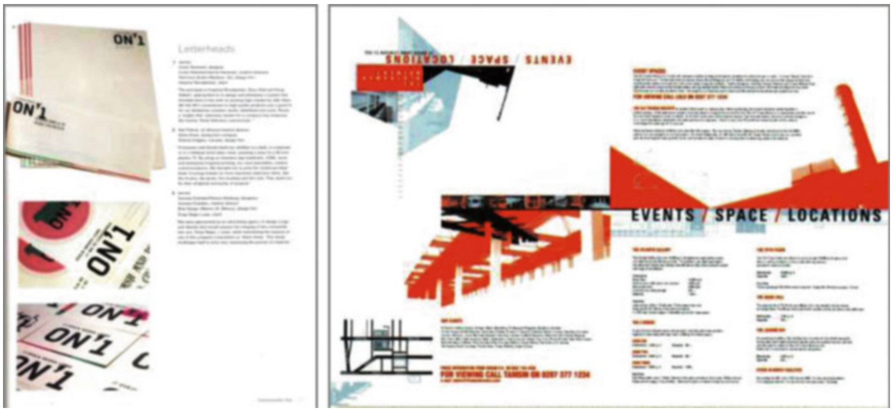


Fig. 14 Breaking the grid by randomising the values of behaviour variables of structure elements

images and other visual elements are used to create more interest by loosening the geometric boundaries. The image does not accord with the rest of the grid. It overflows the grid boundary and also breaks the verticality by many angular shapes. This new value of behaviour variable is decided by the designer during the design process as a response to the overall aesthetic quality of the visual in making. The new values can be based on another grid as discussed before or spontaneously derived from the context. The designer can always choose to work without grids. The values of structure variables of visual can be spontaneously decided by the designer during the design process. Sometimes, designers start working with the grid but occasionally may break away completely from it to work intuitively.

Conclusion

Creating layouts in graphic design is considered to be an intuitive process and is rarely approached from a scientific perspective. Grid is an essential tool used for a systematic way of creating graphic layouts. Here is the first step towards unfolding the process for creative use of grids.

This paper has attempted to externalize the knowledge about grids. Function-Behaviour-Structure ontology is sufficient to articulate the process of layout design. The FBS representation provides a good understanding about the design process involving grids. The knowledge about the variables of grid has proved to be useful to analyze the process of making and breaking grids. These variables can be further used for the flexibility index of the layouts. Once this process is externalized it can be useful for many computer generated design programs and applications.

The next step for this line of research is to check the effect of this ontological knowledge on the design decisions and on the design quality in the context of graphic design.

References

1. Bosshard HR (2000) The typographic grid. Verlag Niggli, Zurich
2. Brockman JM (1981) Grid systems in graphic design. Arthur Niggli, Niederteufen
3. Swann A (1989) How to understand and use GRIDS. Quarto Publishing, London
4. Bokil P, Ranade S (2014) In: Gero JS (ed) Function-behavior-structure representation of the grids in graphic design, design computing and cognition ’12. Springer, Berlin, pp 533–552
5. Gero JS, Kannengiesser U (2012) Representational affordances in design, with examples from analogy making and optimization. Res Eng Des 23:235–249
6. Sembugamoorthy V, Chandrasekaran B (1986) Functional representation of devices and compilation of diagnostic problem solving systems. In: Kolodner J, Riesbeck C (eds) Experience, memory and reasoning. Erlbaum, Hillsdale, pp 47–73
7. Gero JS (1990) Design prototypes: a knowledge representation scheme for design. AI Mag 11 (4):26–36
8. Gero JS, Kannengiesser U (2007) A function-behaviour-structure ontology of processes. AIEDAM 21(3):295–308
9. Krauss RE (1985) The originality of the avant-garde and other modernist myths. MIT Press, Cambridge
10. Goel AK (1997) Design, analogy, and creativity. IEEE Expert Intell Syst Appl 12(3):62–70
11. Qian L, Gero JS (1996) Function-behavior-structure paths and their role in analogy-based design. Artif Intell Eng Des Anal Manuf 10(4):289–312
12. Bokil P (2009) Functions of grid, a key for flexibility in framework. Des Thoughts 2:42–48
13. Bokil P (2013) Knowledge representation of grids in graphic design and its application for analogy-based design. Doctoral Thesis, IIT Bombay, Mumbai
14. Brown DC, Chandrasekaran B (1985) Expert systems for a class of mechanical design activity. In: Gero JS (ed) Knowledge engineering in computer-aided design. North-Holland Press, Amsterdam, pp 259–282
15. Coyne RD, Rosenman MA, Radford AD, Gero JS (1987) Innovation and creativity in knowledge-based CAD. In: Gero JS (ed) Expert systems in computer-aided design. North-Holland Press, Amsterdam, pp 435–465
16. Vignelli M (1976) Grids: their meaning and use for federal designers. National Endowment for the Arts, Washington, DC, Vols. No. 036-000-00038-4
17. Hurlburt A (1978) The grid – a modular system for the design and production of newspapers, magazines, and books. Van Nostrand Reinhold Company, New York
18. Samara T (2005) Making and breaking the grid – a graphic design layout workshop. Rockport Publishers, Beverly
19. ddo (2011) The 892 unique ways to partition a 3 $\times$ 4 grid. 4 March 2011. http://www.dubberly.com/concept-maps/3x4grid.html. Accessed 12 June 2011

Ontology-Based Process Modelling for Design

Andreas Jordan, Matt Selway, Georg Grossmann, Wolfgang Mayer, and Markus Stumptner

Abstract The design process for large systems, e.g., industrial plants, involves large multi-disciplinary teams. Since each discipline has its own specialised concerns, the common thread is describing the functional requirements of an artefact. In the oil and gas industry, Engineering, Procurement & Construction (EPC) companies are responsible for designing industrial plants whose later use requires exchange of information which is often based on different formats and leads to huge costs due to interoperability overhead. One contender in this space is the ISO15926 standard used for industrial design data interchange. We examine several challenges the standard poses for the conceptual and detailed design phases, and provide an alternative framework grounded on a firm ontological foundation incorporating advanced modelling concepts such as multi-level models and roles that provide a basis for the effective development of meta-model mappings, a novel approach for organisations needing to map to ISO15926.

Introduction

The ISO15926 standard is a data management standard being developed to facilitate the integration of data in support of the life-cycle activities and processes of process plants. Its main current use is for design data documentation in EPC companies. The focus of this paper is Part 2 of the standard (ISO15926-2), a generic data model defined in STEP and OWL. Although Part 2 has been suggested as a universal upper ontology, it has been shown that it suffers from significant shortcomings such as terminological confusions that make it difficult to understand and apply [1]. Due to its complexity, it is not always evident how ISO15926 is intended to be used as a data model, hence the need for Templates for data exchange. For design, particularly from the conceptual design to the detailed design phase, the use of ISO15926 is extremely challenging. To the best of our knowledge, no tool exists that guides users in an intuitive way for creating representations of design descriptions.

A. Jordan (✉) • M. Selway • G. Grossmann • W. Mayer • M. Stumptner
University of South Australia, Adelaide, Australia
e-mail: andreas.jordan@mymail.unisa.edu.au

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_31

We argue that the ISO15926 treatment of standard design notions such as function (“role”) or mereology (“part-of”) is not sufficiently defined, leading to increased complexity and reduced functionality when developing design descriptions. The fusion of the ontological and data model aspects into the one model makes it challenging to introduce new modelling elements and results in a large complex model. We introduce a novel combination of techniques by making our ontology multilevel aware through the techniques developed in the Software Engineering domain by Atkinson et al. [2], providing a cleaner separation of concerns, and permitting effective use of constraints for the purposes of developing mappings between our ontology and ISO15926. We also incorporate a more principled notion of roles and show how these conceptual notions benefit design through a more rigorous ontological account, leading to a more concise representation. We perform a comparative evaluation between models produced by our approach to the models produced using ISO15926 by examples, beginning with a conceptual model representing a generic process design into a plant model, and demonstrate how we can use our modelling principles to support the semi-automated construction of executable mappings between our ontology and ISO15926.

Our approach uses the notion of ontological support for design processes as described in [3], including the notion of semantic interoperability “due to the introduction of meta-models that serve as a linking element between different design disciplines”. We are currently in the process of employing this capability in an engineering pilot (known as “Oil and Gas Interoperability Pilot” or simply “OGI Pilot” among the participants) that aims at the automated provision of operational systems from the information produced in design. This is referred to as “digital handover” from EPC to O&M (Operations & Maintenance) companies [4, 5].

The next section discusses the issues with use of the ISO15926 data model. This is followed by introducing the modelling concepts in our framework. We then develop a set of modelling examples focusing on both the conceptual and detailed design phases that show how we reconcile these differences, leading to the definition of executable mappings.

Background

Ontological analysis of designs is a well-studied area. Chen et al. [6] investigate a knowledge-based framework for conceptual design. However, they focus primarily on feature-centric capabilities, limited to the data model level. We build on functional representation, and go beyond this by addressing aspects of key ontological notions required for automated reasoning. Gero et al. [7] extend their function-behaviour-structure ontology [8] to represent processes. The FBS ontology is a domain ontology focusing on the interrelationships between function, behaviour and structure of artificial objects. While similar in aims to the work of

Gero [7, 8], i.e., to model and represent artefacts/artificial objects and processes in our domain ontology, our approaches differ in that we combine the ontological principles of philosophical ontology by extending the foundational ontology, DOLCE, together with the advanced modelling notions from the software engineering domain. Welch et al. [9] focus primarily on two steps of the conceptual design aspect and break this phase down into two steps, the transformation of functional requirements to a behavioural description and matching of physical artefacts to this behaviour. Our ontological framework captures behaviour through the use of roles as proposed in [10]. Kitamura et al. [11] developed an ontology of device function, however, the ontology does not explicitly cover the ontological representation of roles at the language level.[1] Their main focus is on managing expert knowledge for problem solution search. In contrast, we explicitly represent the notion of roles within our multilevel modelling framework improving its capacity to be used among different design tools, and our application focus is on the automated management, validation, and transformation of design knowledge. Compared to the expanded function modelling framework of Kitamura and Mizoguchi [12], we attempt to capture the lifecycle implications resulting from the setting intended for ISO15926.

Issues in Practical Use of ISO15926

The goal of the "OGI Pilot" is to demonstrate digital handover from the design files produced by the CAD suites of participant companies (Bentley, AVEVA, Intergraph – using ISO15926) to the IBM Integrated Information Core (IIC) intended as the central O&M system, which is based on the MIMOSA/CCOM standard. An ISO Technical Specification is being worked on by ISO TC184/WG6 to provide guidelines for sharing design information between the standards. The design data used in the pilot, though describing a fictitious plant, are actual engineering data produced by Worley-Parsons that will be used as a reference data set for testing data model compliance with the standards. The actual transformation is conducted by defining a joint Meta-Model and using it to construct a mapping between the standards.

The pilot's demonstration design is that of a bitumen refinery and is patterned after an ongoing real-world project. While the models are being extended to capture a whole refinery by 2016, the examples currently used in the pilot are based on a debutaniser tower, a specific major part of the plant that takes part in the fractionation process of the refinery. Figure 1 shows the initial conceptual sketch of the debutaniser developed at the outset of the project by a team of chemical engineers from four companies, and Fig. 2 shows a visual representation of the current set of design data as a Process and Instrumentation Diagram (P&ID).

[1] Instead their ontology construction tool Hozo [18] has this aspect built-in.

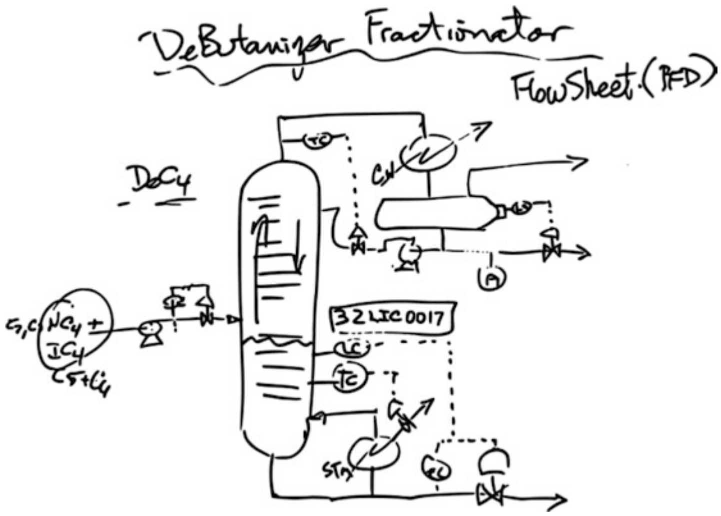


Fig. 1 Debutaniser schematic design

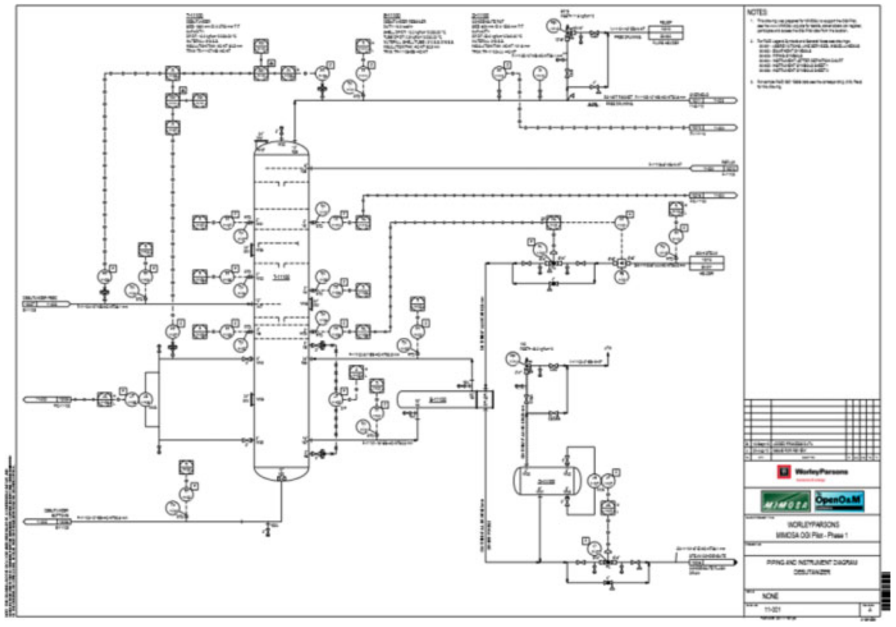


Fig. 2 Debutaniser detailed design

The modelling choices that were made in developing the data model of ISO15926 provides a set of unique challenges for any organisation needing to map to this standard. In this section we focus on the challenges impacting on the design modelling aspects. For example, ISO15926 provides no clear modelling guidelines, nor solid definitions, of the basic building blocks of the data model. This leads to multiple allowable representations of the same model, impeding interoperability. Moreover, it becomes more difficult to merge different aspects of a model and check whether or not they are compatible with one another. More importantly, the decision was made to define the data model in the Web Ontology Language (OWL) which possesses no meta-modelling capability. This means that certain complex engineering constraints cannot be directly expressed [13] and that common situations in a design setting cannot be directly represented in the language. The first problem lies in the representation of type hierarchies. Most modern modelling languages, whether aimed at engineering (STEP/EXPRESS, SysML) or generic ontology modelling (OWL) work on the assumption that a class (or concept)[2] is *instantiated* to produce individual instances, and the instantiation guarantees that the instance satisfies the properties defined in the class. The *specialisation* relationship relates a class A to a more specialised class (subclass) $A^{'}$, generally implying that instances of $A^{'}$ are a subset of the instances of A.

An important difference is that while specialisation can be applied repeatedly to form complex hierarchies or lattices, instantiation is generally restricted to one level of application: from classes to instances. Assume the following example hierarchy: The class Pump is a subclass of Rotating Mechanical Equipment. Centrifugal Pump is a subclass of Pump. We are designing a new pump type called Series 2100 Pump. Our design consists of a complex set of parts and relationships modelled in our design system. It is therefore natural to consider Series 2100 Pump to be a particular Instance of Centrifugal Pump. For certain properties that the class Centrifugal Pump or its superclasses possess (say, MaxRotationSpeed), Series 2100 Pump will provide a particular value. However, once Series 2100 Pump goes into production, it is just as natural to consider the individual pumps installed in a plant as instances of the Series 2100 Pump type. Existing modelling and ontology languages do not support this double instantiation. In practice (and this is the solution chosen in ISO15926), one of these levels is therefore expressed as a new relationship defined as part of the domain model. By doing this, the support provided in the language for creating and validating instance relationships is lost.

ISO15926-2 employs domain model relationships which seemingly introduce a limited means of representing multiple classification. For example, Fig. 3 shows the part/whole relationship between a pump and an impeller, in which we can see two types of *Classification*[3] and three different types of *Composition* relationships (one at each "level of classification") being used. Each of these is introduced by prefixing 'class_of'[4] one or more times to the relationship names. The elements in boxes are

[2] In this paper, we use these two terms interchangeably.

[3] In the text, we use *italic* font for all ISO15926 classes. In contrast, names of model elements (concepts and relations) from our multilevel role-oriented ontology are written in Calibri.

[4] 81 of the 201 concepts comprising the data model are prefixed with either 'class_of' or 'class_of_class_of'.

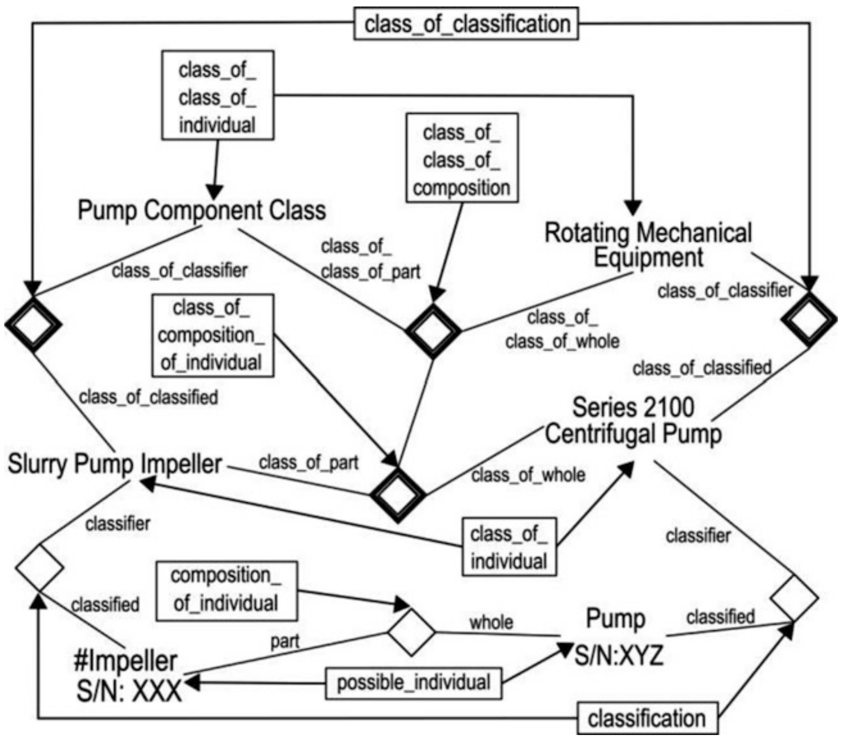


Fig. 3 EXPRESS instance diagram relating an impeller to a pump

concepts from the data model level, while those without are their instances at the model level.

Combining these multiple levels in the same model leads to an explosion of relationships that cannot be automatically validated. Important design-relevant constraints related to these classes cannot be validated or even formulated in a generic manner. The second key issue is the modelling concept referred to in ISO15926 as the “role” of a particular part in the design, which roughly corresponds to the function fulfilled by a particular component. (For example, “we use a pump of Series 2100 as a sump pump”). These “functional roles” are defined through the use of explicit *Classification* and *Specialisation* relationships. The use of *Classification* for an entity filling a role raises some ontological concerns in that it excludes the dependence on the context; with the fact that entity playing a role is only dependent on being Classified by the role. For example, a particular pump could be installed as a sump pump. If it is later removed and installed in a different function elsewhere, it should no longer be Classified by the sump pump role, but since this dependency is not explicitly modelled in ISO 15962 it could continue to be falsely retained by the original *Classification*.

World-Views: 3D (Endurantism) vs 4D (Perdurantism)

A conceptual model based on a 3D view of the world is fundamentally different to modelling in 4D. One of the most important distinctions is recognising what constitutes identity of an object [14]. In the 3D view, considered more in-line with a commonsense understanding of the world, the three spatial dimensions are considered separately from time and objects are recognised as having identity. In contrast, a 4D view treats time as a fourth dimension with the identity of an object being its trajectory through space-time. Neither view has gained clear preference in the ontology literature. The 4D model makes understanding, modelling, and applying ISO15926 more complex; particularly when mapping 3D-based standards to it. Therefore, our target model is based on the 3D world view to facilitate alignment between the conceptual view and implementation structure.

Advanced Modelling Concepts

In this section we introduce a number of advanced language concepts that can be used to overcome the issues previously described.

Multilevel Modelling

Since UML's adoption by the Object Management Group (OMG) in 1997, it has become the standard modelling language in Software Engineering and its derivate, SysML, in Systems Engineering. Although UML has shown significant value in many areas, a number of limitations have been identified over time. A key issue relates to the instantiation relationship which can only carry information concerning attributes and associations across a single level [2]: from classes to their instances. Therefore, the notion of equipment types (say, rotating equipment, of which a particular pump type would be an instance that would in turn be instantiated to create an individual pump), could not be captured while obeying the constraints of the UML framework.[5] In order to represent the model in Fig. 3 more effectively, we employ *multilevel modelling* principles, i.e., supporting more than one instantiation level.

In this paper we have employed the multilevel approach of Atkinson et al. [15] which introduces two key notions. The first is the notion of a "clabject" [2]) a modelling element that combines the properties of a class (in that it can be subclassed and instantiated) and an instance (in that it is an instance of a higher

[5] Although superficially, UML appears to provide multiple levels, they cannot be used for domain modelling. Cf. [3] for further detail.

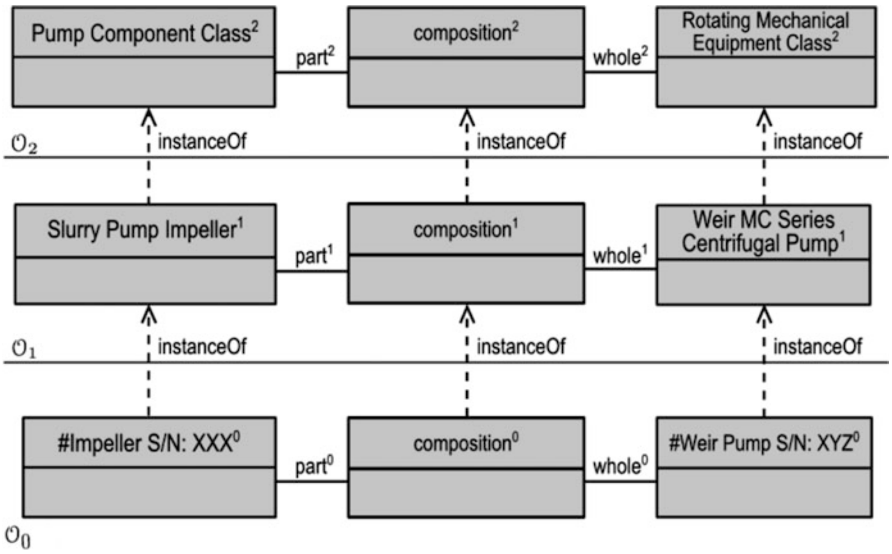


Fig. 4 Multilevel representation of pump and impeller

level of clabject). The second notion introduced is that of *potency* [2] which represents how far down this level of influence extends past the first level of instantiation. (In UML, when a class is instantiated, its attributes turn into slots and the class itself cannot influence further down the instantiation hierarchy.). Figure 4 illustrates the application of multilevel modelling to the pump and impeller in the previous figure.[6] On this basis, multiple levels of instantiation can be built as desired. Each time a clabject is instantiated, potency is reduced until its value reaches 0. When this occurs, the relevant clabject is then the equivalent of an instance at the "normal" two-level instance model level (e.g., representing an individual physical pump installed in the plant). By employing multilevel modelling techniques we can omit the syntactical distinction used in the ISO15926-2 *class_of* prefix naming scheme and replace it by actual instantiation. The model elements in the figures demonstrating our approach are all clabjects, and their level in the model is given by their potency values. Clabjects with potency 0 represent real-world objects, while clabjects with potency 1 represent those used for building the conceptual design models, and these levels can be selectively displayed in a tool. Multi-level modelling principles appear ideal for resolving the issues identified in Fig. 3 for several reasons: (1) the modelling of physical systems often utilises the "Type-Object" pattern [15], e.g., models of assets include both the object itself and its type; (2) the formalisation of modelling levels allows for the simplification of terminology, i.e., no need for the 'class of' prefixes; (3) many aspects that need to

[6] Potency is shown as a superscript next to the element name.

be modelled explicitly in ISO15926 become implicit or inherited according to the semantics of the multilevel modelling approach; and (4) the usage of the data model becomes clearer as interactions between elements on different layers become more evident.

Roles

The ontological concept of *role* can be described as representing "an entity that is played by another entity in a context" [16]. This rather abstract pattern covers a wide sweep of real-world scenarios, usually involving a social context or an agent attempting to achieve a goal. When modelling a social context, a typical role example is student: we say that another entity, e.g. person, can play the role of a student in the context of some learning institution. In the design context, the typical role relationship is based on the concept of function: a particular pump is used as the sump pump in an engineering plant, a particular type of switch as the power button, etc. A number of theories of roles have been proposed, [10, 14, 17–19]. While no universally accepted formalisation has emerged, there is general agreement upon a common set of fundamental characteristics [10]. Roles are *anti-rigid*, i.e. a role is non-essential to all its instances – an entity could go through its lifecycle without ever ending up in that role. Roles are *externally founded*, i.e. roles require external concepts to define them and they are *dynamic*, i.e. entities can stop and start playing one or more roles. The last condition is central in the design context – it means that roles, as a modelling construct, are perfectly suited to represent the notion of *using* a particular part to achieve a particular function.

Within ISO15926, the concept of a role is generally used as a shortcut for device function. For example, a particular gas turbine can play the role of 'auxiliary gas turbine' in a particular plant design. While this is a relatively shallow model compared to the detailed analysis conducted in work such as [12], roles have been used in this form before [16], and it is easy to see how a particular device lifecycle would satisfy the three properties above. Our specific contribution is to combine the role concept with the multilevel modelling concept for a more effective and simple function representation at the level provided by ISO15926 that also permits a more effective validation of design relationships.

We follow the formalisation of Relational Roles presented by Mizoguchi et al. [10] and use the following terms to describe the different aspects of roles: The *Role Concept* is analogous to *Role* in ISO15926: it describes the type of role that is to be performed. In the design context, this naturally expresses the intended function of a part (e.g. Fractionator, Pump, etc.). A *Role Holder* represents the fact that a *Role Player* has stepped into the role defined by the Role Concept. In modelling terms, it means we reify the role concept and the role player as a dedicated new entity which we call RoleHolder. The *Context* is the situation that the role concept is dependent on, while the *Player Concept* represents the class of

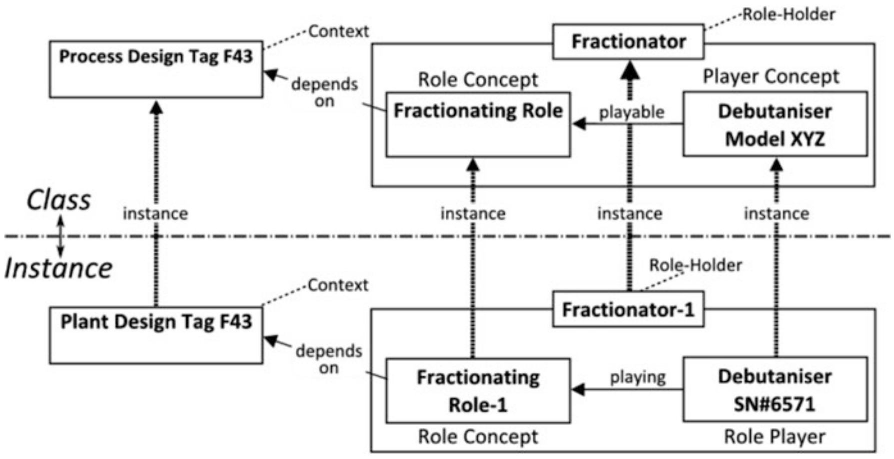


Fig. 5 Relational roles [14]

objects that can become *Role Players*.[7] Figure 5 shows a Debutaniser playing the Fractionating Role in the operation of a plant (in the real-world) and the class of debutanisers selected as the potential player of the Fractionating Role in the process design. In the following sections we demonstrate the benefits of employing a solid ontological notion of roles in both the conceptual and detailed design phases.

Design Support: Process and Plant Design

This section compares and contrasts the modelling of process and plant design between ISO15926 and our framework. A simplified block flow diagram of the debutanising process introduced in the Background section is shown in Fig. 6. We apply the notion of roles, multilevel modelling and a 3D world-view to address the issues outlined earlier in the paper.

Modelling Participation of Streams with Activities

A common modelling activity in conceptual design is the representation of the processes between streams and the activities they are involved in (e.g. pumping, distillation, etc.). In Design Engineering terms, this is a subjective process as the model represents the implicit intentions of the modeller [20].

[7] For simplicity we slightly deviate from the naming conventions in [17].

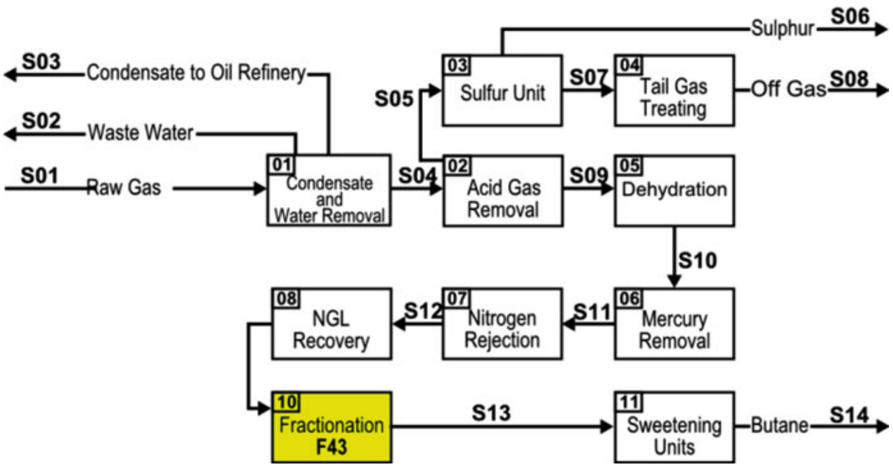


Fig. 6 Block flow diagram – simple process design

The term *stream* here is used according to its definition in ISO15926, where streams are physical objects whose identity is determined by continuity of flow path. This is a different concept from the notion of a system's functional flow in function modelling. A stream can be modelled as either an abstract functional object at the class level (e.g. Functional location in a process design) or as a physical object at the instance level.

Conceptual Design: Modelling Activities

During the design phase, ISO15926 deals with classes of 'things' (such as activities, streams, pumps etc.) and so introduces a *ClassOfActivity* model element. The relevance of the prefix *ClassOf* is to indicate that it is referring to activities at the design level as opposed to its concept *Activity*, which is used to model particular occurrences of activities at the instance level. An example of confusing terminology is provided by the ISO15926 definition for *ClassOfActivity* given as follows: A **class_of_activity** is a **class_of_arranged_individual** whose members are instances of activity [21]. This definition is problematic from a modelling perspective as it defines the concept based on two similar relations: membership and instantiation. The reference to membership in ISO15926 refers to those objects that are explicitly linked via the *Classification* relationship, while instantiation refers to the 'instantiation' of an element from the data model. This definition (which is typical of the 'class of' concepts in ISO15926) is actually defining the type based on the type of objects it is allowed to instantiate. As ISO15926 is based on a 4D world-view, a temporal relation *ClassOfTemporalWholePart* is required to indicate that only a temporal part of the *ClassOfActivity* is being modelled.

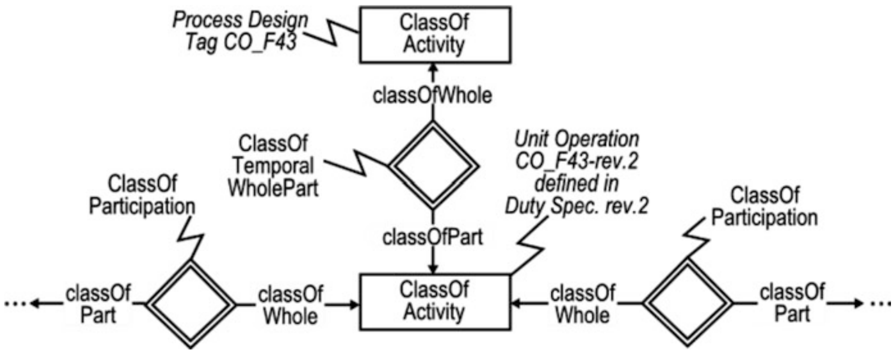


Fig. 7 ISO15926 – modelling an 'activity'

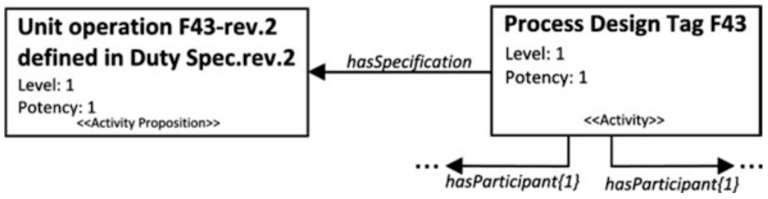


Fig. 8 Our framework – modelling an 'activity'

The temporal part represents a specific 'revision' of the activity in the process. Figure 7 illustrates the process design model in terms of ISO15926. It includes two instances of the reified relationship *ClassOfParticipation* that relates the (allowed) types of participants to the *ClassOfActivity*; these are used later to link the stream being processed by and the performer of the activity.[8] In contrast to the process design model in ISO15926, we instantiate Activity and Activity Proposition (see Fig. 8). These concepts have potency and level values of one, indicating that they are at the design (class) level of the model and that they can be instantiated once to the real-world (instance) level.

The Activity element represents the concept of the activity in the process while the Activity Proposition represents the specification of the activity that the instances of the activity should realise or conform to. This shows that the activity has been *designed* as opposed to other activities or occurrences that are not. Since our approach uses a 3D world-view, rather than requiring a temporal part to represent the revision of the specification, all of the relationships (e.g. hasSpecification, hasParticipant), are associated with a time index to indicate when the relationship holds. This simplifies the change process in design and reduces model complexity.

[8] Note that the double-diamonds indicate relationships at the design (class) level.

Constraining the Performer Role to a Fractionating Function

Prior to discussing the modelling of functions, it is important to clarify our treatment of functions in this paper. For the purposes of our paper we adhere to the definition as given in Kitamura et al. [11] which makes use of the widely accepted definition that the function of artefacts are distinctly related to the intentions of the designer. A shared commitment to a definition of function enables the addition of semantic constraints for functional knowledge that supports validation of models to ensure it complies to the agreed conceptualisation [11]. Moreover it supports re-use of designs due to the shared commitment of the term function whereas previously, definitions of function had been adhoc resulting in functional models whose semantics were not sufficiently aligned to support reuse.

Kitamura et al. in [22] distinguish between what they term 'actual function' which is aligned with the engineering domain in that "a function is directly related to a process performed by an artefact when the artefact is used." Addressing the philosophical definition of function, the authors refer to the term 'capacity function' that describes the possible functions based on the set of properties belonging to the artefact.

To model the "Fractionating Function" (as the performer of the activity) and the 'required stream' (as the stream undergoing fractionation), ISO15926 uses two classes, *ClassOfFunctionalObject* to represent the "Fractionating Function" and *ClassOfArrangedIndividual* to represent the "stream". The class *ClassOfArrangedIndividual* represents classes of objects that are arrangements of components [21], while a *ClassOfFunctionalObject* represents classes of objects whose identity is defined by their continuity of function. Classifying engineering artefacts (e.g. the pumper in a process) as functional objects allows the standard to maintain the designation in a model even when the allocated pump has been replaced as the 'function' of pumping has not changed.[9]

Modelling the association between the "Performer" role and the allowable types (the "Fractionating Function" in this example) that can play the role requires a *RoleAndDomain* to specialise the role and the allowed type through the two relationships *SpecialisationByRole* and *SpecialisationByDomain*, respectively. Specifically, the *RoleAndDomain* is a *ParticipatingRoleAndDomain*, which is simply a *RoleAndDomain* that can participate in an (*ClassOf*)*Activity* (see Fig. 9). The *ParticipatingRoleAndDomain* is then associated with the activity that it is a participant through the *ClassOfParticipation* relationship discussed above. In this situation the activity is the context of the role; however, the ISO15926 model does not make this explicit. Figure 10 represents the Performer role and the Fractionating Function using our approach. As we represent functions as roles in our approach, the model contains nested roles, where the role-holder for the type of object filling the Fractionating Function role (concept) is the potential player of the Performer

[9] Modelling the physical artefact fulfilling a function is out of the scope of this paper.

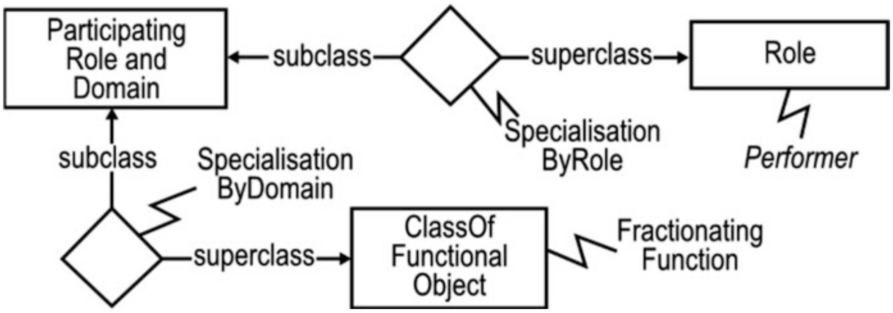


Fig. 9 Conceptual design using ISO15926 –function with role restriction

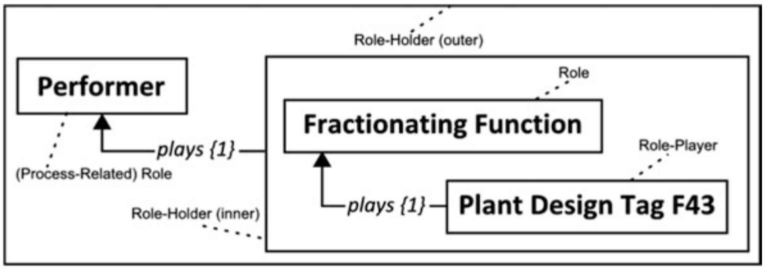


Fig. 10 Conceptual design using roles – function with role restriction

role.[10] This nesting can continue, for example, with the role holder for the functional place in the plant design as the potential player of the Fractionating Function. This results in a more compact representation of the roles and functions. In addition, our approach allows the actual restriction of the Performer role to Players that can fulfil the Fractionating Function, whereas the ISO15926 model only states that if something fills the "Performer" role of the activity, it has the "Fractionating Function", regardless of whether or not it can actually perform that function.

Modelling Fractionating Function, Stream and Activity

The complete example of the *ClassOfActivity*, both its participant roles, and the mapping between the model of ISO15926 and our approach is shown in Fig. 11. To help illustrate the mapping between the ISO15926 model to our domain ontology, the areas outlined in red show the key parts of the translation. Parts 'A' and 'C' in Fig. 11) uses the fact that ISO15926 restricts the possible ends of a *ClassOfParticipation* relationship such that the *classOfPart* relation must be a *ParticipatingRoleAndDomain* which ISO15926 uses to specify a *Role* and a Domain, i.e. the

[10] Context is not shown in this figure.

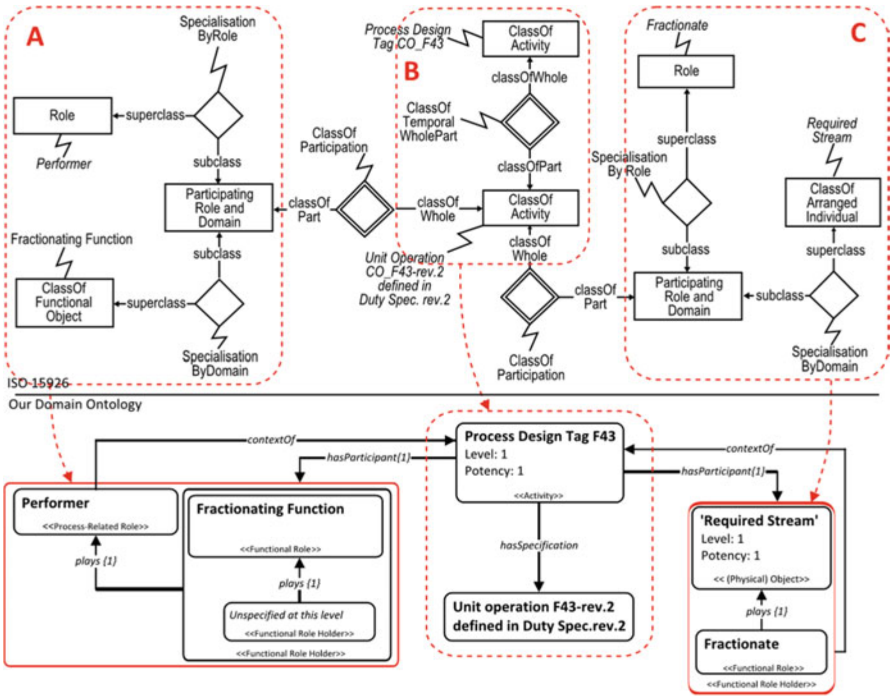


Fig. 11 Process model w. mappings between ISO15926 and domain ontology

ClassOfFunctionalObject. Referring to the area marked ‘A’ in Fig. 11 we can generate an instance of a Role Holder where the Role Concept is equal to “Performer”, the Role Player is the ‘Fractionating Function’, and the context is provided by the *ClassOfActivity* indicated in the area marked ‘B’.

Detailed Plant Design

In this section we demonstrate the detailed design aspect which links the conceptual process design of Fig. 11 to a plant design. A summary of the plant design model is shown in Fig. 12. Part ‘B’ of Fig. 12 shows how ISO15926 models the fact that a particular revision of Plant Design Tag CO F43 (a functional place in the design of the plant intended for the debutaniser) meets the requirements of the process design. This is modelled in ISO15926 by specialising the *ClassOfParticipation* (shown in grey) to another *ParticipatingRoleAndDomain* constrained by a *Specialisation-ByDomain* relationship to the particular revision (Plant Design Tag CO F43-rev.2) defined through the temporal relation *ClassOfTemporalWholePart*. This constraint is used to state that the particular revision of the plant design meets the requirements of the (particular revision of the) process design (for that function

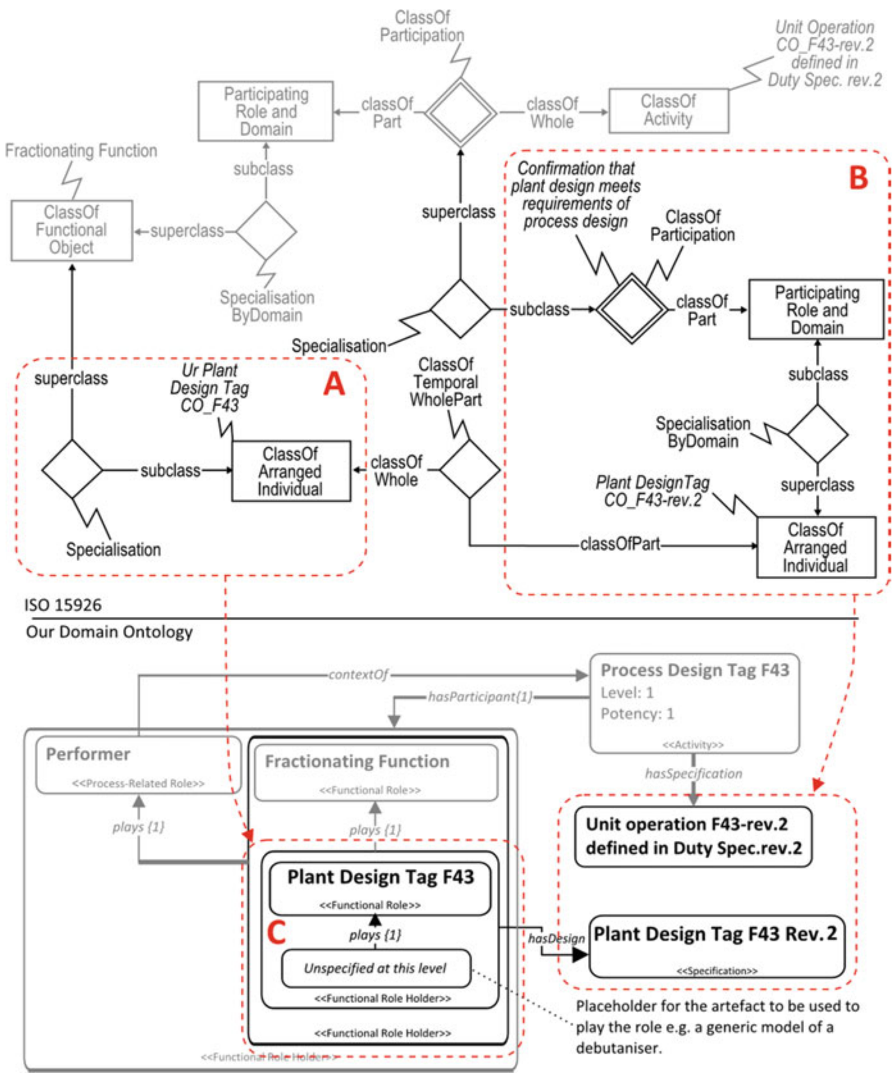

ClassOf Participation
Unit Operation CO_F43-rev.2 defined in Duty Spec. rev.2
Participating Role and Domain
classOf Part
classOf Whole
ClassOf Activity
Fractionating Function
subclass
ClassOf Functional Object
superclass
Specialisation ByDomain
superclass
Confirmation that plant design meets requirements of process design
ClassOf Participation
B
Specialisation
subclass
classOf Part
Participating Role and Domain
superclass
Ur Plant Design Tag CO_F43
A
ClassOf Temporal WholePart
subclass
Specialisation ByDomain
subclass
ClassOf Arranged Individual
classOf Whole
superclass
Plant DesignTag CO_F43-rev.2
Specialisation
classOfPart
ClassOf Arranged Individual
ISO 15926
Our Domain Ontology
contextOf
Process Design Tag F43
Level: 1
Potency: 1
<<Activity>>
hasParticipant{1}
Performer
<<Process-Related Role>>
Fractionating Function
<<Functional Role>>
hasSpecification
plays {1}
plays {1}
Unit operation F43-rev.2 defined in Duty Spec.rev.2
Plant Design Tag F43
<<Functional Role>>
C
plays {1}
hasDesign
Plant Design Tag F43 Rev. 2
<<Specification>>
Unspecified at this level
<<Functional Role Holder>>
<<Functional Role Holder>>
<<Functional Role Holder>>
Placeholder for the artefact to be used to play the role e.g. a generic model of a debutaniser.

Fig. 12 Plant model with mappings between ISO15926 and domain ontology

place). However, there is nothing in the model to actually enforce it. The user may have defined some of the plant design incorrectly so that it does not actually meet the requirements at all. Recall that in Fig. 11, the potential player of the "Fractionating Function" was left unspecified.

In Fig. 12 we are now able to expand the role-holder specifying the role concept to be the Plant design Tag F43. Since the plant design is the specification of a function place in the design (indicated by the specialisation of the "Fractionating Function" in the ISO15926 model), the Plant Design Tag F43 is modelled as a (functional) role in our approach in the same way as the Fractionating Function itself. As we are not including the modelling of the artefact that will actually perform the function in this part of the plant design, the potential player of this role is left unspecified.

Furthermore, rather than the specialisation of the "Fractionating Function" used in ISO15926, we use the role holder (**C** in Fig. 12) produced by the composition of the (unspecified) artefact playing the role of the Plant Design Tag F43 as the potential player of the Fractionating Function role. In terms of mapping between ISO15926 and our domain ontology, the *Specialisation* relationship used to specialise the *ClassOfArrangedIndi- vidual* "Ur Plant Design Tag CO F43" (**A** in Fig. 12) is mapped to the expansion of the unspecified player in Fig. 11 playing the Fractionating Function.

As this was a role-holder itself, we can now assign to the role concept Plant Design Tag F43 an unspecified player of the role. To map the participation of the Plant Design Tag F43 rev. 2 in the activity we create a relationship between the role-holder (**C** in Fig. 12) and the instance of a Specification that represents the Plant Design Tag F43 rev. 2.

Validating Models

One of the challenges with ISO15926 is that there does not appear to be a way to model certain constraints placed on classes at the design level in ISO15926 that instances of these classes can validate against. For example, assume we wish to add a constraint to the design class representing the "Fractionating Function" in Fig. 13 to the effect that any object performing the "Fractionating Function" must not draw more than a certain current, then there is no way to validate this against the actual debutaniser performing the "Fractionating Function" in ISO15926.

However, using our framework we are able to support such validation using the notion of roles and multilevel modelling. By using potency for classes, relationships and attributes (of classes), constraints placed at the design level can be checked down through to the actual instantiation of a physical pump performing the Fractionating Function (Fig. 14).

Conclusion and Future Work

In this paper we have investigated the use of advanced modelling techniques from software engineering and formal ontology research to provide a powerful modelling approach that captures fundamental design relationships expressed in the highly generic data model of ISO15926-2. We have outlined a number of issues that make the data model challenging to use, focusing on the conceptual and detailed design aspects. We illustrated how our framework, through the application of ontology engineering principles, explicit dynamic role relationships and multilevel

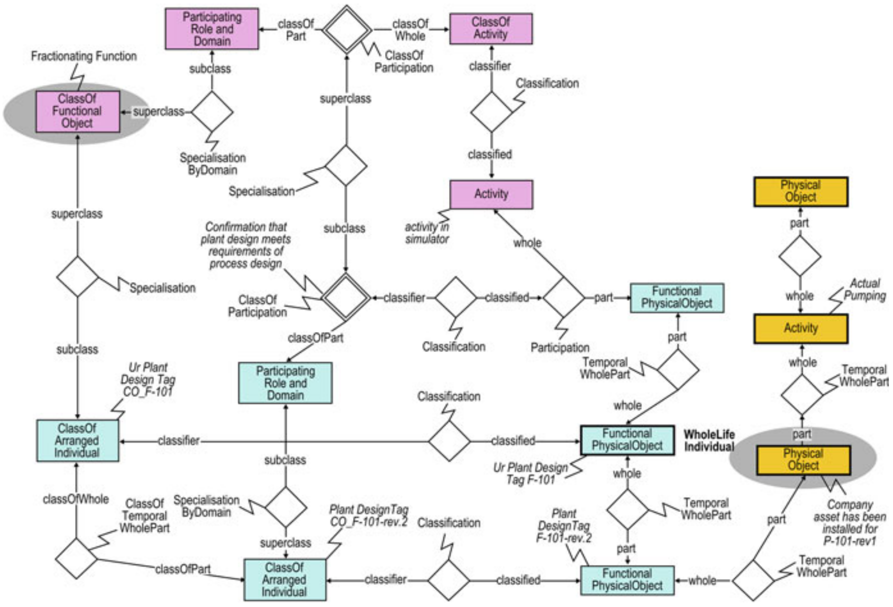


Fig. 13 ISO15926 unable to check if actual instance of pump is consistent constraints placed on its design class

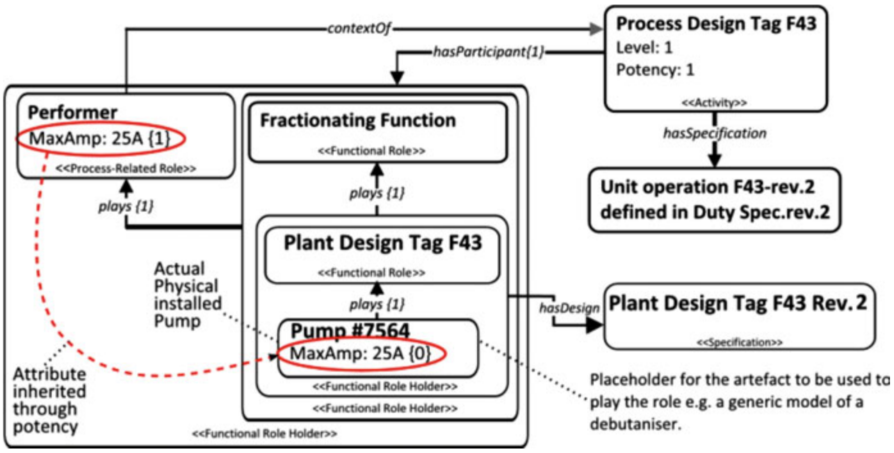

contextOf
hasParticipant{1}
Process Design Tag F43
Level: 1
Potency: 1
<<Activity>>
Performer
MaxAmp: 25A {1}
<<Process-Related Role>>
Fractionating Function
<<Functional Role>>
hasSpecification
Unit operation F43-rev.2 defined in Duty Spec.rev.2
plays {1}
plays {1}
Plant Design Tag F43
<<Functional Role>>
Actual Physical installed Pump
plays {1}
hasDesign
Plant Design Tag F43 Rev. 2
<<Specification>>
Pump #7564
MaxAmp: 25A {0}
Placeholder for the artefact to be used to play the role e.g. a generic model of a debutaniser.
Attribute inherited through potency
<<Functional Role Holder>>
<<Functional Role Holder>>
<<Functional Role Holder>>

Fig. 14 Inheritance through potency of attributes

modelling techniques, is able to provide a more succinct and intelligible model that permits reasoning and semantic validation of constraints. Work on the formalisation of the mappings between our domain ontology and that of ISO15926 along the lines of [22] is currently ongoing within the scope of the OGI Pilot [4]. Furthermore, while we have provided some comparisons between the

3D and 4D modelling approaches, we are working on a formal basis that permits capturing the scope of the latter in a 3D framework, thus enabling a principled approach to processing ISO15926 data whose modellers have so far had to sidestep the issue at implementation level.

References

1. Smith B (2006) Against idiosyncrasy in ontology development. In: Proceedings of the FOIS 2006, IOS Press, Amsterdam, pp 15–26
2. Atkinson C, Kühne T (2001) The essence of multilevel metamodeling. In: Proceedings of the UML2001, Springer, pp 19–33
3. Maier F et al. (2008) Ontology-based process modelling for design optimisation support. In: DCC '08, Springer, pp 513–532
4. Berger S et al (2010) Metamodel-based information integration at industrial scale. In: Model driven engineering languages and systems, Springer, Vol. 6395, pp 153–167
5. Mintchell G (2012) Open O&M demonstrates' information interoperability' for oil and gas applications. Autom World 2012:24–25
6. Chen Y et al (2011) A general knowledge-based framework for conceptual design of multi-disciplinary systems. In: DCC'10, Springer, pp 425–443
7. Gero J, Kannengiesser U (2006) A function-behaviour-structure ontology of processes. In: DCC'06, Springer, pp 407–422
8. Gero JS, Kannengiesser U (2002) The situated function — behaviour — structure framework. In: Artificial intelligence in design '02, Springer, pp 89–104
9. Welch R, Dixon J (1994) Guiding conceptual design through behavioral reasoning. Res Eng Des 6(3):169–188
10. Mizoguchi R et al (2012) Ontological analyses of roles. In: Proceedings of the FedCSIS '12, pp 489–496
11. Kitamura Y et al (2006) An ontological model of device function: industrial deployment and lessons learned. Appl Ontol 1(3–4):237–262
12. Kitamura Y, Mizoguchi R (2013) Ontological characterization of functions: perspectives for capturing functions and modeling guidelines. AI EDAM 27(Special Issue 03):259–269
13. Felfernig A et al (2003) Configuration knowledge representations for Semantic Web applications. AI EDAM 17(1):31–50
14. Masolo C et al (2005) Relational roles and qua-individuals. In: AAAI Fall Symposium on Roles, AAAI, pp 103–112
15. Atkinson C et al (2009) A flexible infrastructure for multilevel language engineering. IEEE TSE 35(6):742–755
16. Mizoguchi R et al (2007) The model of roles within an ontology development tool: hozo. Appl Ontol 2(2):159–179
17. Masolo C et al (2004) Social roles and their descriptions. In: KR2004, pp 267–277
18. Sciore E (1989) Object specialization. TOIS 7(2):103–122
19. Vieu L et al (2008) Artefacts and roles: modelling strategies in a multiplicative ontology. In: Proceedings of the FOIS '08
20. Erden MS (2008) A review of function modeling: approaches and applications. AI EDAM 22 (2):147–169
21. ISO 15926 (2003) Industrial automation systems and integration – integration of lifecycle data for process plants including oil and gas production facilities. Part 2: data model
22. Kitamura Y, Mizoguchi R (2010) Characterizing functions based on ontological models from an engineering point of view. In: Proceedings of the FOIS '10

Studying the Sunk Cost Effect in Engineering Decision Making with Serious Gaming

Sean D. Vermillion, Richard J. Malak, Rachel Smallman, and Sherecce Fields

Abstract One potential cause of project cost overrun is due to sunk costs. The sunk cost effect is a behavioral phenomenon where decision makers tend to continue a course of action, even an inferior one, if there has been a prior investment of resources in that course of action. In this paper, we study the sunk cost effect and the influence of decision framing on outcomes in an engineering decision-making context. We adopt an approach to human-studies by utilizing computer-enabled serious gaming as a platform to immerse subjects into a design scenario. Through an exploratory study encompassing two experiments, we find that decision framing as well as game mechanics influence subjects' susceptibility to the sunk cost effect. Decisions framed as *either-or* type questions and allowing subjects to test stated claims seem to mitigate the sunk cost effect.

Introduction

The design and realization of large, complex systems are often wrought with setbacks that strain resources to their limits. As the engineering process is largely a human-driven process, one can surmise that setbacks are largely human-driven as well. To understand how to remedy these setbacks, researchers must first understand how engineers and other stakeholders make decisions. The von Neumann-Morgenstern (vNM) axioms of rationality provide a mathematical framework for understanding and evaluating decisions [1]. While we generally favor increased rigor in decision making research within the study of design and systems engineering, it is conceivable that researchers' preference and belief assumptions embedded within decision models could be undermined by empirical data. In fact, the literatures of economics and psychology provide several examples of human decision makers acting contrary to theoretical predictions [2]; Austin-Breneman et al. [3] provide a notable engineering design example. In this paper, we adopt an approach to research that interprets empirical data from a theoretical perspective; this

S.D. Vermillion (✉) • R.J. Malak • R. Smallman • S. Fields
Texas A&M University, College Station, TX, USA
e-mail: sdvermillion@tamu.edu

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_32

approach is discussed in greater depth in Vermillion et al. [4]. In particular, we use this approach and serious gaming as the mechanism for experimentation to study the sunk cost effect in engineering decision making.

Sunk Cost Effect

The *sunk cost effect* is a tendency for decision makers to continue an endeavor once an investment of resources has been made in that endeavor, even when that endeavor is failing or a better alternative is available. As the design and systems engineering processes are recursive and decisions are made at every stage of these processes, it is intuitive to think the sunk cost effect can have major repercussions in the success of a product or system. Arkes and Blumer [5] point out that the US Congress decided to continue Tennessee-Tombigbee Waterway Project in 1981 despite severe setbacks on the basis that several billion dollars have already been invested in the project, i.e. they felt discontinuing the project would have been a waste of this investment. The development of the Concorde supersonic airliner is another example of decision making on the basis of sunk costs; the British and French governments chose to continue development based on capital already invested despite bleak financial prospects [6]. On a lower level, Viswanathan and Linsey [7] use the sunk cost effect to describe students' design fixation when using physical models.

Due to the implications that sunk costs could have on decision making, it is a well-studied topic in psychology. In a definitive paper, Arkes and Blumer [5] present results from a series of experiments in sunk cost conducted with surveys. They show that sunk costs are a basis upon which decision makers make choices, and this behavior likely stems from a desire to not appear wasteful. Furthermore, decision makers who have incurred a sunk cost tend to inflate a project's likelihood of success. Specifically in the domain of R&D project management, Garland [8] and Keil et al. [9] show through surveys that willingness to continue with a project that is over budget and unlikely to succeed increases with the amount of resources invested. However, Northcraft and Neale [10] and Keil et al. [9] show this willingness is mitigated by presenting an alternative. Generalizing this result, it seems rationalizing and explicitly defining an alternative course of action leads to a mitigation of the sunk cost effect [11]. We extend previous research in the sunk cost effect by using a more immersive mechanism for experimentation – serious gaming – as well as studying the influence of sunk costs and problem framing in a more concrete engineering design scenario.

Serious Gaming

Serious gaming is the practice of using games for purposes beyond entertainment [12, 13]. Games provide fully controllable environments such that players can interact amongst themselves and with in-game artifacts to achieve objectives. To encourage players to achieve these objectives, serious gaming exploits gaming's inherent capacity to motivate participation through entertainment and other rewards. Due to gaming's power to motivate, serious gaming has been extensively used for training and education. Within engineering education, serious gaming has been used to augment traditional instruction [14–18]. Beyond education and training, gaming has been used to facilitate participating in design activities [19] and enhance user experience in engineering CAD tools [20].

In this research, we use serious gaming within the research framework detailed in Vermillion et al. [4] as a means to represent the complexity in the design and systems engineering processes. Since using serious gaming in this context is relatively novel, the experiments discussed in this paper are largely exploratory. The experiments described in the next section use two different game designs meant to reflect the same systems design scenario. Results are then analyzed from a theoretical perspective and conclusions are made related to the differences in framing and game design.

Experiment Design

Subjects play a game called *Manned Mission*: *Mars* in which they assume the role of an engineer designing various systems needed for a manned Mars mission. Subjects are hereinafter referred to as players. The game consists of a series of levels, and each level tasks the player with designing a different system for the mission including a probe to search for a landing location, a rover, a lander, and a crew habitation module for astronaut travel to Mars. Players arrange differently colored blocks, representing different components in the system they are designing. See Fig. 1 for an example of a level's interface. The arrangement of these blocks affects the system's robustness, R; the probability of mission success, Pr(Success), is monotonically increasing with robustness, R. After players design their systems, the Mars mission is simulated so players can see if the mission was successful, i.e. the systems and astronauts successfully reach Mars.

The probe, rover, and lander levels are learning levels intended to introduce game mechanics to the players. After these learning levels are completed, players move on to the crew habitation module level; game mechanics in this level depend on the treatment the player is randomly assigned. Each experiment conducted contains at least a control treatment, i.e. a treatment without a built-in sunk cost component, and a sunk cost treatment. In both treatments, players are given information that another space organization has a design for a crew habitation module with very high robustness, R_2, with the insinuation that the player cannot create a design with a robustness, R_1, greater than this alternative's, i.e. $R_2 > R_1$ and thus $\Pr(\text{Success}|R_2) > \Pr(\text{Success}|R_1)$ will always be true despite the actions of the player. Given this information, players must make the decision whether to continue with their own in-house crew habitation module design or adopt the other space

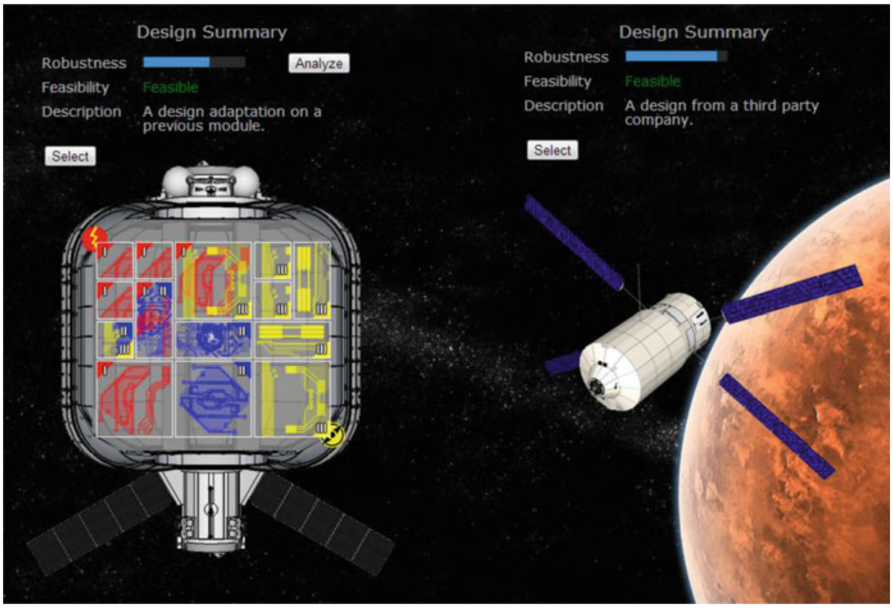
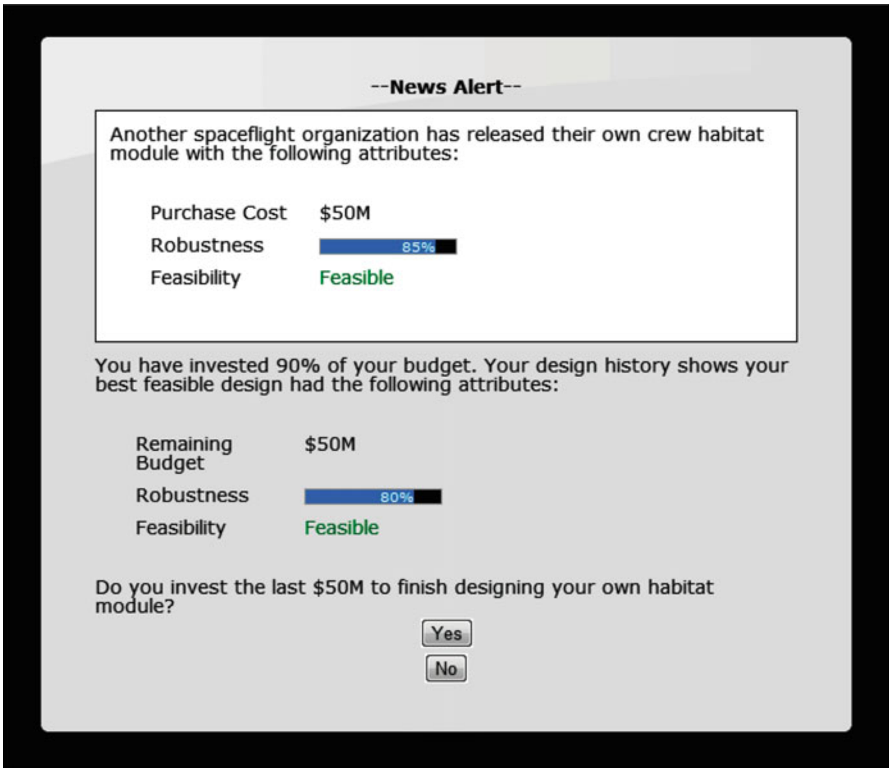


Fig. 1 Game interface example from the control treatment in Experiment 1

organization's design. In the control treatment, this decision is presented at the beginning of the level. In the sunk cost treatment, players are given a budget of $400 million, and each time they analyze a design candidate, i.e. click on an *analyze* button to evaluate an arrangement's robustness, $50 million is deducted from the budget; the decision is presented after players have invested 90 % of their allotted budget. Figure 2 shows an example of how the decision is given to players.

The experiments discussed in the following subsections are conducted under different conditions including differences in the way the decision is framed and game mechanics. Each treatment in the first experiment explicitly asks the player to choose between the two crew habitation module alternatives, i.e. *do you want to continue developing your design or purchase the other organization's design*? In the control treatment of the first experiment, players are allowed to experiment with an in-house design before they choose, as shown in Fig. 1. The interface for the sunk cost treatment in the first experiment is similar to Fig. 2 but with the aforementioned question.

Each treatment in the second experiment asks the player whether or not they would like to continue with an in-house design given the information available to them. A *Yes* response corresponds to continuing with an in-house response and a *No* response corresponds to abandoning the in-house design. The second experiment has four treatments: two control treatments and two sunk cost treatments. The control treatments differ only in when the question is asked; the prompt is shown in Fig. 3. One control treatment, labeled hereafter as Control – Beginning, asks the question prior to any game play, and the other, labeled hereafter as Control –


--News Alert--

Another spaceflight organization has released their own crew habitat module with the following attributes:

Purchase Cost	$50M
Robustness	85%
Feasibility	Feasible

You have invested 90% of your budget. Your design history shows your best feasible design had the following attributes:

Remaining Budget	$50M
Robustness	80%
Feasibility	Feasible

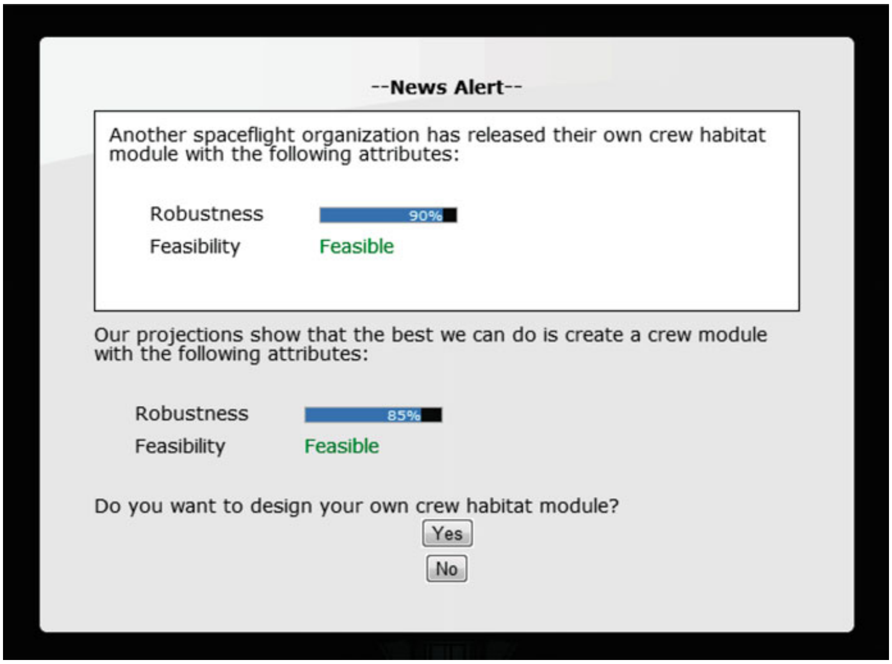
Do you invest the last $50M to finish designing your own habitat module?

Fig. 2 Decision prompt example from the sunk cost – implicit treatment

Midgame, asks the question at the beginning of the crew habitation module level; the idea here is to determine whether or not the game itself induces sunk cost behavior. In the control treatments of the second experiment, players are not allowed to experiment prior to making the decision. The two sunk cost treatments differ only by phrasing of the decision. One sunk cost treatment, labeled Sunk Cost – Implicit, only implies the other organization's design is purchasable, as shown in Fig. 2, and the other, labeled Sunk Cost – Explicit, explicitly tells the player that the alternative design will be purchased in the event that they choose not to continue with their in-house module. Game play in the sunk cost treatments of the second experiment are the same as that of the sunk cost treatment in the first experiment.

Subjects

Subjects are recruited from the Texas A&M undergraduate psychology pool. Students in this pool are typically in their first or second year of school, and the


--News Alert--

Another spaceflight organization has released their own crew habitat module with the following attributes:

Robustness 90%

Feasibility Feasible

Our projections show that the best we can do is create a crew module with the following attributes:

Robustness 85%

Feasibility Feasible

Do you want to design your own crew habitat module?

Fig. 3 Decision prompt example from the control treatments in Experiment 2

pool represents a wide variety of declared majors. For experiment 1, 78 students are recruited as subjects, and for experiment 2, 220 students are recruited as subjects. For the exploratory nature of this study in sunk cost with serious gaming, undergraduates offer a plentiful subject pool. Subjects play the game in a supervised, laboratory setting and are not allowed to interact with each other. The game takes approximately 30 min to play.

Gameplay Analysis Strategy

Gameplay results are analyzed using two statistical procedures. The Pearson's chi-squared test for independence informs on whether response proportions are independent of treatment. Let π_x be the population proportion of action x responses; the hypotheses for this test are the following:

H$_0$: π_x *is independent of treatment.*
H$_a$: π_x *is associated with treatment.*

The specific action x used in these hypotheses is defined for each experiment in the following subsections. The threshold value is set at $\alpha = 0.05$ such that conclusions are made with 95 % confidence. Secondly, confidence intervals for each response proportion informs on whether one response is more frequent than another. Let π_x^i be the population proportion of action x responses in treatment i; the hypotheses for this test are the following:

H$_0$: $\pi_x^i = 0.5$
H$_a$: $\pi_x^i \neq 0.5$

Just as with the test for independence, conclusions are made with 95 % confidence.

Results

Experiment 1

Results are summarized in Table 1 and Fig. 4. The two options available to the players, *continue my design* and *purchase new design*, are labeled *Design 1* and *Design 2*, respectively, in both Table 1 and Fig. 4. For the test for independent with $\pi_{Design1}, \chi^2 = 0.1657$ so $p = 0.6840 > 0.05$; H$_0$ for the test for independence cannot be rejected with 95 % confidence. Therefore, response is not necessarily dependent on treatment. For the population proportion test, $\pi_{Design\ 1}^{Control} \in [0.08, 0.36]$ and $\pi_{Design\ 1}^{Sunk\ Cost} \in [0.13, 0.40]$; therefore, H$_0$ for population proportions is rejected for

Table 1 Results from Experiment 1

		Choice		
		Design 1	Design 2	Total
Treatment	Control	8	28	36
	Sunk cost	11	31	42
	Total	19	59	78

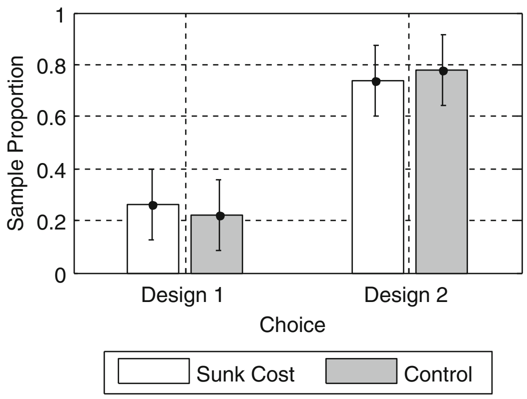


Fig. 4 Sample proportions from Experiment 1 with 95 % confidence intervals

Table 2 Results from Experiment 2

		Choice		
		Yes	No	Total
Treatment	Control – beginning	33	25	58
	Control – midgame	32	19	51
	Sunk cost – implicit	35	19	54
	Sunk cost – explicit	36	21	57
	Total	136	84	220

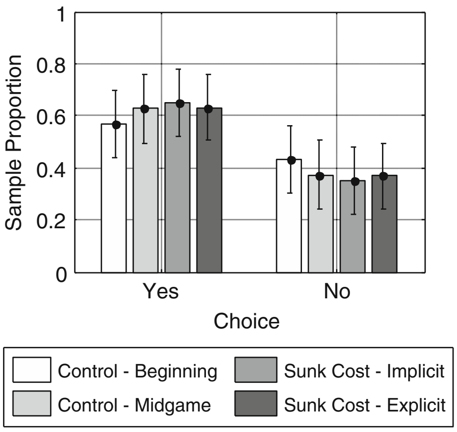


Fig. 5 Sample proportions from Experiment 2 with 95 % confidence intervals

both treatments. In fact, players choose *purchase new design* (*Design 2*) more frequently.

Experiment 2

Results are summarized in Table 2 and Fig. 5. The options *Yes* and *No* correspond to continuing an in-house crew module design and abandoning an in-house crew module design, respectively. For the test for independent with π_{Yes}, $\chi^2 = 0.863$ so $p = 0.8345 > 0.05$; H_0 for the test for independence cannot be rejected with 95 % confidence. Therefore, choice is not necessarily dependent on treatment. For the population proportion test in the Control – Beginning treatment, $\pi_{Yes}^{C-B} \in [0.4415, 0.6964]$ so H_0 for population proportions cannot be rejected. For the Control – Midgame treatment, $\pi_{Yes}^{C-M} \in [0.4948, 0.7601]$ so H_0 for population proportions cannot be rejected. For the Sunk Cost – Implicit treatment, $\pi_{Yes}^{SC-I} \in [0.5208, 0.7755]$ so H_0 for population proportions can be rejected in favor of H_a. Finally for the Sunk Cost – Explicit treatment, $\pi_{Yes}^{SC-E} \in [0.5064, 0.7568]$ so H_0 for population proportions can be rejected in favor of H_a. Therefore, there is no clear trend in choosing in the control treatments, or at least a trend cannot be claimed with 95 % confidence; in the sunk cost treatments, players tend to choose *Yes* more frequently than *No*, although just slightly.

Analysis

The results from the experiments are interesting in that treatment has seemingly no effect on decision choices; therefore, the presence of a sunk cost does not necessarily cause different behavior in this game. Furthermore, trends in the results of both experiments do not generally agree as the Sunk Cost – Explicit treatment in Experiment 2 should intuitively give the same results as the sunk cost treatment in Experiment 1. To gain insight into why the results turned out the way they did, the decisions in each treatment of each experiment are examined normatively, i.e. using formal mathematics and accepted rules of rationality. Prior to analyzing the results normatively, decisions with sunk cost are discussed both from an economic perspective and a behavioral perspective.

According to classical economic theory, sunk costs should have no impact on a decision as it is common to all alternatives [21]. The value, v, of alternative x is formulated as the following:

$$v(x) = b(x) - c(x) - s, \tag{1}$$

where $b(x)$ is the benefit gained by alternative x, $c(x)$ is the cost implementing and operating alternative x, and s is any cost expended up until the decision. The alternative that should be chosen, x^*, is the alternative that maximizes v_i such that

$$x^* = \arg\max_x v(x). \tag{2}$$

As sunk cost, s, is constant and common for all alternatives, x^* in Eq. 2 is the same as x^* in the following:

$$x^* = \arg\max_x b(x) - c(x). \tag{3}$$

Therefore, sunk costs are irrelevant in this decision formulation, and choosing anything other than x^* is deemed irrational. This is more of a prescriptive formulation for a decision with a sunk cost as opposed to a descriptive one.

To describe the behavior associated with the Sunk Cost Effect, Thaler [22] uses Prospect Theory, developed by Kahneman and Tversky [23]. This theory attempts to rationalize what classical economic theory deems irrational and provides a monotonic value function, v_p, similar to that shown in Fig. 6. Prospect Theory states that as a decision maker experiences loss, they become risk taking, and as a decision maker experiences gains, they become risk averse. As sunk cost is a loss, decision makers tend to be risk taking and are thus more willing to take a risk on the

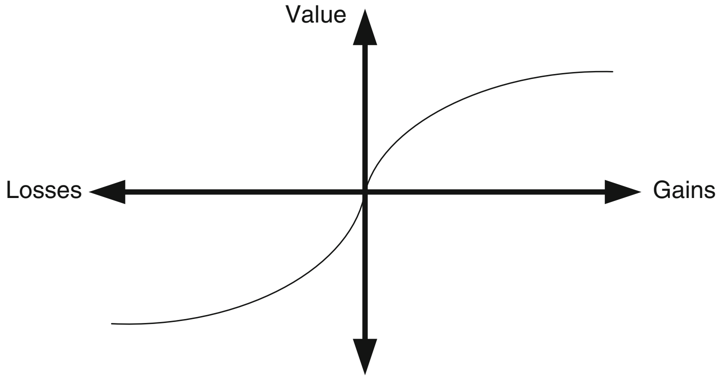


Fig. 6 Prospect theory value function [23]

success of a current project. This value function is routinely split into two functions at the inflexion point such that $v_p^+ \in (0, +\infty]$ represents the concave value function in the gain region and $v_p^- \in [-\infty, 0]$ represents the convex value function in the loss region. Let b be the benefit of an endeavor, c be the future costs associated with an endeavor, and s be the sunk cost. Furthermore, suppose the value of the benefit is equal to the value of its associated future costs such that $v_p^+(b) = -v_p^-(-c)$; therefore, a decision maker would be indifferent between undertaking the endeavor and not assuming no prior investments. In the presence of a sunk cost, a decision maker is compelled to choose to continue since

$$v(b) + v(-c - s) > v(-s), \tag{4}$$

where the left hand side is the total value for continuing the endeavor and the right hand side represents the value for abandoning the endeavor. This inequality holds with the assumption that $v_p^+(b) = -v_p^-(-c)$ since v_p^- is convex and monotonic such that $v_p^-(-c - s) > v_p^-(-c) + v_p^-(-s)$. This approach to describe the Sunk Cost Effect is really limited to decisions involving continuing or abandoning an endeavor.

In this paper, the Von Neumann-Morgenstern (vNM) utility theorem is used to analyze decisions mathematically. This theorem states that the preferences of a decision maker satisfying the vNM axioms of rationality can be captured by a utility function, u, that maps each decision outcome to a real number; uncertain outcomes are often described as a lottery, and a lottery L_1 is said to be more preferable to another lottery, L_2, if and only if $\text{Eu}[L_1] > \text{Eu}[L_2]$, where $\text{Eu}[L_i]$ is the expected value of utility, u, in lottery L_i [1]. Therefore, a vNM-rational decision maker should choose an action that leads to lottery L_1 since L_1 is more preferable than L_2. Analysis in the following subsections are primarily concerned with the probabilities contained within the outcome lotteries to gain insight into what player may have believed in order to choose the way they did.

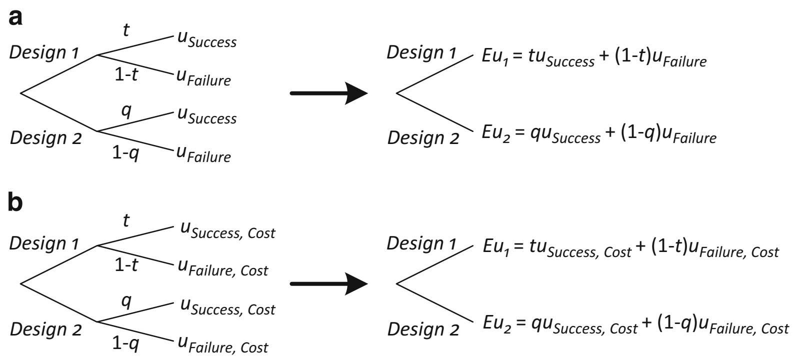

a
Design 1
Design 2
t
1-t
q
1-q
u_Success
u_Failure
Eu_1 = tu_Success + (1-t)u_Failure
Eu_2 = qu_Success + (1-q)u_Failure
b
u_Success, Cost
u_Failure, Cost
Eu_1 = tu_Success, Cost + (1-t)u_Failure, Cost
Eu_2 = qu_Success, Cost + (1-q)u_Failure, Cost

Fig. 7 Experiment 1 decision models: (**a**) control treatment and (**b**) sunk cost treatment

Experiment 1

The decisions for each treatment are modeled in Fig. 7. The following assumptions for the utilities of outcomes are made for normative analysis:

$$\begin{aligned} u_{Success|Design\ 1} = u_{Success|Design\ 2} > u_{Failure|Design\ 1} = u_{Failure|Design\ 2}, \\ u_{Success,\ Cost|Design\ 1} = u_{Success,\ Cost|Design\ 2} \\ > u_{Failure,\ Cost|Design\ 1} = u_{Failure,\ Cost|Design\ 2}, \end{aligned} \tag{5}$$

where $u_{z\ |\ x}$ is the utility for outcome z given action x. The probabilities in these decision trees are the following:

$$\begin{aligned} t = \Pr(\text{Success}|R_1), \\ q = \Pr(\text{Success}|R_2). \end{aligned} \tag{6}$$

With the insinuation that the best R_1 the player can have is still less than R_2, $t < q$ and thus $\text{Eu}_1 < \text{Eu}_2$ in both treatments as formulated in Fig. 7. This is true despite risk attitude so the claim by Prospect Theory that decision makers become risk taking in the face of a sunk cost is irrelevant. Therefore, rational players in both treatments should choose *purchase new design*, or *Design 2*, if they uphold the belief that $R_1 < R_2$. From the results of Experiment 1, the belief that $R_1 < R_2$ is generally upheld in both treatments such that *Design 2* is chosen.

In the control treatment, players are allowed to experiment with designs such that they can learn first-hand that there is no arrangement that leads to a robustness, R_1, greater than R_2 thus solidifying this belief and compelling players to generally choose *purchase new design*. If the Sunk Cost Effect were present in the sunk cost treatment, players would inflate the probability that they could achieve $R_1 > R_2$ as suggested by Arkes and Blumer [5]. However, players have several turns to attempt to maximize R_1. By the time the decision prompt appears, players see the alternative crew module has a higher robustness, R_2, than any design they had previously created thus mitigating inflation.

Experiment 2

The results of Experiment 2 are peculiar in that treatment does not affect choice, there are no trends in choices in the control treatments, and in both sunk cost treatments, players tend to choose *Yes*. To understand why is to understand the actual decisions being made. The decisions in this experiment should be modeled similarly to those in Fig. 7 save for the choices being *Yes* and *No*. However, the outcome of choosing option *No* is not explicitly defined for the player, except for in the Sunk Cost – Explicit treatment; for example, the prompt shown in Fig. 3 does not discuss an outcome for answering *No*. Therefore, players may have uncertainty as to what will happen if they choose *No*. To capture what players assume is the outcome, they are asked this post-decision; these results are shown in Fig. 8.

The decisions in Experiment 2 are modeled with the two most common outcomes, *buy Design 2* and *mission scrapped*, in Fig. 9; m is the subjective probability, i.e. belief, that Design 2 will be purchased if the in-house design is not continued. The probabilities t and q are defined similarly as in Eq. 6. For players

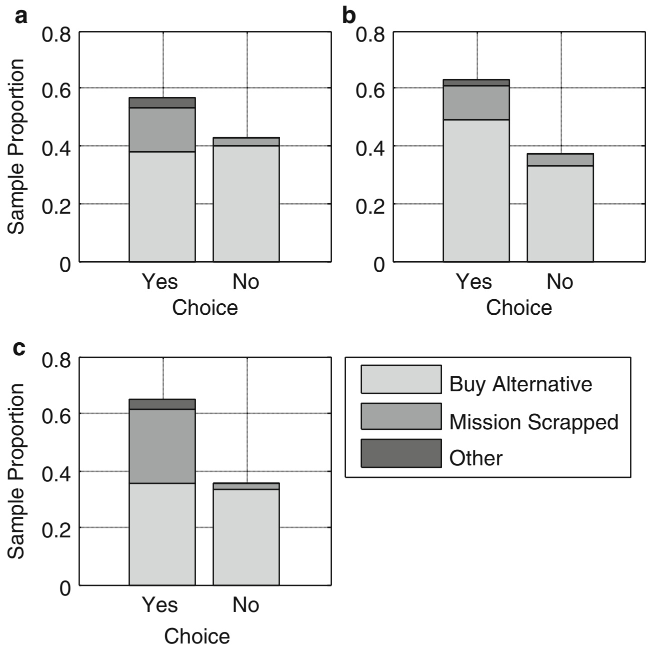


Fig. 8 Perceived outcome of players in **a** control – beginning, **b** control – midgame, and **c** sunk cost – implicit treatments

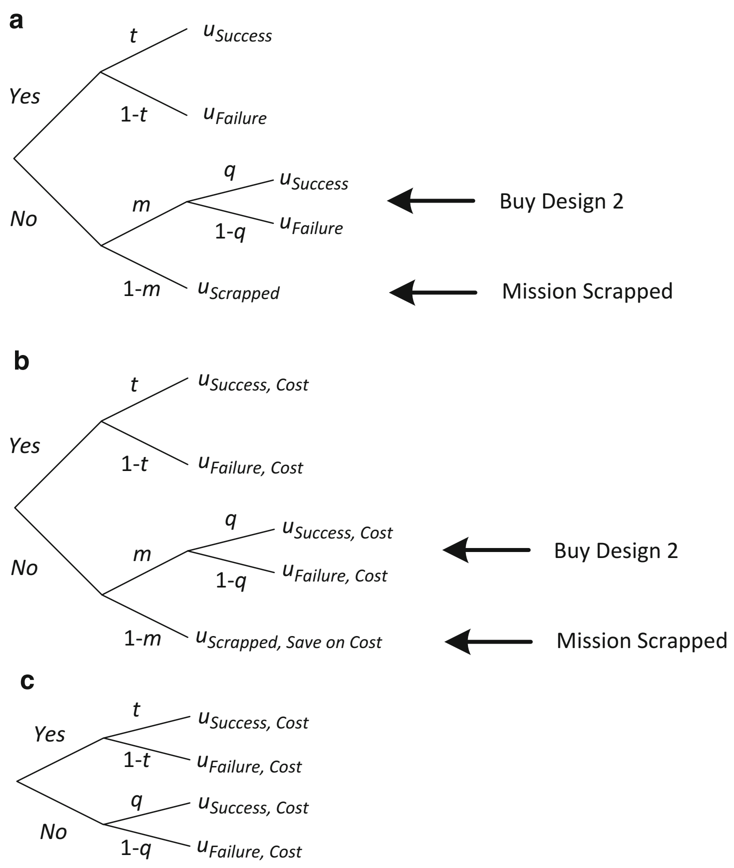


Fig. 9 Experiment 2 decision models: **a** control – beginning, **b** control – midgame, and **c** sunk cost – implicit

believing buying Design 2 is the outcome, i.e. $m = 1$, normative analysis is more or less similar to that of Experiment 1; those believing R_1 cannot be larger than R_2 should choose *No* despite treatment. In the Control – Beginning treatment, those believing $m = 1$ are generally split between choosing *Yes* and *No* insinuating that only approximately half of players upheld the belief that $R_1 < R_2$ for all possible R_1. The same can be said for the Control – Midgame and Sunk Cost – Implicit treatments. The control treatments, in contrast to that in Experiment 1, require players to choose immediately when the decision prompt is given, which does not allow the player to experiment with designs. Without this opportunity, players in these treatments may not solidify that they cannot create a design such that $R_1 > R_2$. As the only factor changing between the sunk cost treatment in Experiment 1 and the Sunk Cost – Implicit treatment is how the decision is framed, perhaps the framing contributes to an inflation of confidence that R_1 can be greater than R_2 that is characteristic of the Sunk Cost Effect.

The decision for the Sunk Cost – Explicit treatment, see Fig. 9c, is exactly the same as the decision for the sunk cost treatment in Experiment 1, see Fig. 7b, as the player is explicitly told that purchasing the alternate design will happen in the event of choosing *No*. However, more players choose *Yes*, corresponding to the *continue my design* option in Experiment 1, than *No*, corresponding to the *purchase new design* option in Experiment 1. Therefore, player do not uphold the belief that $R_1 < R_2$ for all feasible R_1 values as often as in Experiment 1. Although the decision explicitly defines outcomes for the players, the decision is still framed as a *Yes*-*No* question.

Players believing the mission will be scrapped, i.e. $m = 0$, seemingly choose *Yes* more often than *No* despite treatment. These players choosing *Yes* are choosing a riskier option as opposed to the certainty of the mission being scrapped such that

$$tu_{Success} + (1 - t)u_{Failure} > u_{Scrapped} \tag{7}$$

in the control treatments and

$$tu_{Success,Cost} + (1 - t)u_{Failure,Cost} > u_{Scrapped,Save\ on\ Cost} \tag{8}$$

in the Sunk Cost – Implicit treatment. This is not the outcome of a *No* response as the game is designed; a lack of information given to the player and the player's desirability to at least attempt the Mars mission are the driving factors into these players generally choosing to continue the in-house crew module design.

Discussion

The game is designed to convey to the player that they cannot create a crew habitation module with greater robustness than a presented alternative. Since greater robustness leads to greater likelihood of a successful mission, players are expected to choose not to continue with their own design but to adopt the alternative. Through statistical and theoretical analysis of Experiment 1, players in both the control and sunk cost treatments do tend to adopt the belief that R_1 will always be less than R_2. There is a logical reason for this; players in both treatments are afforded the ability to test the belief that R_1 will always be less than R_2 prior to deciding.

Through statistical and theoretical analysis of Experiment 2, players in all four treatments do not generally adopt the belief that R_1 will always be less than R_2 and therefore inflate their confidence that they can achieve a crew habitation module design with a higher robustness than the alternative. Although such inflation is a symptom of the sunk cost effect, the presence of inflation does not necessarily mean players fell susceptible to the sunk cost effect. Results in the two control treatments and the Sunk Cost – Implicit treatment of Experiment 2 are skewed as some of the players did not understand the outcomes properly, i.e. they assumed the mission out be scrapped if they did not continue. These players, in general, prefer to take the chance to reach Mars as opposed to not even trying.

For those in the two control treatments correctly believing the alternative would be purchased if they chose *No*, they do not get a chance to experiment with design to reaffirm the fact that the alternative will always be better. Perhaps being told this appears to be not enough. However, in control treatments of the survey experiments, subjects are simply told facts and they are generally adopted [5, 8, 9]. Therefore, the game environment is likely the cause of this failure to adopt beliefs. Games are inherently thought of as fun, and players most likely want to have fun while playing them. Fun stems from positive experiences, and one such category of positive experience is the use of cognitive competence, or the use of intellectual powers [24–26]. Being told information and accepting certain beliefs in a game environment may not stimulate players' cognitive competence and thus players choose to test the information for themselves.

For those in the Sunk Cost – Implicit treatment believing the alternative would be purchased and those in the Sunk Cost – Explicit treatment, both the decision model and game play are identical to the sunk cost treatment in Experiment 1. As expected from prior studies in sunk cost, it seems players in the Sunk Cost – Implicit treatment fell susceptible to belief inflation characteristic to the sunk cost effect. However, it would have been expected that the Sunk Cost – Explicit treatment in Experiment 2 would yields results similar to the sunk cost treatment in Experiment 1. The only difference is in how the decision is framed to the player. In both sunk cost treatments of Experiment 2, the decision framing is concerned with just one of the alternatives as it asks the player whether to continue an in-house design or not in light of other information, such as information in the Sunk Cost – Explicit treatment that the alternative crew module design is purchasable. Whereas in the sunk cost treatment in Experiment 1, the decision is framed to include both alternatives asking the player which design they prefer to continue. Perhaps this nuance is enough to produce such different behavior.

Framing effects, like the sunk cost effect, are well studied in psychology [27–29]. One framing effect occurs when outcomes to decisions are presented as gains or losses. As mentioned earlier, Prospect Theory states decision makers become risk taking in the face of losses and risk averse in the face of gains. With this in mind, the *yes-no* type framing in the sunk cost treatments of Experiment 2 can perhaps be said to be framed with a loss as a *No* choice results in the loss of, or the loss at a chance for, the in-house design. However, the *either-or* type framing in the sunk cost treatment of Experiment 1 is perhaps more neutral in that *choosing purchase new design* explicitly counters the loss of the in-house module with the gain of the better alternative. The experiments of Garland [8] and Keil et al. [9] ask subjects how likely they are to continue a course of action, which is fundamentally different question than those used in this paper; ultimately, if decision framing is not an important factor, results from Experiment 1 sunk cost treatment, Experiment 2 Sunk Cost – Explicit treatment, and those found in the literature would agree.

Conclusions

In the experiments discussed in this paper, the sunk cost effect is only mitigated when the decision has a clear alternative and the decision is framed as an *either-or* type question. Therefore, the presence of a clear alternative is not the only driving force to mitigation. However, the scenario captured in the gaming environment is still at a high level, i.e. dealing with millions of dollars, similar to previous studies in sunk cost. This leaves the door open for studies in sunk cost at lower levels in the engineering design process. The benefit of adopting the gaming approach to human-studies is that games can quickly be adapted from these high level scenarios to low level scenarios.

The gaming approach to conducting experiments has many nuances from traditional experimental methods. Gaming environments invite an expectation of fun, and players take actions that stimulate their cognitive competence. Games for research wishing to convey information should do so in a manner that takes this into account. The fictional rewards that accompany games have little value in the real world allowing players the freedom to test information given to them. From experimental economics, experiments using hypothetical points and rewards have results with more variability than those with a monetary reward [30]. In future studies involving the conveyance of information, like conveying that $R_1 < R_2$ in this study, the use of a monetary reward may mitigate the need to allow player to reinforce the information through gameplay.

References

1. Von Neumann J, Morgenstern O (2007) Theory of games and economic behavior, 60 Anniversaryth edn. Princeton University Press, Princeton
2. Camerer CF (2011) Behavioral game theory: experiments in strategic interaction. Princeton University Press, Princeton
3. Austin-Breneman J, Honda T, Yang MC (2012) A study of student design team behaviors in complex system design. J Mech Des 134:124–504
4. Vermillion SD et al (2014) Linking normative and descriptive research with serious gaming. Procedia Comput Sci 28:204–212
5. Arkes HR, Blumer C (1985) The psychology of sunk cost. Organ Behav Hum Decis Process 35 (1):124–140
6. Arkes HR, Ayton P (1999) The sunk cost and Concorde effects: are humans less rational than lower animals? Psychol Bull 125(5):591
7. Viswanathan VK, Linsey JS (2013) Role of sunk cost in engineering idea generation: an experimental investigation. J Mech Des 135(12):121002–121002
8. Garland H (1990) Throwing good money after bad: the effect of sunk costs on the decision to escalate commitment to an ongoing project. J Appl Psychol 75(6):728
9. Keil M, Truex DP III, Mixon R (1995) The effects of sunk cost and project completion on information technology project escalation. Eng Manag IEEE Trans On 42(4):372–381
10. Northcraft GB, Neale MA (1986) Opportunity costs and the framing of resource allocation decisions. Organ Behav Hum Decis Process 37(3):348–356

11. Staw BM, Ross J (1987) Behavior in escalation situations: antecedents, prototypes, and solutions. Res Organ Behav 9:39–78
12. Abt CC (1987) Serious games. University Press of America, Lanham
13. Zyda M (2005) From visual simulation to virtual reality to games. Computer 38(9):25–32
14. Deshpande AA, Huang SH (2011) Simulation games in engineering education: a state-of-the-art review. Comput Appl Eng Educ 19(3):399–410
15. Hauge J, Pourabdollahian B, Riedel JKH (2013) The use of serious games in the education of engineers. In: Emmanouilidis C, Taisch M, Kiritsis D (eds) Advances in production management systems. Competitive manufacturing for innovative products and services. Springer, Berlin, pp 622–629
16. Oliveira M et al (2013) Serious gaming in manufacturing education. In: Serious games development and applications. Springer, Cham, pp 130–144
17. Siddique Z et al (2013) Facilitating higher-order learning through computer games. J Mech Des 135(12):121004–121004
18. Ross AM, Fitzgerald ME, Rhodes DH (2014) Game-based learning for systems engineering concepts. Procedia Comput Sci 28:430–440
19. Brandt E (2006) Designing exploratory design games: a framework for participation in participatory design? In: Proceedings of the ninth conference on participatory design: expanding boundaries in design – Volume 1, ACM, Trento, pp 57–66
20. Kosmadoudi Z et al (2013) Engineering design using game-enhanced CAD: the potential to augment the user experience with game elements. Comput Aided Des 45(3):777
21. Sullivan WG, Wicks EM, Luxhoj JT (2000) Engineering economy, vol 12. Prentice Hall, London
22. Thaler R (1980) Toward a positive theory of consumer choice. J Econ Behav Organ 1(1):39–60
23. Kahneman D, Tversky A (1979) Prospect theory: an analysis of decision under risk. Econometrica J Econometric Soc 47:263–291
24. Vorderer P (2001) It's all entertainment-sure. But what exactly is entertainment? Communication research, media psychology, and the explanation of entertainment experiences. Poetics 29(4–5):247–261
25. Ritterfeld U, Cody MJ, Vorderer P (2009) Serious games: mechanisms and effects. Routledge, New York
26. Aldrich C (2009) The complete guide to simulations and serious games: how the most valuable content will be created in the age beyond Gutenberg to Google. Pfeiffer, San Francisco
27. Tversky A, Kahneman D (1981) The framing of decisions and the psychology of choice. Science 211(4481):453–458
28. Nelson TE, Oxley ZM, Clawson RA (1997) Toward a psychology of framing effects. Polit Behav 19(3):221–246
29. Druckman JN (2001) Evaluating framing effects. J Econ Psychol 22(1):91–101
30. Camerer CF, Hogarth RM (1999) The effects of financial incentives in experiments: a review and capital-labor-production framework. J Risk Uncertain 19(1–3):7–42

Using a JPG Grammar to Explore the Syntax of a Style: An Application to the Architecture of Glenn Murcutt

Ju Hyun Lee, Michael J. Ostwald, and Ning Gu

Abstract The two classic computational approaches to design are focussed on either space (syntax) or form (grammar) but rarely combine the two. This paper describes a method that selectively merges aspects of Space Syntax and Shape Grammar into a unique method for accommodating functional relationships into a grammar-based process and thereby providing a syntactic description of an architectural style. This new approach, called a Justified Plan Graph (JPG) grammar, offers both an insight into an architectural style and a way of producing and assessing JPG-based variations of that particular style. The JPG grammar is demonstrated in this paper using the designs of Glenn Murcutt. Seven of the steps in the grammar are described and then the tendency of these rules being applied by Murcutt is calculated. The findings of this paper suggest that the JPG grammar could be used to explore both analytical and generative issues associated with distinct architectural styles.

Node, Link and Shape

Despite acknowledging that architecture is inherently a product of both space and form, the most widely accepted approaches to computational analysis and generation still tend to focus on either one or the other, but rarely both. Space Syntax [1, 2] and Shape Grammar [3, 4] – respectively concerned with space and form – have both been accepted in architectural design and have been the subject of extensive development and application since the 1980s. A few attempts have been made to connect the insights developed separately in these fields but these have tended to use syntax, to understand the grammar used to generate forms. In essence, such combined methods, while using syntactical operations (graph theory mathematics), are largely devoid of any connection to the functional qualities of space, using syntax only to assist in deciding which grammatical rules to apply [5, 6].

For this reason, the present authors have developed a framework which starts with defining spatial or syntactic issues (by way of a variation of convex space

J.H. Lee (✉) • M.J. Ostwald • N. Gu
The University of Newcastle, Callaghan, Australia
e-mail: Juhyun.Lee@newcastle.edu.au

© Springer International Publishing Switzerland 2015
J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_33

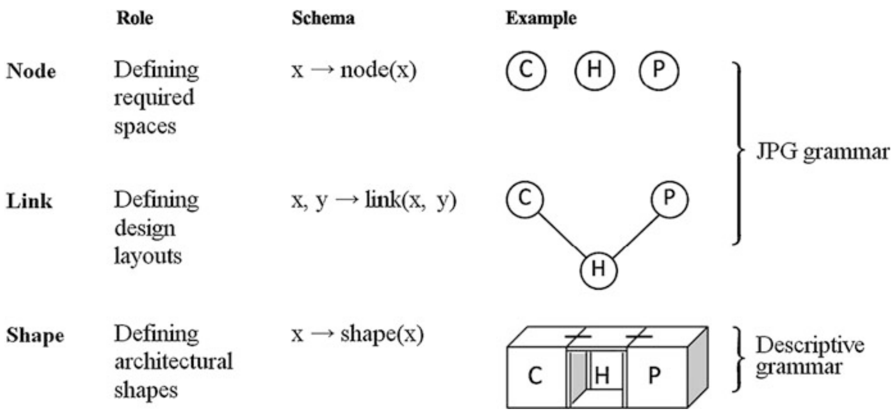


Fig. 1 Three stages of the research framework: node, link, and shape

analysis using a Justified Plan Graph or JPG) and then generates associated forms using a Shape Grammar. This new approach inverts the more common relationship where grammar is privileged over syntax, using syntax (space) for the first time to generate grammar (form) [7]. We hypothesise that the new approach can facilitate a mathematically informed exploration of architectural style. The new framework, called a JPG grammar, commences with three stages wherein *Nodes*, *Links* and *Shapes* are defined (Fig. 1).

The first stage, *Node*, identifies functional spaces and defines them as nodes in a graph. Each node conventionally represents either a convex space or a functionally defined room, but the breath of potential variations of such types are too extensive to be useful and so the framework adopts Amorim's concept of dwelling 'sectors' [8]. A dwelling sector is a zone made up of functionally-related rooms. By grouping such rooms, sector-based nodes are capable of designating initial functional requirements. The second stage, *Link*, defines the connectivity or relationship between each node and configures the functional relationships expressed in this way in a graph. Thus, the combination of sector nodes and links produces the proposed JPG grammar developed here. Finally, the third stage, *Shape*, provides support for shape-based configurational processes that include the development of a descriptive grammar for an architectural style. The JPG grammar is therefore a widely applicable approach which commences with an examination of functional syntactic configuration, before, in the last stage, a style specific, descriptive grammar is applied.

JPG Grammar

A JPG consists of nodes and topological links that are formed by two basic schemas, x → node (x) and x, y → link (x, y) (Fig. 1). Based on these schemas, a JPG grammar can then be developed over several sequential rules corresponding to the generation steps. Most past researchers [3, 9, 10] selectively adopt sequential design steps. For example, Stiny and Mitchell [10] use 8 steps for several of their examples, whilst Hanson and Radford [3] use 12 steps to generate a Murcutt style house. These sequential steps also suggest a possible idealised design process for an architect. However, the process of generating a JPG could also be regarded as a common design stage, that is, a functional zoning process. The seven steps required for this process can be further described in terms of their purpose, rules and examples, each of which are described hereafter (Fig. 2).

In the first step of the JPG grammar, rules identify the functionally required nodes and then locate an exterior node as a carrier. This step defines the topological size of the JPG (the number of nodes) and the different types of nodes in the graph. The second step defines a core node [11]. The grammar distinguishes a core node as an important sector which includes the main entrance so that it directly links to an exterior node, because the core node plays a significant role in configuring spatial programs as well as the forms required for the future design stage (*Shape*) [11].

The third step adds the first set of links starting from a core node to functionally adjacent nodes at the second depth, whilst the fourth step identifies a second set of links starting from a node to functionally adjacent ones at the third depth. These two steps are likely similar, however the division allows for identifying the different topological role of a core node as well as for sequentially forming the tree-like structures of a JPG. With the "check step" (4–1 in Fig. 2), the application of the fourth step confirms that all the required nodes are linked in the JPG.

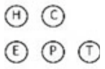
The fifth and sixth steps support the additional configuration of the JPG. A link generated by the fifth step configures a sub-entrance into such a node or a garage sector node at the first depth and the sixth step adds a link between any two nodes as required. These two additional steps can transform a "tree-like" structure to a "ring-like" structure in the JPG. The final step terminates the generation process for the JPG.

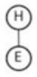
The JPG grammar is significant because it is capable of both rule-based and syntax-based analysis and it facilitates the exploration of the patterns and inequality genotypes of domestic designs [7]. It can also be used to identify design processes that lead to a particular design style, as well as ways of producing variations of that style. In the remainder of this paper an example application of the JPG grammar is presented using the domestic architecture of Glenn Murcutt.

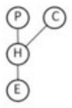
An Application of the JPG Grammar

Glenn Murcutt's architecture, in part because of its apparent simplicity and consistency, has been the subject of both space syntax [12, 13] and shape grammar research [3, 14]. The present application of the JPG grammar investigates a set of ten Murcutt houses built between 1975 and 2005. These houses include works previously analysed as part of both syntactical and grammatical applications. The first part of this section is concerned with the tendency of the applied rules in the ten cases and the second with the generation of a dominant JPG.

	Step	Rule	Example	

1	Create the required nodes	x → node(x)	node(E)
			+ node(H)
			+ node(P)
			+ node(C)
			+ node(T)
				

2	Define a core node * A core node includes a main entrance so that it directly links to an exterior sector node being regarded as a carrier.	E, α → link(E, α), where α is a core node	node(H)c + link(E,H)	

3	Add a first set of links starting from the core node to functionally adjacent nodes at the second depth	α, x → link(α, x),
		where α is a core node
			link(H,P)
			+ link(H,C)
				

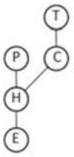
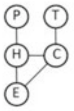
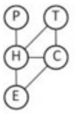
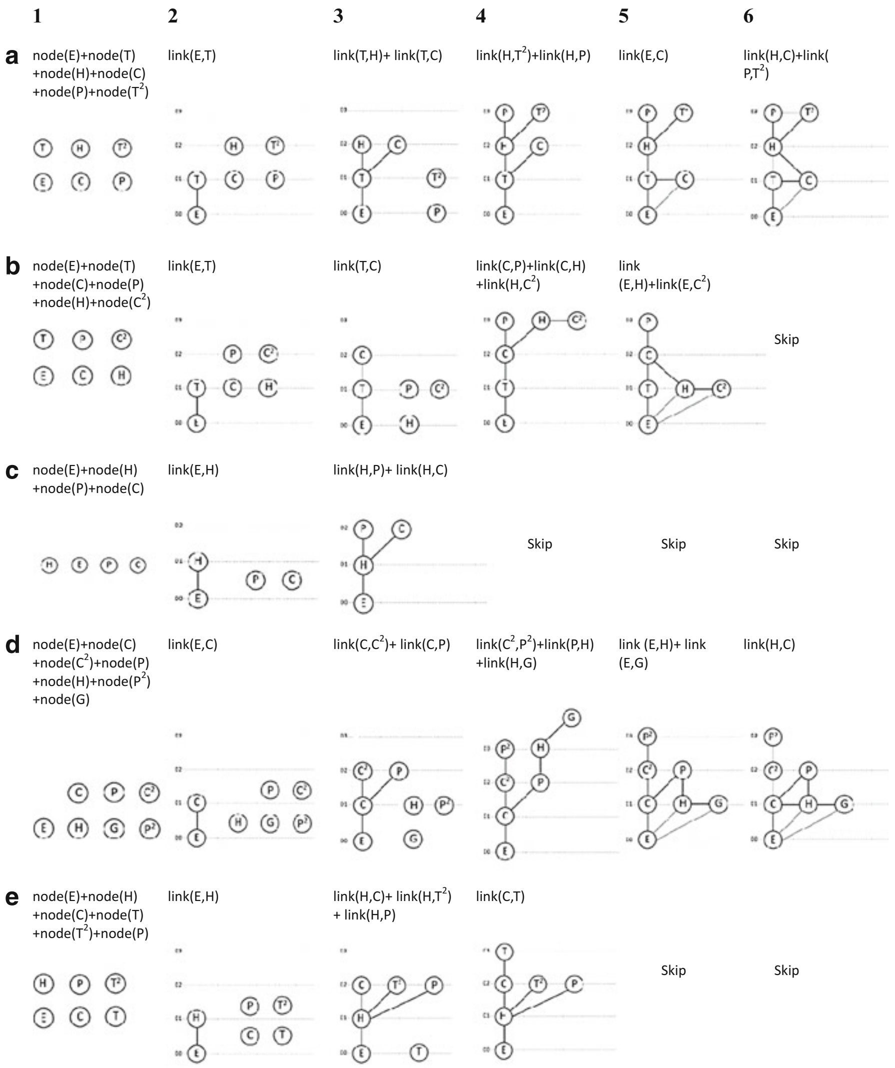
4	Add a second set of links starting from a node to functionally adjacent nodes at the third depth	β, x → link(β, x), where β is a node(s) linking to a required node	link(C,T)	
4-1	Check if all required nodes are linked	If not, apply β, x → link(β, x)		
5	Add additional links between a node and the exterior node (i.e. inserting a sub-entrance into a node or a garage sector node)	E, x → link(E, x)	link(E,C)	
6	Add additional links between any two nodes as required	x, y → link(x, y)	link(H,T)	
7	Termination			

Fig. 2 Sequential steps and rules for the construction of a JPG grammar

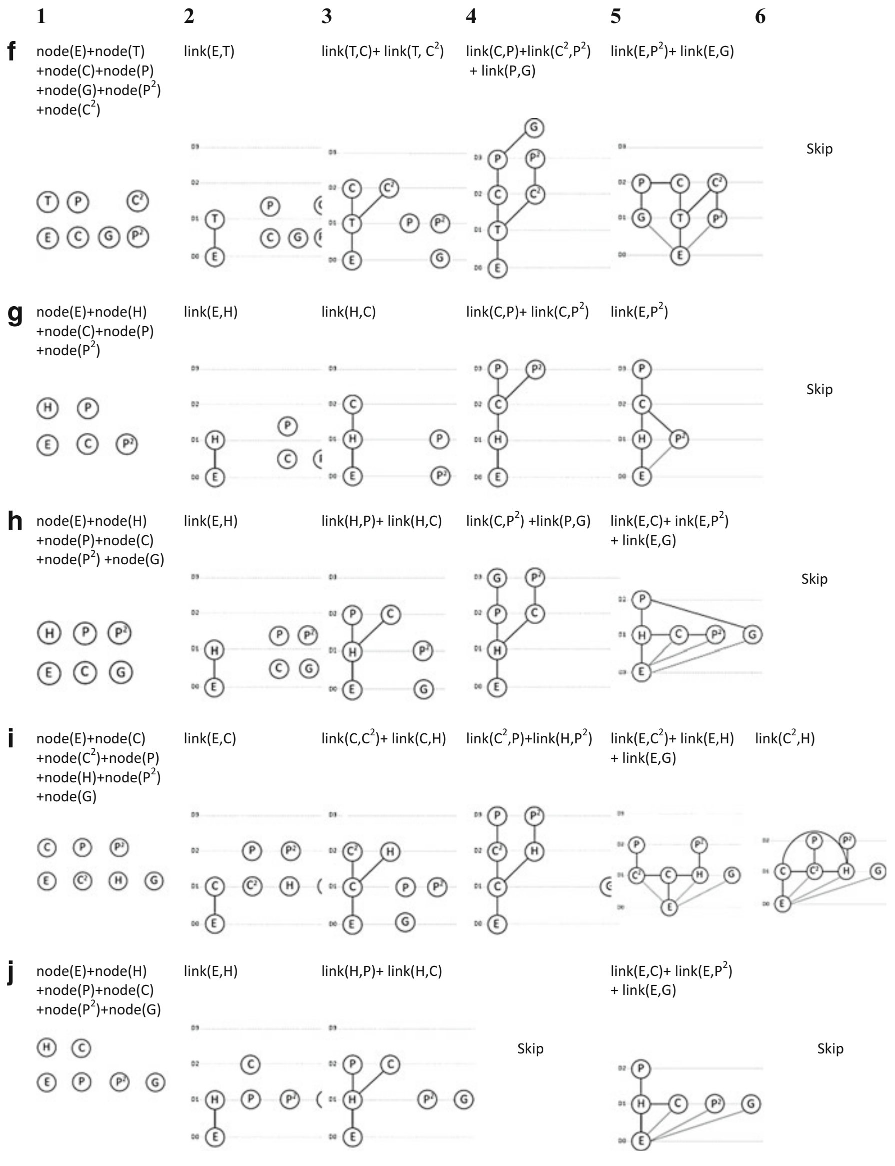
Six functional sectors adopted for exploring Murcutt's domestic architecture are: Exterior, Hall, Common, Private, Transit and Garage. Exterior (E) is simply the site or exterior environment and the starting point of the design generation in a JPG. Hall (H) includes corridors, hallways and linking spaces. Common (C) spaces include living rooms, dining rooms, foyers and kitchens while private (P) contains bedrooms and bathrooms. These four sectors may be common requirements for a domestic building. Transit (T) spaces are intermediate zones between interior and exterior, or occasionally between two interior spaces. A garage (G) is for the storage of cars, but also includes adjacent workshops, laundries and service areas. If there is a second grouping of each sector, it can be represented by a second sector node such as P^2 and C^2.

Figures 3 and 4 illustrate the sequence of graph generation for the ten cases and the applied rules.


1
2
3
4
5
6
a
node(E)+node(T) +node(H)+node(C) +node(P)+node(T^2)
link(E,T)
link(T,H)+ link(T,C)
link(H,T^2)+link(H,P)
link(E,C)
link(H,C)+link(P,T^2)
b
node(E)+node(T) +node(C)+node(P) +node(H)+node(C^2)
link(E,T)
link(T,C)
link(C,P)+link(C,H) +link(H,C^2)
link (E,H)+link(E,C^2)
Skip
c
node(E)+node(H) +node(P)+node(C)
link(E,H)
link(H,P)+ link(H,C)
Skip
Skip
Skip
d
node(E)+node(C) +node(C^2)+node(P) +node(H)+node(P^2) +node(G)
link(E,C)
link(C,C^2)+ link(C,P)
link(C^2,P^2)+link(P,H) +link(H,G)
link (E,H)+ link (E,G)
link(H,C)
e
node(E)+node(H) +node(C)+node(T) +node(T^2)+node(P)
link(E,H)
link(H,C)+ link(H,T^2) + link(H,P)
link(C,T)
Skip
Skip

Key A: Marie Short House, B: Nicholas House, C: Carruthers House, D: Fredericks Farmhouse, E: Ball-Eastaway House

Fig. 3 Sequence of graph generation for the first five cases (1975–1982)


1
2
3
4
5
6
f
node(E)+node(T) +node(C)+node(P) +node(G)+node(P^2) +node(C^2)
link(E,T)
link(T,C)+ link(T, C^2)
link(C,P)+link(C^2,P^2) + link(P,G)
link(E,P^2)+ link(E,G)
Skip
g
node(E)+node(H) +node(C)+node(P) +node(P^2)
link(E,H)
link(H,C)
link(C,P)+ link(C,P^2)
link(E,P^2)
Skip
h
node(E)+node(H) +node(P)+node(C) +node(P^2) +node(G)
link(E,H)
link(H,P)+ link(H,C)
link(C,P^2) +link(P,G)
link(E,C)+ ink(E,P^2) + link(E,G)
Skip
i
node(E)+node(C) +node(C^2)+node(P) +node(H)+node(P^2) +node(G)
link(E,C)
link(C,C^2)+ link(C,H)
link(C^2,P)+link(H,P^2)
link(E,C^2)+ link(E,H) + link(E,G)
link(C^2,H)
j
node(E)+node(H) +node(P)+node(C) +node(P^2)+node(G)
link(E,H)
link(H,P)+ link(H,C)
Skip
link(E,C)+ link(E,P^2) + link(E,G)
Skip

Key F: Magney House, G: Simpson-Lee House, H: Fletcher-Page House, I: Southern Highlands, J: Walsh House

Fig. 4 Sequence of graph generation for the second five cases (1984–2005)

Table 1 The frequency of each node occurring at the first step

Node (E)	Node (C)	Node (P)	Node (H)	Node (P^2)	Node (G)	Node (C^2)	Node (T)	Node (T^2)
10	10	10	9	6	5	4	3	2

Tendency of the Applied Rules

The relative frequency of the rules used to generate the JPG grammar can be recorded within the set of ten designs. This information signifies an architect's tendency to select a particular rule or pattern. Since the JPG grammar is based on both topological and syntactic relationships, the tendency of each rule forms a syntactic type or design instance of Glenn Murcutt's. In order to investigate the tendency, the applied rules at each step of the JPG grammar are initially sorted and grouped. Then, the frequency of each applied rule is calculated. In this way the paper quantifies the tendency to present a dominant JPG through each rule being applied in the given cases.

At the first step of the JPG grammar, rules create the required sector nodes and then locate an exterior sector node as a carrier. Thus, this step defines the topological size of each JPG (the number of nodes) and the types of sectors.

The topological size of each JPG in the cases ranges from four to seven sectors. Five cases consist of six sectors (50 %) and three cases seven sectors. That is, six or seven can be selected as a topological size. Six sectors can be selected using the frequency of each node occurring in the ten cases (See Table 1); that is, node (E), node (C), node (P), node (H), node (P^2) and node (G). If choosing seven nodes, node (C^2) will be added. That is, these selected nodes conform to the application frequency.

In contrast to this determination using only rule-frequency, the design process can also be determined using functional requirements. First, the generation of four sectors (Exterior, Common, Private and Hall) can always be adopted as a fundamental rule and this also conforms to the basic functional sectors for a house. There can also be two additional sectors: garage (G) and transit (T) sectors. If a designer adds a garage sector as a fifth sector node, then the cases tend to accommodate both an additional node (P^2) and node (C^2) so that the topological size of a generated JPG becomes seven. This is because both node (P^2) and node (C^2) are applied to three out of five cases, when a garage sector appears in the ten houses. If a transit sector is considered as the fifth sector of a JPG design, the cases tend to adopt both node (T) and node (T^2), including the four basic sectors, to form six sectors. This is because two of three cases adopt both node (T) and node (T^2). Using these tendencies we can determine three groups of six types of rules that generate sector nodes at the first step of the JPG grammar. That is:

- Rule 1.1.1: $x \rightarrow \text{node}(x)$, where $x = \{E, H, C, P, P^2, G, T\}$; $F = 3$
- Rule 1.1.2: $x \rightarrow \text{node}(x)$, where $x = \{E, H, C, P, P^2, G, C^2\}$; $F = 2$

- Rule 1.2.1: x → node(x), where x = {E, H, C, P, T, T^2}; $F = 2$
- Rule 1.2.2: x → node(x), where x = {E, H, C, P, T, C^2}; $F = 1$
- Rule 1.3.1: x → node(x), where x = {E, H, C, P, P^2}; $F = 1$
- Rule 1.3.1: x → node(x), where x = {E, H, C, P}; $F = 1$

F represents the frequency of each rule applied in the ten cases. The first group, Rule 1.1.x, considers a garage sector node, whilst Rule 1.2.x deals with transit nodes. The last group of rules, Rule 1.3.x, generates four basic sector nodes including a second private space (P^2). Thus, those rules can be further categorised into three:

- Rule 1.1: x → node(x), where x = {E, H, C, P, P^2, G, C^2, T}; $F = 5$
- Rule 1.2: x → node(x), where x = {E, H, C, P, T, T^2, C^2}; $F = 3$
- Rule 1.3: x → node(x), where x = {E, H, C, P, P^2}; $F = 2$

The second step configures a core node. A frequent tendency indicates that a hall sector (H) as a core dominates at the second step (50 %). A transit sector (T) is located as a core in three cases and a common sector in two cases. Applying a transit sector as a core follows Rule 1.2. The rules of the second step are:

- Rule 2.1: E, H → link(E, H); $F = 5$
- Rule 2.2: E, T → link(E, T); $F = 3$
- Rule 2.3: E, C → link(E, C); $F = 2$

The categorisation for the rules at the third and fourth steps is based on considering the next adjacent functional space after the configured nodes at the previous step. The rules at the third step add a first set of links starting from a core node. Two or more links are often generated in the third step so that they sequentially form a tree-like structure of a JPG. Seven cases adopt a pair of links in this step through Rule 3.1 and 3.2, whilst two cases (B, G) adopt only one link. The rules of the third step are:

- Rule 3.1: α, x → link(α, x), where x = P and {C, C^2}; $F = 4$
- Rule 3.2: α, x → link(α, x), where x = {H, C, C^2}; $F = 3$
- Rule 3.3: α, x → link(α, x), where x = only C; $F = 2$
- Rule 3.4: α, x → link(α, x), where x = C, P, T^2; $F = 1$

F represents the frequency of each rule and α represents a core node of a JPG. All the rules at the third step include a link to a common sector. This may not only be a natural feature in domestic buildings, but also a consequent outcome following the results of the second step. Rule 3.3 configures only a link to the common sector, whilst Rule 3.1 includes a link to a private sector and Rule 3.2 generates a link to a hall sector.

A pair of links, link (H, P) and link (H, C), dominates this step using Rule 3.1. This is because the dominant rule set is Rule 3.1 and half of the cases at the previous step use a hall as a core node, which in turn tends to be adopted as a circulation zone. However, the core nodes in two cases (B, G) adopting Rule 3.3 are linked to only a common sector at this step. Thus, the common sector provides the circulation function. Case E adopts Rule 3.4 linking to three sectors at this step, so the core node has the highest integration *i*) and control (*CV*) value. Those cases (B, E, G) can be regarded to generate a specific syntactic structure.

The fourth step defines a second set of links starting from a node to functionally adjacent nodes at the third depth. The purpose of the step is to make all the generated nodes at the first step linked to form a JPG. Two cases (C, J) skip this step, while the total of 15 links are generated in the other eight cases. Following the dominant outcomes of the third step, many links (ten links) are also starting from a common sector. The common and hall sectors may be generally used as a circulation zone that naturally links to the other spaces and often forms a chain or loop. However, it is an interesting finding that the feature is defined sequentially in the JPG grammar.

In addition, the third step adopts a pair of links that transforms the core node into a b-type (tree-type) node, while rules at the fourth step tend to define sectors deeper down that are even located in the fourth depth (See cases B, D, F). This may be one of the features of Murcutt's architecture that it relies on the use of long corridors or passages in parallel to form a circulation loop instead of "a generous circulation loop [12]" in Fig. 5.

The rules also generate more than two separate linear links, but this only occurs in a few cases (A, B, G) where the tree-like link exists at the fourth step. Interestingly, two of the three cases (B, G) also adopt Rule 3.3 at the previous step and consequently configures the circulation node for indoor movements. The rules for generating two links at the fourth step are:

- Rule 4.1: β, x $\rightarrow$ link(β, x), where x = {P, P^2}; $F = 3$
- Rule 4.2: β, x $\rightarrow$ link(β, x), where x = {P, P^2,H}; $F = 2$
- Rule 4.3: β, x $\rightarrow$ link(β, x), where x = {P, T, T^2}; $F = 2$
- Rule 4.4: skip to step 5; F = 2
- Rule 4.5: β, x $\rightarrow$ link(β, x), where x = {P^2, G}; $F = 1$

Here *F* represents the frequency of each rule and β represents a node(s) linking to a configured node(s) at the first step. Rules of the fourth step (except for the "skipping" rule) can define links to private sectors. Rule 4.1 forms two links to private sectors (P, P^2), whilst Rule 4.2 adds a link to a hall sector and Rules 4.3, 4.5, respectively add a link to a transit sector, a garage sector. Each rule is applied to

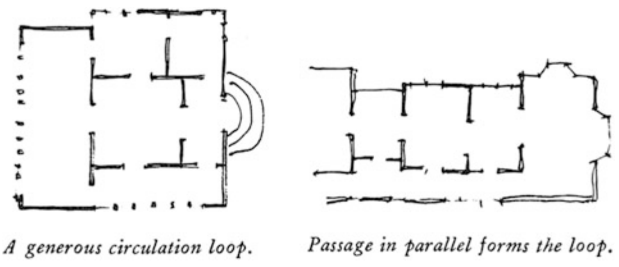

A generous circulation loop.
Passage in parallel forms the loop.

Fig. 5 Circulation loops (Alexander et al. [12]; p. 630)

Table 2 The application frequency of each link at the third and fourth steps. Numbers below each link indicate frequency

Link(H, x)					
Link(H, C)	Link(H, P)	Link(H, T^2)	Link(H, C^2)	Link(H, P^2)	Link(H, G)
6	5	2	1	1	1
Link(C, x)					
Link(C, P)	Link(C, P^2)	Link(C, H)	Link(C, C^2)	Link(C, T)	
5	4	2	2	1	
Link(T, x)			**Link(P, x)**		
Link(T, C)	Link(T, H)	Link(T, C^2)	Link(P, G)	Link(P, H)	
3	1	1	2	1	

make all nodes linked, for example, if only P and P^2 are unlinked nodes after the third step, Rule 4.1 will be applied.

Compared to the third step, β of links generated at this step often consist of two nodes. Thus, we need one more rule to form links. Table 2 shows the links that are generated at the third and fourth steps to construct JPGs of the case of Murcutt's ten houses. Using the table, the rules of the fourth step avoid conflict and do not generate certain links. For example, when β is C and P, Rule 4.5 generates only two links, link (C, P^2) and link (P, G), because there is no link (P, P^2) and link (C,G) in the table.

After the fourth step, cases B, D and F still require one more configuration to link a node at the further depth. For example, case B adopts Rule 4.1 to form two links, link (C, P) and link (C, H) in Fig. 3 and then form link (H, C^2). The additional set (4–1 in Fig. 2) of the fourth step forms two types of links: link (H, G or C^2) and link (P, G) in cases B, D, F. The JPG grammar in this paper considers the rules forming a link to a node at the fourth depth as an additional set of rules, to be dealt with using a separate step when the JPG grammar has to accommodate these additional links at the fourth depth.

The fifth step in the grammar defines a link between a node and the exterior node to configure a sub-entrance or a garage sector node. Thus, the linked node is located at the first depth of a JPG. In eight of Murcutt's houses, five sectors (G, P^2, C^2, C, H) are linked to the exterior at this step. Therefore, these rules identify a link to an exterior sector node by inserting a sub-entrance into a node or a garage sector node at the first depth. The rules are applied to form 16 links in the eight cases. The last of these cases (H, I, J) using Rules 5.1 and 5.2 to add three links at this step. That is, spaces in the later houses tend to connect directly to the exterior sectors so that the exterior sector would become the most integrated space. After this step, many cases also show particular ring-type (c-d-type) nodes. The rules also indicate that common sector and garage nodes often link to the exterior node. Rule 5.1 often generates three links (two of three cases) and Rule 5.2 configures two links. Rule 5.3 generates only one link to the exterior node.

- Rule 5.1: E, x → link(E, x), where x = {G, C, P^2}; $F = 3$ (three links)
- Rule 5.2: E, x → link(E, x), where x = G and {C^2, H}; $F = 2$ (two links)

- Rule 5.3: E, x → link(E, x), where x = C or P^2; $F = 2$ (one link)
- Rule 5.4: skip to step 6; $F = 2$
- Rule 5.5: E, x → link(E, x), where x = {C^2, H}; $F = 1$(two links)

The sixth step in the grammar adds a link between any two remaining nodes. Only three cases (A, D, I) use the sixth step. The particular use of the link (x, y) rule is link (H, C or C^2). Thus, a hall and a common sector node are often linked together. The dominant rule at the sixth step however is skipping to termination.

- Rule 6.1: skip to termination; $F = 7$
- Rule 6.2: H, x → link (H, x), where x = {G, C^2}; $F = 2$
- Rule 6.3: P, G → link (P,G); $F = 1$

The final step terminates the generation of the JPG. By investigating the applied rules which are used at each step, to generate the ten Murcutt houses, we are able to explore the sequential syntactic structure of each JPG. The following section identifies a dominant JPG which encapsulates most spatial relationships presented in Murcutt's architecture.

Dominant JPG and the Syntactic Style

Table 3 indicates the applied rule used to generate the JPG of each case and the dominant rule of each step. Cases H and I, built more recently, may be regarded as a typical instance in terms of the functional relationships of spaces in Murcutt's domestic buildings. This is because many dominant rules are applied in the two cases. In contrast, cases A and B, Murcutt's first two houses in the set, may be a less typical of the whole group. That is, using the frequency of each rule we can generate an idealised JPG that represents the architect's most prevalent syntactic style.

In order to effectively produce the dominant JPG, we need to consider the rules applied at the previous steps. This is because many rules follow the results of the

Table 3 The applied rule to generate the JPG of each case

	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6
Case A	Rule 1.2	Rule 2.2	Rule 3.2	Rule 4.3	Rule 5.3	Rule 6.3
Case B	Rule 1.2	Rule 2.2	Rule 3.3	Rule 4.2	Rule 5.5	Rule 6.1[a]
Case C	Rule 1.3	Rule 2.1[a]	Rule 3.1[a]	Rule 4.4	Rule 5.4	Rule 6.1[a]
Case D	Rule 1.1[a]	Rule 2.3	Rule 3.1[a]	Rule 4.2	Rule 5.2	Rule 6.2
Case E	Rule 1.2	Rule 2.1[a]	Rule 3.4	Rule 4.3	Rule 5.4	Rule 6.1[a]
Case F	Rule 1.1[a]	Rule 2.2	Rule 3.2	Rule 4.1[a]	Rule 5.1[a]	Rule 6.1[a]
Case G	Rule 1.3	Rule 2.1[a]	Rule 3.3	Rule 4.1[a]	Rule 5.3	Rule 6.1[a]
Case H	Rule 1.1[a]	Rule 2.1[a]	Rule 3.1[a]	Rule 4.5	Rule 5.1[a]	Rule 6.1[a]
Case I	Rule 1.1[a]	Rule 2.3	Rule 3.2	Rule 4.1[a]	Rule 5.2	Rule 6.2
Case J	Rule 1.1[a]	Rule 2.1[a]	Rule 3.1[a]	Rule 4.4	Rule 5.1[a]	Rule 6.1[a]

[a]Is the dominant rule of each step

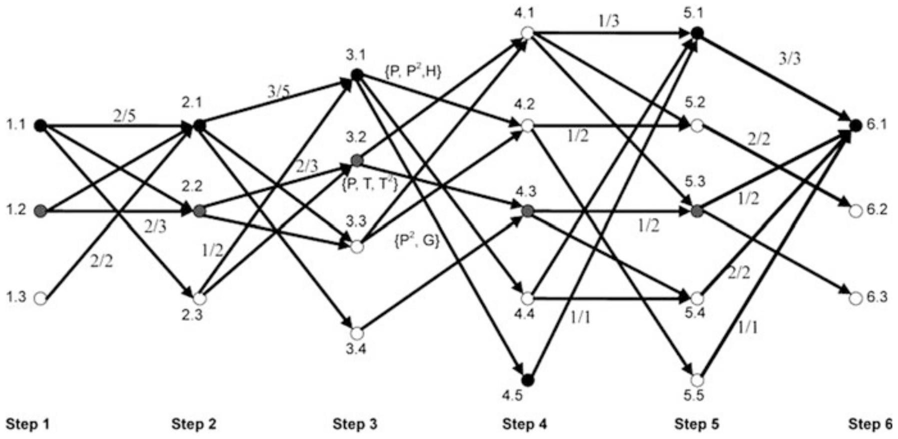


Fig. 6 Dominant paths through the applied rules

previous step as their condition. For example, adopting Rule 1.2 at the first step tends to trigger Rule 2.2 and Rule 2.2 also tends to be followed by Rule 3.2. Thus, it is possible to calculate the mathematical likelihood of certain rule combinations being used in terms of the transition probabilities and conditions (between steps 3 and 4 in Fig. 6). The dominant paths can then be used for generating new JPGs.

Using these dominant paths, it is possible to produce a JPG that most conforms to the key elements of Murcutt's syntactic style. In order to generate such a JPG from the paths, there are three approaches:

1. Follow a dominant path in terms of the transition probability when transiting to the next step,
2. Follow a dominant rule ("upper rule") when each transition probability is the same, (because a "upper rule" is more frequently applied)
3. Refer to the frequency identified in Tables 1 and 2 when the rules are conflicting.

For the purposes of the present paper we adopt the dominant rule approach, Rule 1.1, and selects six nodes, E, H, C, P, P^2, G that are also the most dominant nodes (Table 1). Rule 1.1 is followed by Rule 2.1 at the second step (the transition probability is two fifth). Rule 2.1 generates a link (E, H) so that a hall sector becomes a core node. Rule 3.1 is then applied producing a pair of links, link (H, P) and link (H, C). Since two nodes, P^2 and G, are unlinked, Rule 4.5 is then applied. Rule 5.1 follows the fourth step and configures three links to the exterior, being link (E, G), link (E, C), and link (E, P^2). Finally, through the "skipping" rule – Rule 6.1, a JPG in Fig. 7a is constructed.

Interestingly, this JPG is the same as one of ten cases that were examined, H – the Fletcher-Page House. This implies that this house, better represents the way

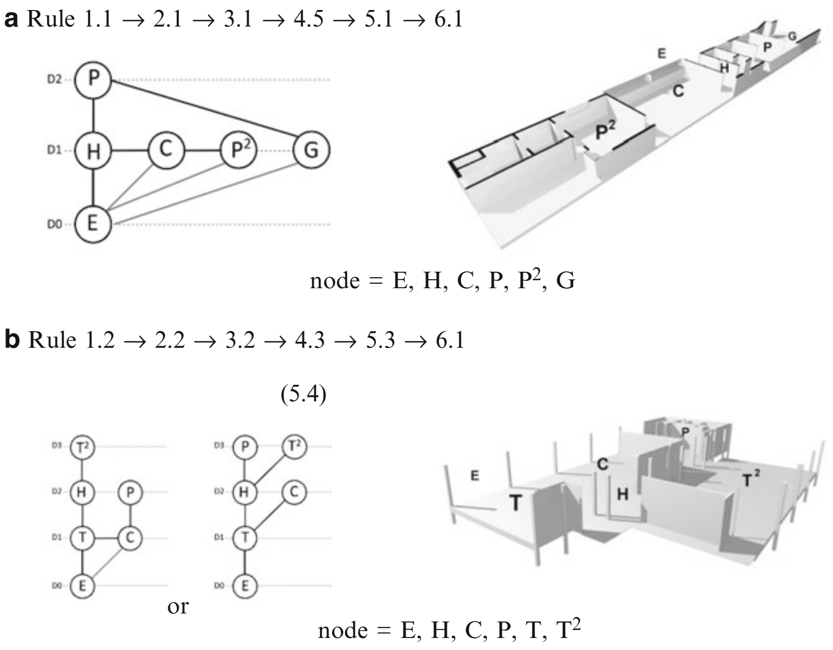

a Rule 1.1 → 2.1 → 3.1 → 4.5 → 5.1 → 6.1
D2
D1
D0
P
H
C
P²
G
E
node = E, H, C, P, P², G
b Rule 1.2 → 2.2 → 3.2 → 4.3 → 5.3 → 6.1
(5.4)
D3
D2
D1
D0
T²
T
or
node = E, H, C, P, T, T²

Fig. 7 Two dominant JPGs generated by the grammar and their corresponding forms

Murcutt creates functional relationship in space, than any of the other cases that were analysed. However, if we were to select the other six nodes including transit sectors – being E, H, C, P, T, T^2 – the outcome is changed. This alternative instance of the idealised application of this rule-set, creates a design which is similar to the form of A – the Marie Short House (See Fig. 7b). Nonetheless, the first JPG is still an ideal application of the rules supporting the generation of a syntactic archetype, because it is based on the dominant tendency identified through the JPG grammar analysis.

Discussion, the JPG and Form

Ultimately, the JPG grammar is not strictly a grammar in the conventional design sense, but instead it is a tool that captures information about the functional and spatial properties of a design, and then allows for the development of a generalised set of rule conditions, which are a reflection of aspects of an architectural style. From such a foundation, further design solutions may be created that conform to the grammar. However, our preliminary conceptual proposal, including the part of the process associated with *shape*, is not demonstrated in this paper. The intention is that the combination of *node* and *link* stages will be continued into the *shape*, stage, using a variety of three dimensional (3D) forms or modular systems (like *Lego* or *Froebel* blocks) [11]. A significant question then, is how to connect the outcome of the JPG grammar, described in this paper, to a corresponding form in the third stage. While a complete explanation of this final stage is beyond the scope of the present paper, the remainder of this section suggests how it will occur.

A possible solution to linking the JPGs and forms in Fig. 7 will be to adopt an inference system based on contextual constraints [15–17]. Past research has considered two types of reasoning methods: rule-based reasoning and case-based reasoning [18, 19]. Adopting the former is effective for producing a case specific solution and for the prioritisation of rules. The application of the JPG grammar to develop a dominant solution, as described in this paper, is understood as the adoption of the case-based reasoning.

In contrast, rule-based reasoning highlights the semantic conditions that can be referred to as design constraints and/or contexts understood through personal knowledge. Combining rule-based and goal-based reasoning is a good example of a rule-based inference system [19]. For example, Coyne and Gero [16, 17] propose the regression of design rules to support the process of chaining backwards and a goal search into the nature of the design process. This allows us to use reasoning about context to effectively produce a suitable form at the *shape* stage. The descriptive grammar at the third stage which generates forms adopts a rule-based inference approach considering syntactic constraints configured by the JPG grammar. In this way we can adopt a context sensitive design grammar [16] taking the general form:

$$u\mathrm{X}v \to u\mathrm{Y}v,$$

where X and Y are lists of attributes and u and v are lists of attributes which remain unchanged by the rule [16]. Through this process it is possible to investigate the forms being applied in Murcutt's domestic architecture in order to define shape-types and properties. The shape grammar may define three or four form properties, e.g. module types (room and hall module), wall types (solid, semi-transparency and transparency), connection types (one, two and three links) and roof types. In this way, schemes, node (H) + link (E,H) + link (H,C), can be transformed to shape (H, room, solid, [E, C]). That is,

$$\mathrm{node(H)} + \mathrm{link(E,H)} + \mathrm{link(H,C)} \to \mathrm{shape(H,\ room,\ solid,[E,C])}$$

Thus, the grammar produces a solid-room-type module linking to the exterior and common sector. To present such a shape grammar at the third stage of the JPG process, further research will be required to extend the grammar to represent semantic properties of the design style. Syntactic rule conditions will also be articulated in future research developing this idea.

Conclusion

Ultimately, the JPG grammar is a widely applicable grammar that can be applied to other styles or to another body of work. Critically, it can be articulated in terms of the frequency of applied rules, a categorisation that allows for the exploration of a particular architectural style or type. While it may possess a level of ambiguity in the way it defines functional sectors rather than actual rooms, by being simplified at the early stage in its development it contributes to a better understanding of the functional relationships implicit in the architecture that is being analysed.

It must be acknowledged that the JPG grammar (Fig. 2) is a sequential or linear process, while the design process undertaken by architects, like Murcutt, would rarely follow such a simple procedure. However, the present demonstration of the JPG grammar is not intended to replace conventional design processes, but rather to provide a better understanding of the syntactic style of architecture and thereby to support academics and professionals. In addition, the linear procedure used for the JPG grammar mirror the approach found in many previous studies [3, 9, 10] selectively adopting sequential design steps. Nonetheless, using a recursive process for certain steps would be worth considering to achieve the syntactic goal of each JPG. For example, the syntax-based analysis of Murcutt's domestic architecture [7] can provide syntactic knowledge and then topological constraints for generating a JPG using the grammar. Such knowledge will allow for an appropriate abstraction for the search for a design [17]. The information about the context can then be used in determining which rule should be applied at each step of the JPG grammar. Thus, while the grammar described in this paper is limited to exploring the syntax of an architectural style and to capturing a dominant JPG, the JPG grammar can accommodate rule-based reasoning processes to support a better automated system for the generation of realistic designs.

Through the application of the JPG grammar to Murcutt's architecture, the study showed that the grammar is capable of effectively exploring architecture in a particular style as well as capturing the possible design processes used to create these famous houses. Finally, identifying the tendency and paths of applied rules enables us to generate an ideal JPG. By using the transition probability and frequency indicators it can be applied to other studies on shape grammar to generate a statistical or stylistic form.

References

1. Hillier B (1999) Space is the machine: a configurational theory of architecture. Cambridge University Press, Cambridge
2. Hillier B, Hanson J, Graham H (1987) Ideas are in things: an application of the space syntax method to discovering house genotypes. Environ Plan B 14:363–385
3. Hanson NLR, Radford AD (1986) Living on the edge: a grammar for some country houses by Glenn Murcutt. Archit Aust 75:66–73

4. Stiny G, Gips J (1972) Shape grammars and the generative specification of painting and sculpture. In: Freiman CV (ed) Information processing 71. North-Holland, Amsterdam, pp 1460–1465
5. Eloy S (2012) A transformation grammar-based methodology for housing rehabilitation: meeting contemporary functional and ICT requirements. Universidade Técnica de Lisboa, Lisbon
6. Heitor T, Duarte J, Pinto R (2004) Combing grammars and space syntax: formulating, generating and evaluating designs. Int J Archit Comput 2:492–515
7. Lee JH, Ostwald MJ, Gu N (2013) Combining space syntax and shape grammar to investigate architectural style: considering Glenn Murcutt's domestic designs. In: Proceedings of 9th international space syntax symposium, Sejong University Press, Seoul, pp 005(1–13)
8. Amorim LMDE (1999) The sectors' paradigm: a study of the spatial and functional nature of modernist housing in Northeast Brazil. Doctoral thesis, University of London
9. Cagdas G (1996) A shape grammar: the language of traditional Turkish houses. Environ Plan B 23:443–464
10. Stiny G, Mitchell WJ (1978) The Palladian grammar. Environ Plan B 5:5–18
11. Koning H, Eizenberg J (1981) The language of the prairie: Frank Lloyd Wright's prairie houses. Environ Plan B 8:295–323
12. Alexander C, Ishikawa S, Silverstein M, Jacobson M, Fiksdahl-King I et al (1977) A pattern language: towns, buildings, construction. Oxford University Press, New York
13. Ostwald MJ (2011) Examining the relationship between topology and geometry: a configurational analysis of the rural houses (1984–2005) of Glenn Murcutt. J Space Syntax 2:223–246
14. Hanson NLR, Radford T (1986) On modelling the work of the architect Glenn Murcutt. Des Comput 1:189–203
15. Coyne RD, Gero JS (1985) Design knowledge and sequential plans. Environ Plan B 12:401–418
16. Coyne RD, Gero JS (1985) Design knowledge and context. Environ Plan B 12:419–442
17. Gero JS, Coyne RD (1985) Logic programming as a means of representing semantics in design languages. Environ Plan B 12:351–369
18. Golding AR, Rosenbloom PS (1996) Improving accuracy by combining rule-based and case-based reasoning. Artif Intell 87:215–254
19. Lee JH, Lee H, Kim MJ, Wang X, Love PED (2014) Context-aware inference in ubiquitous residential environments. Comput Ind 65:148–157

Part IX
Design Ideation

A Step Beyond to Overcome Design Fixation: A Design-by-Analogy Approach

Diana P. Moreno, Maria C. Yang, Alberto A. Hernández, Julie S. Linsey, and Kristin L. Wood

Abstract Design fixation is a phenomenon that negatively impacts design outcomes, especially when it occurs during the ideation stage of a design process. This study expands our understanding of design fixation by presenting a review of de-fixation approaches, as well as metrics employed to understand and account for design fixation. The study then explores the relevant ideation approach of Design-by-Analogy (DbA) to overcome design fixation, with a fixation experiment of 73 knowledge-domain experts. The study provides a design fixation framework and constitutes a genuine contribution to effectively identify approaches to mitigate design fixation in a wide range of design problems.

Introduction

A number of methods have been developed to combat design fixation. Design by Analogy (DbA) has shown effectiveness in generating novel and high quality ideas, as well as reducing design fixation. The present study explores a number of research questions related to design fixation: (1) have the approaches for addressing fixation been presented in a cumulative way, integrated to understand challenges and implications in different fields (2) are there comprehensive metrics to understand and account for fixation; (3) can a better understanding of DbA approaches be developed to manage design fixation analyzing fixation present in transactional

D.P. Moreno (✉) • M.C. Yang
Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: diana_moreno@sutd.edu.sg

A.A. Hernández
Tecnológico de Monterrey, Monterrey, Mexico

J.S. Linsey
Georgia Institute of Technology, Atlanta, GA, USA

K.L. Wood
Singapore University of Technology and Design, Singapore, Singapore

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_34

problems; and (4) does a particular semantic DbA approach, provide domain experts the ability to overcome fixation for transactional problems?

Background and Context

Design Fixation

Definitions of fixation differ with context of design objectives, human activity, or field of knowledge. Examples include memory fixation, problem solving fixation [1], cognitive fixation [2], conceptual fixation, knowledge fixation, functional fixation, operational fixation [3], design fixation [4], and [5].

Design fixation is described as the inability to solve design problems by: employing a familiar method ignoring better ones, self-imposing constraints [6], or limiting the space of solutions by means of developing variants [1, 5], and [7]. A number of causes can contribute to fixation [8] and [9]: expertise [8], designer's unfamiliarity with principles of a discipline or domain knowledge [9], and [10], personality types [11], unawareness of technological advances, or conformity due to proficiency in the methods and supporting technologies of an existing solution [1].

During the design process, design fixation can emerge when example solutions are presented [8, 12–16], when a considerable amount of resources are invested on a potential solution [17], when there are weak or ill-defined problem connections either internally (within elements of the problem) or externally (between the problem and other problems) [18], and when there are more vertical (refined version of same idea) than lateral transformations (moving from one idea to another) [19].

Design fixation research is critical due to its impact on design outcomes and the potential, if mitigated, to improve designers' abilities to generate innovative solutions. Studies from design, engineering, and cognitive science, provide findings across a number of fields, as described below.

Ideation Approaches to Overcome Fixation

Design process success depends highly on ideation stage results [20, 21]. Extensive studies have focused on the improvement of metrics to evaluate ideation processes and associated mechanisms: quality, quantity, novelty (originality), workability (usefulness), relevance, thoroughness (feasibility), variety, and breath [18, 22–30]. Some metrics consider design fixation in a quantitative way; others as a qualitative incidental discovery, measured indirectly through other ideation metrics.

Recent ideation studies show some degree of effectiveness in overcoming design fixation. Based on this information, Table 1 is developed as a framework to

Table 1 Cumulative framework of approaches to overcome fixation

Trigger or source	Implementation method	Method/technique/approach	Reference(s)
Intrinsic	Individual level	Problem re-representation/reframing	[8, 20, 31, 32]
		Enabling incubation	[2, 15, 33–40]
	Group level	Diversify personality type	[8, 11]
	Individual ∩ group	Level of expertise or domain knowledge	[9, 10, 38, 41–44]
Extrinsic	Individual level	Abstract formulation of the problem	[32]
		Use of C-K expansive examples	[12]
		Pictorial examples	[9, 13, 16, 45, 46]
		Audio recorded examples	[47, 48]
		Provide analogies	[15, 49]
		Provide analogies along with open design goals	[16, 50, 51]
		Use of design heuristics	[52]
		Idea generation enabled with computational tools	[8, 18, 53]
		Graphical representations	[13, 46, 54]
		Case-based reasoning and case-based design	[9, 55, 56]
		Use word graphs	[57]
		Word trees	[48, 58, 59]
	Group level	Electronic brainstorming (EBS)	[47, 48]
		6-3-5/C-Sketch	[39, 42, 60]
	Individual ∩ group	Provide de-fixation instructions	[13]
		Develop physical artifacts (prototyping)	[13, 17, 61–63]
		SCAMPER	[38, 39, 64]
		Provide a creative design environment	[38, 65]
		Perform product dissection	[66]
		Develop functional modeling	[8, 67]
Intrinsic ∩ Extrinsic	Group level	Translating the design process into a Linkography	[68]
Intrinsic ∩ Extrinsic	Individual ∩ group	TRIZ	[69]
		Conduct a morphological analysis	[70]

understand, cumulatively, approaches to overcome design fixation. The table is defined by means of two parameters:

- *Trigger or source* provided by the method, which is divided into Intrinsic and Extrinsic.
- *Implementation method* corresponds to the number of designers involved, either Individual or a Group.

There are methods that can be found at the intersection of the parameters presented, for example: functional analysis is an extrinsic method that can be applied individually or as a group.

Intrinsic Approaches

Intrinsic approaches are techniques and methods where ideas are triggered from intuition or previous experience. Problem Re-representation or reframing is a method that increases retrieval cues for analogical inspiration or expands design space exploration [8, 20, 31, 32]. Incubation focuses on disconnecting from the problem by taking a break or performing a non-related task, to access other critical information where insightful ideas may emerge and enabling development of novel or original solutions [2, 15, 33–40].

At the Group level, Diversify personality type relates to the way people prefer to interact with others. This has been found to have an impact in design activities. For example, extroverted persons get more involved in dissection activities that have the potential to increase creativity [8, 11]. Level of expertise or domain knowledge is at the intersection of individual and group level. This attribute emerges with designer's immediate knowledge, and can be expanded when working in teams, by using distant and/or different domain knowledge due to interactions with others [9, 10, 38, 41]. However, some results indicate that novices generate more original concepts [42], while others show that experts consider details in their solutions due to a more evident association between problem and previous knowledge [43]. Due to prior exposition to a wide range of problems, situations and solutions [9], Experts have the ability to frame and break down a problem into more manageable parts [41], and [10], to work with incomplete or ill-defined problems [44], to identify relevant information, patterns and principles in complex design problems [10].

Extrinsic Approaches

Extrinsic approaches are techniques and methods that make use of heuristics, prompts or with stimulus/assistance external to the designer. Abstract formulation of the problem promotes divergent thinking processes and generation of original ideas [32]. Another set of approaches correspond to the use of examples: C-K expansive examples allow exploration of knowledge beyond baseline [12], Pictorial examples allows designers to consider additional design information without constraining the design [9, 13, 16, 45, 46], audio recorded examples enable retrieval of long-term memory concepts and concepts distantly associated, showing a positive impact on the number of original ideas generated [47, 48].

A third set of approaches provide analogies that assist in restructuring the problem and triggering new clues to developed solutions [15, 49]. In addition, open design goals may influence cognitive processes in filtering information that will be then incorporated in the concept solutions increasing originality [16, 50, 51]. Design heuristics promote divergent thinking by providing multiple sequential and/or systematic ways to approach the problem and generate novel and original solutions [52]. Idea generation enabled with computational tools allows alternating among types of problem representation and providing semantic or visual stimulus that will generate more productive ideas [8, 18, 53]. Graphical representations offer a cognitive structure by means of external representation which highlights design complexity, condenses information and enables lateral transformations [13, 46, 54].

Case based reasoning (CBR) and case-based design, which is the application of CBR to design, have roots in analogy reasoning by learning from experience [9, 55]. CBR is used in Artificial Intelligence (AI) to contrast existing experiences/solutions/cases with new unsolved problems to find similarities and extend existing solutions to those new problems. Word graphs [57] and Word trees [48, 58, 59] provide a synergic combination of analogies, semantic and graphical information, computational tools and graphical representations that generate synergic results.

At the Group level, Electronic Brainstorming (EBS) enables interaction between members by a computer interface that prompts sets of ideas for overcoming production blocking [47, 48]. 6-3-5/C-Sketch combines "use of examples," "use of design heuristics" and "use of graphical representations" that provide a sequential structure with visual and textual information [39, 42].

Table 1 considers six approaches at the intersection of individual and group levels: Providing de-fixation instructions makes designers aware of features/elements that should be avoided, overcoming repeating ideas and producing novel ideas [13]. Development of physical artifacts deals with design complexity (mental load). These models represent mental concepts as well as identify and manage fixation features [13, 17, 61, 62]. However, introducing critical feedback during concept generation with prototyping may increase design fixation [63]. SCAMPER is a set of seven brainstorming operator categories that allow problem reframing and increase creativity through the use of analogies and metaphors that expand the design space [38, 39, 64]. A creative design environment is considered an approach to overcome fixation since designers may be motivated by a nurturing and encouraging environment [38, 65]. Product dissection allows "examination, study, capture, and modification of existing products." The method improves form and function understanding to develop new and different ideas [66]. Functional modeling enables functionality representation, to explore alternative means to link customer needs with product function, thus generating novel solution approaches [8, 67].

Two sets of methods are at the intersection of *source* possible levels. At the group level, Linkography translates design process into graphs that represent the designers' cognitive activities [68]. At the intersection of *implementation method* two approaches are: TRIZ, which facilitates solutions by matching contradictions in design problems to design parameters and fundamental principles [69]. A study comparing graphical representations (sketching), control, and TRIZ showed that TRIZ was best in enhancing novelty [70, 71]. Morphological analysis enables generation of new solutions by combining different elements recorded in a matrix of functions versus solution principles per function [70].

The cumulative information presented above provides a better understanding of current approaches as well as implicit opportunities for integration to evaluate possible applications. The presented classification implies the location of new approaches and possible outcomes.

Existing Design Fixation Metrics

This section investigates existing metrics to assess fixation applicable to a broad spectrum of design problems ranging from service to products.

Direct Metrics

These methods inform a designer when fixation is happening and provide a crisp range of understanding for the concept of fixation. Table 2 shows the proposed classification for direct metrics found in the literature. These definitions are coincident to the fixation definition in section "Design Fixation" and enable fixation identification and accountability.

Indirect Metrics

Indirect metrics estimate fixation through indicators, but are not explicitly measured (Table 3). These indicators gauge if a designer is fixated, but do not provide additional information to validate the result.

Design-by-Analogy (DbA) Method

We explore a DbA approach due to its relevance, effectiveness and its potential to have synergic results when integrated with other approaches.

There is evidence that design solutions can be found or adapted from preexisting systems or solutions from other domains [70, 71] for example: astronauts' vortex cooling systems were later adapted as a means to mold and cool glass bottles [72]. Inspiration from analogous domains can be achieved by associations between shared characteristics, attributes, properties, functions, or purposes [73, 74]. Once an association among a design problem and a solution in another domain is established, a solution to the design problem can be developed [41, 58, 75–77].

WordTree [58, 59] and Idea Space System (ISS) [57, 78] are two DbA methods successfully applied in engineering and architecture. These two methods share principles and are based on semantic transformation of textual representations of design problems by means of Princeton's WordNet or VisualTheasaurus which is a

Table 2 Direct metrics classification

Class	Metric(s)	Author(s)
Repeated features	Calculation of fixation percentage. Lower values indicate non-fixated designs	[11]
	$\% \, Fixation: \frac{\# \, of \, similar \, features}{number \, of \, questions \, rated \, by \, the \, coders \, for \, each \, design}$	
	Comparison of the number and percentage of features included in solution to a provided example	[8, 9]
	Obtaining low values for both variety and novelty metrics:	[17]
	$Novelty = 1 - frequency = 1 - \frac{number \, of \, ideas \, in \, a \, bin}{total \, number \, of \, ideas}$ $Variety = \frac{number \, of \, bins \, a \, participant' sidea \, occupy}{total \, number \, of \, bins}$	
	Measurement of functional fixation through dependent measures: (1) frequency of a given functional category at participant level, (2) number of functionally distinct designs, and n novelty* that measures solution uniqueness	[16]
	$*Novelty = 1 - \frac{\# \, of \, functionally \, similar \, designs \, generated \, by \, other \, subjects}{total \, \# \, of \, designs \, for \, all \, subjects}$	
	Originality score (at feature level) and technical feasibility of solutions from a score table. Originality evaluated after comparing features in designs with standard elements. Higher design feasibility corresponds to higher fixation	[42]
	Evaluation of similarity between design brief of the project and the proposed solutions	[63]
Non-redundant ideas	Correlation of the number of non-redundant ideas generated with the total number of unique ideas generated	[48]
	Presence of both low quantity and originality in generated solution. Originality is defined as statistical infrequency of a particular solution, which is a percentage from 0 to 1	[12]

Table 3 Direct metrics classification

Class	Metric(s)	Author(s)
Self-assessment	Commitment to an idea via self-assessment. Surveys ask about perception of fixation reduction, generation of unexpected ideas and workflow improvement	[57]
Design movements	Linkography and Shannon's entropy principle – analyze all possible moves on graph and when moves are interconnected, the ideas are convergent and might be a sign of fixation	[68]
	Goel's type of transformations: vertical and lateral [19]. If more lateral than vertical, fixation can be prevented	[54]
Improvement of a response	Calculation of fixation effects for remote associates test (RAT) subtracting the number of problems solved correctly between fixating and non-fixating stimuli conditions.	[2, 50]
Negative features	Assignation of a discrete value that ranged from 0 to 10 that corresponds to the number of neutral and negative fixation features that were found at given check-in periods	[62]
	Fixation identification as a focus in external features (form) and lower variety	[66]

visual display of the WordNet database. Both methods enable re-representation of the problem and expansion of solution space due to new semantic associations, finding and exploring potential analogies and analogous domains [15, 57–59, 78, 79]. ISS uses a drawing table, a pen that records textual descriptions, sketches, as well as images; and a vertical screen that display wordtrees from WordNet. Using the WordTree method, a designer constructs a diagram of "key problem descriptors (KPDs)," focusing on key functions and customer needs of the given design problem [59]. KPDs are then placed in a tree diagram and semantically re-represented by hypernyms and troponyms selected from WordNet. The WordTree Diagram facilitates associations; therefore, analogies and/or analogous domains can be identified. All analogies, analogous domains and new problem statements can then be used to enrich group idea generation.

Transactional and Product Design Problems

Product design results in tangible artifacts, while transactional design emerges as "Services," virtual products of an intangible nature. Shostack defines services as acts that only exist in time [80]. Vermeulen notes features that differentiate services from products: intangibility, simultaneity of production and consumption, heterogeneity and perishability [81].

Currently, services and products are interconnected to varying degrees and may be considered as part of a continuum. This interconnection implies the potential of tools and methods for conceptual design from engineering and architecture can be

Table 4 Experiment phases and treatments

Treatment	Phase I	Phase II	Sample size	Gender (female/male)
Control	NT	NT	36	11/25
Experimental	NT	WT	37	12/25

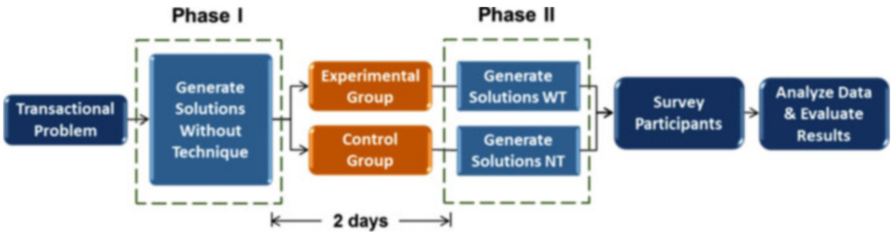


Fig. 1 Experimental execution diagram

transferred to transactional fields to assist idea generation and manage design fixation. It has been stated that early stage of development for services is no different than for physical products and that it is at the detailed design phase where the methods diverge [82], which supports the transferability of design methods between domains.

Experimental Method

Seventy-three transactional process experts were recruited from a professional development program in Mexico. Participants were from a variety of disciplines and involved 22 product and 14 service companies. Domain knowledge expertise was based on professional background and work role.

A transactional design problem was adopted from a previous study [26]: “Reduce overdue accounts/unpaid credits”.

The experiment included a control and experimental treatment, and two phases (Table 4). The control did not specify a method (No Technique – NT) for either phase. Phase I of the Experimental treatment was the same as NT, and Phase II used a DbA method (With Technique – WT). Phase I and phase II were held with 2 days in between, with the same design problem in both. Groups were distributed by background, gender, and other demographics.

In all phases, participants were asked to individually create as many solutions for the transactional problem as possible, recording solutions as text and/or sketch/ diagrams. In Phase I, all participants were given 15 min to generate solutions using intuition alone (Fig. 1).

For phase II, participants were divided into two groups, NT and WT, in separate rooms. NT participants were asked to continue generating solutions without a specific method for 15 min. WT participants were given a 15 min tutorial of the

WordTree DbA method [24] and WordTree software (Thinkmap's Visualthesaurus©). Each participant was provided with a computer with Thinkmap's Visualthesaurus©. Relevant information and graphical associations between words were displayed by the software. Participants were then asked to generate solutions to the transactional design problem using the method and software tools for 15 min.

During phase II, WT participants were asked to select words that re-represented the design problem. The goal was to understand semantic retrieval from the participants' memories that allowed them to switch domains while developing analogous problem statements. Participants were required to list all alternative solutions they developed after extracting useful information from the provided software tools.

At the end of both phases, listed ideas were collected, coded, analyzed and rated by two domain knowledge expert raters. Participants were also asked to fill out a survey after completing each phase.

Analysis

The ideas were sorted into bins of similar ideas. Coding and analyses establish connections to the comprehensive map of approaches to overcome fixation section ("Ideation Approaches to Overcome Fixation") as well as fixation's existent metrics section ("Existing Design Fixation Metrics").

Accounting for Fixation

To compare the results of present study with existing ones, we present our approach that captures the semantic nature of transactional problems. Design fixation was assessed using the procedure outlined by Linsey et al. [8] and Viswanathan and Linsey [83]. The proposed metric elaborates what a repeated idea is for the study and, instead of reporting an absolute value, contrasts this value against the total number of ideas developed. This approach provides a sense of the intensity of design fixation.

A design fixation definition is implemented as shown in Eq. 1:

$$Fixation = \frac{Total\ \#\ of\ repeated\ ideas}{Total\ \#\ of\ generated\ ideas} = \frac{Q_R}{Q_{Total}} = \frac{R_w + R_B}{Q_{Total}} \quad (1)$$

Quantity of Ideas

Three representations are defined: (1) Quantity of total ideas generated (Q_{Total}), (2) Quantity of repeated ideas (Q_R), where a repeated idea occurs when a participant develops a slight variation of a previous idea, and (3) Quantity of Non-repeated ideas (Q_{NR}), which corresponds to the remaining number of Q_{Total} once repeated ideas have been removed.

$$Q_{Total} = \sum all\ ideas\ generated = Q_R + Q_{NR} \tag{2}$$

Equation 2 shows that Q_{Total} can be expressed as the summation of all ideas generated at different levels such as by phase (I, II), experimental group (WT, NT), and participant. Q_{Total} can alternatively be defined as the addition of its two sub-components: Q_{NR} and Q_R.

Two phases of the experiment offer two sources for repeated ideas (Q_R):

- Repeated ideas within a phase (R_W): all repeated ideas across all participants for which frequency (F) is greater than 1.

$$R_{W\,i} = \sum_{j=1}^{b} \sum_{k=1}^{n} F_{ijk} - 1\ \forall F_{ijk} > 1 \tag{3}$$

  where F_{ijk} = frequency of repeated ideas for the ith phase, jth bin, and kth participant; i = phase number (1, 2); b = number of bins (117); n = number of participants. A unit is subtracted from F_{ijk} to maintain accountability of the total of ideas generated.
- Repeated ideas between phases (R_B): all ideas that were generated in Phase I that reappear in phase II at bin and participant levels (Eq. 4).

$$R_B = \sum_{j=1}^{b} \sum_{k=1}^{n} F_{2jk}\ \forall\ F_{1jk} > 1\ \text{AND}\ F_{2jk} > 0 \tag{4}$$

where F_{ijk} = frequency of repeated ideas for the ith phase, jth bin, and kth participant; i = phase number (1, 2); b = number of bins (117); n = number of participants.

Results

Statistical Data Validation

A retrospective power study was performed to validate power of statistical tests and assumptions [84]. For t-tests, power factors were: significance level $\alpha = 0.05$; variability and minimum difference depending on the metric being evaluated; and actual sample sizes of the study ($NT = 36$, $WT = 37$). All power values were higher than 91 % for evaluated metrics, corresponding to a suitable power to perform statistical analysis. Normality of data was evaluated and met using Anderson Darling's Normality Test.

Table 5 Quantity of generated ideas, repeated ideas, and non-repeated ideas

Q_{Total}			
Control Group		Experimental Group	
Ph I	*Ph II*	*Ph I*	*Ph II*
326	328	286	193
t-value= 0.08, *p*-value=0.940		*t*-value=-3.37, *p*-value=0.002	
Anova Ph I		*F*=1.82, p-value=0.182	

Q_R			
Control Group		Experimental Group	
Ph I	*Ph II*	*Ph I*	*Ph II*
45	172	47	52
t-value=6.63 *p*-value=0.000		*t*-value=0.45 *p*-value=0.658	
Anova Ph I		*F=0.00 p-value=0.953*	

+

Q_{NR}			
Control Group		Experimental Group	
Ph I	*Ph II*	*Ph I*	*Ph II*
281	156	239	141
t-value = -4.97 *p*-value = 0.000		*t*-value = -4.19 *p*-value= 0.000	
Anova Ph I		*F=2.75 p-value=0.102*	

Quantity of Ideation

To calculate fixation section "(Accounting for Fixation)", we first determined Q_{Total}, Q_R and Q_{NR}. Table 5 presents quantity of ideas across phases and group levels. Q_{Total} corresponds to 1,133 ideas, including 316 Q_R ideas, and 817 Q_{NR} ideas.

ANOVA (last row Table 5) shows no statistically significant difference in the quantity of ideas generated in phase I for both experimental and control groups. This result is expected because phase I corresponds to an equivalent non-assisted scenario for both groups.

A paired t-test comparing phase I and II for the control group's Q_{Total} shows no statistically significant difference, which is expected because phase II is also non-assisted. A paired t-test comparing phase I and II for the experimental group's Q_{Total} shows a statistically significant difference, consistent with previous cognitive studies where intervention scenarios add significant load due to cognitive processing [16, 85, 86].

For Q_R, a paired t-test comparing phase I and II for the control and experimental groups showed a statistically significant difference in quantity of ideas of control group, which is consistent with literature that design fixation in the form of repeated ideas can be higher if no method is employed [11, 48, 83]. Finally, for Q_{NR}, a paired t-test comparing phase I and II for the control and experimental groups shows a statistical significant difference in the quantity of ideas for both scenarios.

Table 6 summarizes the quantity of repeated ideas. An example of a repeated idea is "impose a penalty" and "make credit performance public." "Impose penalty" was a solution idea stated in phase I and then repeated in phase II by the same person (R_B). "Make credit performance public" was an idea stated by a participant more than once during a single phase (R_W).

Table 6 Repeated ideas by group, source and phase

	WT $_{(n=37)}$		NT $_{(n=36)}$	
Repeated ideas	Ph I	Ph II	Ph I	Ph II
Total R_W	47	24	45	40
Total R_B	0	28	0	132
Total	**47**	**52**	**45**	**172**
Average	1.3	1.4	1.3	4.8

Table 7 Fixation (%) by phases of both groups

Group	Experimental		Control	
Phase	*Ph I*	*Ph II*	*Ph I*	*Ph II*
Fixation (%)	16.4 %	26.9 %	13.8 %	52.4 %

For the NT group, the quantity of R_W is almost the same for both phases, and there is a large quantity of R_B, that is, participants repeated ideas they created in phase I. The WT group reduced by half the number of repeated ideas within in phase II, and the number of ideas participants repeated from phase I was close to half. When studying the average of repeated ideas per participant, a distinctively different value exists from the control group, Phase II. The other three averages were almost identical.

Fixation

Table 7 shows the results of applying Eq. 1 to assess fixation in transactional design problem solving.

No statistically significant difference in the design fixation metric using a two sample t-test is found when comparing phase I of the experimental and control groups (t-value = 0.89, p-value = 0.376). This result may indicate a base level of fixation for non-assisted scenarios. Table 7 shows that the fixation percentage is lower after applying the method (phase II), and a two sample t-test between phase II of the experimental and control groups shows a statistically significant difference in fixation (t-value = −4.33, p-value = 0.000) between both groups.

Discussion and Conclusions

The literature offers several overlapping metrics and indicators for fixation. Direct and indirect metrics were grouped in an attempt to unify metric criteria. A proposed fixation metric builds on previous work to include transactional problems. The results of this study of a transactional problem are comparable with the ones obtained for engineering and architecture, allowing possible generalization of conclusions.

In this study, there was a reduction in total number of ideas for the experimental group which is believed to be a reflection of the load that the applied method adds cognitively. However, analysis of the quantity of repeated ideas shows that the WordTree DbA Method helped overcome fixation to pre-developed solutions as compared to a control. The quantity of non-repeated ideas was reduced in phase II for both the control and experimental conditions, though this may be due to the fact that the experimental group was new to the method and software. More proficiency in the DbA method may increase the quantity of ideas, and merits further studies.

Differences in quantity were translated into fixation percentages revealing that there is a base level of fixation for non-assisted scenarios that remains statistically the same after applying the WordTree DbA Method, while for phase II of a non-assisted scenario, it doubles. These results highlight the efficiency of the WordTree DbA Method as utilized by design experts to effectively manage design fixation.

Would the fixation level during the ideation stage be significantly different using a DbA method compared with a non-assisted scenario? From the study results, there is evidence that in a non-assisted scenario, a significant portion of the allotted time was devoted to developing solutions that are not distinctive from each other (repeated production exceeded the non-repeated), while analogical transfer provided by the WordTree DbA Method enables problem re-representation, exploration of divergent words and effective space solution exploration to solve the problem.

The DbA method used here combines some previous approaches from the proposed cumulative framework to overcome design fixation that supports its strength and robustness. From Table 1, the studied DbA method incorporates elements from different categories. It includes: reframing, by characterizing the problem and problem re-representation. The 2 day break between phases served as an incubation period. It considers expertise that allowed working with incomplete information, framing the problem, identifying relevant information and developing more solutions. It provides and enables analogical exploration. It uses software tools that provide visual representation of the semantic cognitive process allowing problem and solution representation.

The positive results obtained with the experiment are not only aligned with existent research in design fixation, but also with reported results in the psychology field [87]. Leynes et al. [80] found that fixation can be overcome in two ways: first with an incubation period of 72 h, and the second (and closely related to our approach) by presenting alternative semantic information to participants. They found that block and unblock effects occur in different parts of the brain. The results of the present study align with this result because after the semantic stimulation and analogy exploration, the participants were able to overcome design fixation.

Acknowledgements The authors wish to acknowledge the support provided by the ITESM-BMGI Lean Six Sigma Program and the SUTD-MIT International Design Centre (IDC, idc.sutd.edu.sg). Any opinions, findings, or recommendations are those of the authors and do not necessarily reflect the views of the sponsors.

References

1. Luchins A, Luchins E (1959) Rigidity of behaviour: a variational approach to the effect of einstellung. University of Oregon Books, Eugene
2. Smith S, Blankenship S (1991) Incubation and the persistence of fixation in problem solving. Am J Psychol 104:61–87
3. Youmans R, Ohlsson, S (2005) Training produces suboptimal adaptation to redundant instrument malfunction in a simulated human-machine interface. In: Proceedings of the Twenty-Seventh Annual Conference of the Cognitive Science Society, Lawrence Erlbaum Associates, Stresa
4. German T, Barrett H (2005) Functional fixedness in a technologically sparse culture. Psychol Sci 16(1):1–5
5. Jansson D, Smith S (1991) Design fixation. Des Stud 12(1):3–11
6. Youmans R (2007) Reducing the effects of fixation in creative design. ProQuest, Ann Arbor
7. Luchins A, Luchins E (1970) Wertheimer's seminars revisited: problem solving and thinking, vol I. State University of New York at Albany, Albany
8. Linsey J, Tseng I, Fu K, Cagan J, Wood K, Schunn C (2010) A study of design fixation, its mitigation and perception in engineering design faculty. ASME J Mech Des 132(4):041003
9. Purcell A, Gero J (1996) Design and other types of fixation. Des Stud 17(4):363–383
10. Cross N (2004) Expertise in design: an overview. Des Stud 25(5):427–441
11. Toh C, Miller S, Kremer G (2012) Increasing novelty through product dissection activities in engineering design. In: International Design Engineering Technical Conferences (IDETC), ASME, Chicago
12. Agogué M, Kazakçi A, Weil B, Cassotti M (2011) The impacts of examples on creative design: explaining fixation and stimulation effects. In: Eighteenth International Conference on Engineering Design, Copenhagen
13. Christensen B, Schunn C (2007) The relationship of analogical distance to analogical function and pre-inventive structures: the case of engineering design. Mem Cognit 35(1):29–38
14. Jensen D (2010) Effects of an early prototyping experience: can design fixation be avoided? In: Annual Conference & Exposition, American Society for Engineering Education, Louisville
15. Smith S, Linsey J (2011) A three-pronged approach for overcoming design fixation. J Creat Behav 45:83–91
16. Tseng I, Moss J, Cagan J, Kotovsky K (2008) Overcoming blocks in conceptual design: the effects of open goals and analogical similarity on idea generation. In: International Design Engineering Technical Conference/Computers and Information in Engineering, ASME, Brooklyn
17. Viswanathan V, Linsey J (2011) Design fixation in physical modeling: an investigation on the role of sunk cost. In: International Conference on Design Theory and Methodology, Washington, DC
18. MacCrimmon K, Wagner C (1994) Stimulating ideas through creativity software. Manag Sci 40:1514–1532
19. Goel V (1995) Sketches of thought. Bradford-MIT Press, Cambridge, MA
20. Andersson P (1994) Early design phases and their role in designing for quality. J Eng Des 5 (4):283–298
21. Perttula M, Sipilä P (2007) The idea exposure paradigm in design idea generation. J Eng Des 18(1):93–102
22. Girotra K, Terwiesch C, Ulrich K (2010) Idea generation and the quality of the best idea. Manag Sci 56(4):591–605
23. Linsey J, Clauss E, Kortoglu T, Murphy J, Wood K, Markman A (2011) An experimental study of group idea generation techniques: understanding the roles of idea representation and viewing methods. ASME J Mech Des 133(3): 031008-031008-15, doi:10.1115/1.4003498
24. Linsey J, Markman A, Wood K (2012) Design by analogy: a study of the wordtree method for problem representation. ASME J Mech Des (JMD) 134

25. McAdams D, Wood K (2002) A quantitative similarity metric for design by analogy. ASME J Mech Des 124:173–182
26. Moreno D, Hernandez A, Yang M, Otto K, Holta-Otto K, Linsey J, Linden A (2014) Fundamental studies in design-by-analogy: a focus on domain-knowledge experts and applications to transactional design problems. Des Stud 35:232–272, doi:10.1016/j.destud.2013.11.002
27. Oman S, Tumer I, Wood K, Seepersad C (2013) A comparison of creativity and innovation metrics and sample validation through in-class design projects. J Res Eng Des (RED) Spec Issue Des Creat 24(1):65–92
28. Shah J, Kulkarni S, Vargas-Hernandez N (2000) Evaluation of idea generation methods for conceptual design: effectiveness metrics and design of experiments. J Mech Des 122 (4):377–384
29. Shah J, Smith S, Vargas-Hernandez N (2003) Metrics for measuring ideation effectiveness. Des Stud 24(2):111–134
30. Srivathsavai R, Genco N, Hölttä-Otto K, Seepersad C (2010) Study of existing metrics used in measurement of ideation effectiveness. In: ASME IDETC Design Theory and Methodology Conference, ASME, Montreal
31. McKerlie D, MacLean A (1994) Reasoning with design rationale: practical experience with design space analysis. Des Stud 15(2):214–226
32. Zahner D, Nickerson J, Tversky B, Corter J, Ma J (2010) A fix for fixation? Re-representing and abstracting as creative processes in the design of information systems. Artif Intell Eng Des Anal Manuf 24(2):231–244
33. Christensen B, Schunn C (2005) Spontaneous access and analogical incubation effects. Creat Res J 17:207–220
34. Kohn N, Smith S (2009) Partly vs. completely out of your mind: effects of incubation and distraction on resolving fixation. J Creat Behav 43(2):102–118
35. Smith S (1995) Creative cognition: demystifying creativity. In: Hedley C, Antonacci P, Rabinowitz M (eds) The mind at work in the classroom: literacy & thinking. Erlbaum, Hillsdale, pp 31–46
36. Smith S (1995) Getting into and out of mental ruts: a theory of fixation, incubation, and insight. In: Sternberg R, Davidson J (eds) The nature of insight. MIT Press, Cambridge, pp 229–250
37. Smith S, Blankenship S (1989) Incubation effects. Bull Psychon Soc 27:311–314
38. Youmans R, Arciszewski T (2012) Design fixation: a cloak of many colors. In: Proceedings of the 2012 Design, Computing, and Cognition Conference, College Station
39. Vargas-Hernandez N, Shah J, Smith S (2010) Understanding design ideation mechanisms through multilevel aligned empirical studies. Des Stud 31(4):382–410
40. Smith S (1994) Getting into and out of mental ruts: a theory of fixation, incubation, and insight. In: Sternberg R, Davidson J (eds) The nature of insight. MIT Press, Cambridge, pp 229–251
41. Ball L, Ormerod T, Morley N (2004) Spontaneous analogizing in engineering design: a comparative analysis of experts and novices. Des Stud 25:495–508
42. Genco N, Hölttä-Otto K, Seepersaad C (2012) An experimental investigation of the innovation capabilities of undergraduate engineering students. J Eng Educ 101(1):60–81
43. Björklund T (2012) Initial mental representations of design problems: differences between experts and novices. Des Stud 34:135–160
44. Kolodner J (1997) Educational implications of analogy: a view from case-based reasoning. Am Psychol 52(1):57–66
45. Christiaans H, Van Andel J (1993) The effects of examples on the use of knowledge in a student design activity: the case of the 'flying Dutchman'. Des Stud 14(1):58–74
46. Yang M (2009) Observations on concept generation and sketching in engineering design. Res Eng Des 20(1):1–11
47. Dugosh K, Paulus P, Roland E, Yang H (2000) Cognitive stimulation in brainstorming. J Pers Soc Psychol 79(5):722–735
48. Dugosh K, Paulus P (2004) Cognitive and social comparison processes in brainstorming. J Exp Soc Psychol 41:313–320

49. Chrysikou E, Weisberg R (2005) Following the wrong footsteps: fixation effects of pictorial examples in a design problem-solving task. J Exp Psychol Learn Mem Cogn 31(5):1134–1148
50. Moss J, Kotovsky K, Cagan J (2007) The influence of open goals on the acquisition of problem relevant information. J Exp Psychol Learn Mem Cogn 33(5):876–891
51. Moss J, Cagan J, Kotovsky K (2007) Design ideas and impasses: the role of open goals, 16th International Conference on Engineering Design (ICED07). ICED, Paris
52. Yilmaz S, Seifert C, Gonzalez R (2010) Cognitive heuristics in design: instructional strategies to increase creativity in idea generation. Artif Intell Eng Des Anal Manuf 24:335–355
53. Dong A, Sarkar S (2011) Unfixing design fixation: from cause to computer simulation. J Creat Behav 45:147–159
54. Rodgers P, Green G, McGown A (2000) Using concept sketches to track design progress. Des Stud 21(5):451–464
55. Heylighen A, Neuckermans H (2003) (Learning from experience)? Promises, problems and side-effects of case-based reasoning in architectural design. Int J Archit Comput 1(1):60–70
56. Yan W, Zanni-Merk C, Rousselot F, Cavallucci D (2013) Ontology matching for facilitating inventive design based on semantic similarity and case-based reasoning. Int J Knowl-Based Intell Eng Syst 17(3):243–256
57. Segers N, De Vries B, Achten H (2005) Do word graphs stimulate design? Des Stud 26 (6):625–647
58. Linsey J, Wood K, Markman A (2008) Increasing innovation: presentation and evaluation of the WordTree design-by-analogy method, In: ASME Design Theory and Methodology Conference, DETC2008, ASME, New York City
59. Linsey J, Markman A, Wood K (2008) WordTrees: a method for design-by-analogy. In: ASEE Annual Conference, ASEE, Pittsburgh
60. Weaver J, Kuhr R, Wang D, Crawford R, Wood K, Jensen D. (2009) Increasing innovation in multi-function systems: evaluation and experimentation of two ideation methods for design. In: International Design Engineering Conference & Computers and Information in Engineering, IDETC/CIE, ASME, San Diego
61. Yang M (2004) An examination of prototyping and design outcome. In: Design Engineering Technical Conference (DETC), ASME, Utah, pp 1–6
62. Youmans R (2010) The effects of physical prototyping and groupwork on the reduction of design fixation. Des Stud 32:115–138
63. Kershaw T, Hölttä-Otto K, Lee Y (2011) The effect of prototyping and critical feedback on fixation in engineering design. In: 33rd Annual Conference of the Cognitive Science Society CogSci'11, Boston
64. Blosiu J (1999) Use of synectics as an idea seeding technique to enhance design creativity. In: IEEE SMC '99 Conference Proceedings, 1999 I.E. International Conference on Systems, Man, and Cybernetics, IEEE, Tokio, vol 3, pp 1001–1006
65. Chakrabarti A (2013) Understanding influences on engineering creativity and innovation: a biographical study of 12 outstanding engineering designers and innovators. Int J Des Creat Innov 1(1):56–68
66. Grantham K, Okudan G, Simpson T, Ashour O (2010) A study on situated cognition: product dissection's effect on redesign activities, International Design Engineering. ASME, Montreal
67. Little A, Wood K, McAdams D (1997) Functional analysis: a fundamental empirical study for reverse engineering, benchmarking and redesign. In: Design Engineering Technical Conferences 97, ASME, Sacramento
68. Kan J, Gero J (2008) Acquiring information from linkography in protocol studies of designing. Des Stud 29(4):315–337
69. Altshuller G (1999) The innovation algorithm: TRIZ, systematic innovation, and technical creativity. Technical Innovation Center, Worchester
70. Otto K, Wood K (1998) Product evolution: a reverse engineering and redesign methodology. Res Eng Des 10(4):226–243

71. Keller R, Eckert C, Clarkson P (2009) Using an engineering change methodology to support conceptual design. J Eng Des 20(6):571–587
72. Hernández-Luna A, Cárdenas-Franco L (1988) Optimal design of glass molds using CAD/CAE and response surface methodology techniques. Comput Graph 12(3):391–399
73. Gentner D, Markman A (1997) Structure mapping in analogy and similarity. Am Psychol 52:45–56
74. Hey J, Linsey J, Agogino A, Wood K (2007) Analogies and metaphors in creative design. In: Proceedings of the Mudd Design Workshop VI, Claremont
75. Bonnardel N (2000) Towards understanding and supporting creativity in design: analogies in a constrained cognitive environment. Knowl-Based Syst 13(7):505–513
76. Linsey J, Clauss E, Wood K, Laux J, Markman A (2007) Increasing innovation: a trilogy of experiments towards a design-by analogy method. In: ASME 2007 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, IDETC/CIE 2007, ASME, Las Vegas
77. Markman A, Wood K, Linsey J, Murphy J, Laux J (2009) Supporting innovation by promoting analogical reasoning. In: Markman A, Wood K (eds) Tools for innovation. Oxford University Press, New York, pp 85–103
78. Segers N, De Vries B (2003) The idea space system: words as handles to a comprehensive data structure. In: 10th International Conference on Computer Aided Architectural Design Futures, Digital Design – Research and Practice, Dordrecht, pp 31–40
79. Verhaegen P, D'hondt J, Vandevenne D, Dewulf S, Duflou J (2011) Identifying candidates for design-by-analogy. Comput Ind 62(4):446–459
80. Shostack G (1982) How to design a service. Eur J Mark 16(1):49–61
81. Vermeulen P (2001) Organizing product innovation in financial services. Nijmegen University Press, Nijmegen
82. Cagan J, Vogel C (2013) Creating breakthrough products, 2nd edn. Pearson Education, New Jersey
83. Viswanathan V, Linsey J (2012) A study on the role of expertise in design fixation and its mitigation. In: Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE, ASME, Chicago
84. Clark-Carter D (2010) Quantitative psychological research, 3rd edn. Psychology Press, Ney York
85. Chan J, Fu K, Schunn C, Cagan J, Wood K, Kotovsky K (2011) On the benefits and pitfalls of analogies for innovative design: ideation performance based on analogical distance, commonness, and modality of examples. ASME J Mech Des (JMD), 133(8), doi:10.1115/1.4004396
86. Chan J, Fu K, Schunn C, Cagan J, Wood K, Kotovsky K (2011) On the effective use of design-by-analogy: influences of analogical distance and commonness of analogous design on ideation performance. In: International Conference on Engineering Design, ICED, Copenhagen
87. Leynes P, Rass O, Landau J (2008) Eliminating the memory blocking effect. Memory 16 (8):852–872

The Design Study Library: Compiling, Analyzing and Using Biologically Inspired Design Case Studies

Ashok Goel, Gongbo Zhang, Bryan Wiltgen, Yuqi Zhang, Swaroop Vattam, and Jeannette Yen

Abstract We describe an interactive tool called the Design Study Library (DSL) that provides access to a digital library of *case studies* of biologically inspired design. Each case study in DSL describes a project in biologically inspired design from the inception of a design problem to the completion of a conceptual design. The approximately 70 case studies in DSL come from several years of collaborative design projects in an interdisciplinary class on biologically inspired design. Compilation of these case studies enables deeper analysis of biologically inspired design projects. An analysis of some 40 case studies indicates that environmental sustainability was a major factor in about two thirds of the projects. DSL also appears to support learning about biologically inspired design. Preliminary results from a small, formative pilot study indicate that DSL supports learning about the *processes* of biologically inspired design.

Background, Motivation and Goals

Biologically inspired design (also known as biomimicry, or biomimetics, or bionics) is a widespread and important movement in modern design, pulled by the growing need for environmentally sustainable development and pushed by the desire for creativity and innovation in design [1–9]. The paradigm of biologically inspired design espouses the use of analogies to biological systems in generating conceptual designs for new technologies. Although nature has inspired many a designer in the history of design from Sushruta (the "father of surgery" in ancient India) to Leonardo da Vinci to the Wright brothers, it is only over the last generation that the paradigm has become a movement. One prominent example is the design of windmill turbine blades inspired by humpback whale flippers [10]. The designers adapted the shape of the whale flippers—specifically, the

A. Goel (✉) • G. Zhang • B. Wiltgen • Y. Zhang • S. Vattam • J. Yen
Georgia Institute of Technology, Atlanta, GA, USA
e-mail: goel@cc.gatech.edu

© Springer International Publishing Switzerland 2015
J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_35

bumps on their leading edges—to turbine blades, creating blades that improve lift and reduce drag, increasing the efficiency of the turbine [11].

The rapid growth of the movement of biologically inspired design has led to an apparently equally rapid proliferation of courses and programs for learning about the paradigm. For example, Biomimicry 3.8 Institute offers a variety of courses on biomimicry for professional designers. Georgia Institute of Technology offers a sequence of undergraduate courses as well as a graduate course on biologically inspired design. We adopt the perspective of educational science and technology to these courses and programs on biologically inspired design. From our educational science and technology perspective, although the courses on biologically inspired design across different educational programs vary greatly in scope, depth, methodology and pedagogy [12–14], all courses cover two core elements of biologically inspired design: (1) Knowledge, i.e., the content, acquisition, representation, organization, access and use of knowledge of biological and technological systems, and (2) Process, i.e., the methods and practices of biologically inspired design. It is noteworthy that both Baumeister et al. and Yen et al. use case studies in motivating biologically inspired design.

The rapid growth of the movement of biologically inspired design has also led to a rapid proliferation of interactive tools for supporting its practice [15]. For example, Biomimicry 3.8 Institute [16] has developed a publicly available webportal called AskNature that provides access to a functionally indexed digital library of brief textual and visual descriptions of biological systems for generating design concepts [16]. Chakrabarti and his colleagues [17] developed an interactive tool called IDEA-INSPIRE that contains a functionally indexed digital library of multimodal representations (including structured representations) of biological and technological systems for design ideation. Vincent and his colleagues [18] developed BioTRIZ, a biomimetic version of the famous TRIZ system for supporting engineering design. Shu and her colleagues have developed natural language processing techniques for accessing biological analogies relevant to a design problem from biology texts [19]. Nagel et al. [20] describe an engineering-to-biology thesaurus. DANE provides access to a digital library of functionally indexed multimodal representations (including structured representations) of biological and technological systems for idea generation in conceptual design [21, http://dilab.cc.gatech.edu/dane/]. Biologue [22] is an interactive tool for collaborative tagging of biology articles with semantic labels and accessing biology articles based on those semantic tags. It is noteworthy that all of these tools focus exclusively on capturing and providing access to the knowledge elements of biologically inspired design—content, representation and organization of knowledge; they do not cover the process elements—the methods and practices of biologically inspired design.

This raises a basic question: how can we design, build and use an interactive tool that captures the processes of biologically inspired design? This question is difficult to answer in part because at present there is little agreement on the fundamental processes of biologically inspired design. Goel et al. [15], for example, provide an anthology of several different processes that have been developed by different researchers. Baumeister et al. [12], for example, uses a high-level process called the "design spiral." In contrast, Yen et al. [14] use a process for biologically inspired design based on the notions of compound analogies [23], within- and cross-domain analogies [24], problem-driven and solution-based analogies [25], and analogical problem evolution [26].

Our fundamental hypothesis in this paper is that given a specific methodology of biologically inspired design, it is useful to compile a digital library of case studies of biologically inspired design in that methodology. Thus, in this paper, we describe the Design Study Library (or DSL for short), a digital library of about 70 case studies of biologically inspired design from 7 years (2006–2012) of a course on biologically inspired design. We hypothesize that a compilation of case studies would enable deeper analyses of biologically inspired design projects. For example, an open question in biologically inspired design is its relationship with sustainability. We present a preliminary analysis of 42 designs from the perspective of sustainability. Perhaps more importantly, we hypothesize that use of a digital library of case studies such as DSL by novice designers would result in learning about the processes of biologically inspired design. We present a formative pilot study indicating that the use of DSL resulted in an improvement in the novice designers' understanding of biologically inspired design processes. Although preliminary and small scale, the study nevertheless opens the door developing an interactive educational technology for supporting learning about the processes of biologically inspired design.

Biologically Inspired Design Case Studies

All case studies in DSL come from student projects produced over multiple years in a Georgia Tech class on biologically inspired design. The class, ME/ISyE/MSE/PTFe/BIOL 4740, is a yearly, interdisciplinary, project-based undergraduate class. The class is taken by mostly senior level students. During the time period covered by our current collection of case studies (2006–2012) the class was taught jointly by biology and engineering faculty and was composed of students from biology, biomedical engineering, industrial design, industrial engineering, mechanical engineering, and a variety of other disciplines.

During the class, students work in small teams of four to five on design projects. The projects involve identification of a design problem of interest to the team and conceptualization of a biologically inspired solution to the identified problem. Yen et al. [13, 14] provide details of the learning and teaching in this course, including the nature of the design projects.

The Design Study Library

DSL is a web-based, interactive, digital library that allows users to upload, search for, and read design case studies.

Organization

DSL uses a two-level organization scheme as shown in Fig. 1. A project represents the higher level of organization. Currently, DSL contains about 70 projects. A project consists of one or more documents. Together, all the documents in a single project comprise a single design case study. One can view each document as a chapter in the design case study.

Searching

DSL affords three methods for searching for a project or document: Search Files by Author, Search Files Key Word, and Search Files By Tag. All three methods present documents, not projects, as search results. We believe that some users may want to read specific documents of a design case study such as a document

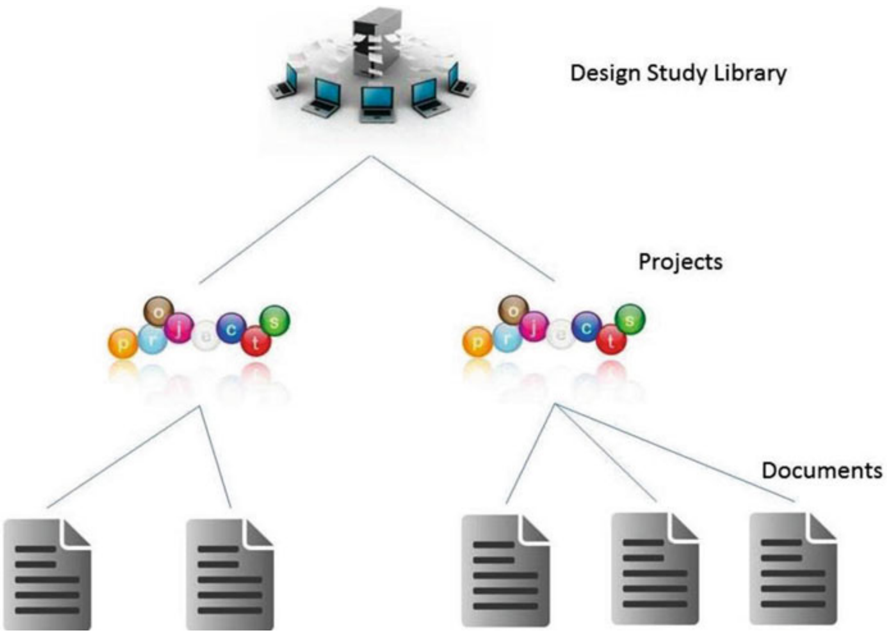


Fig. 1 Two-level organization of the case studies in DSL

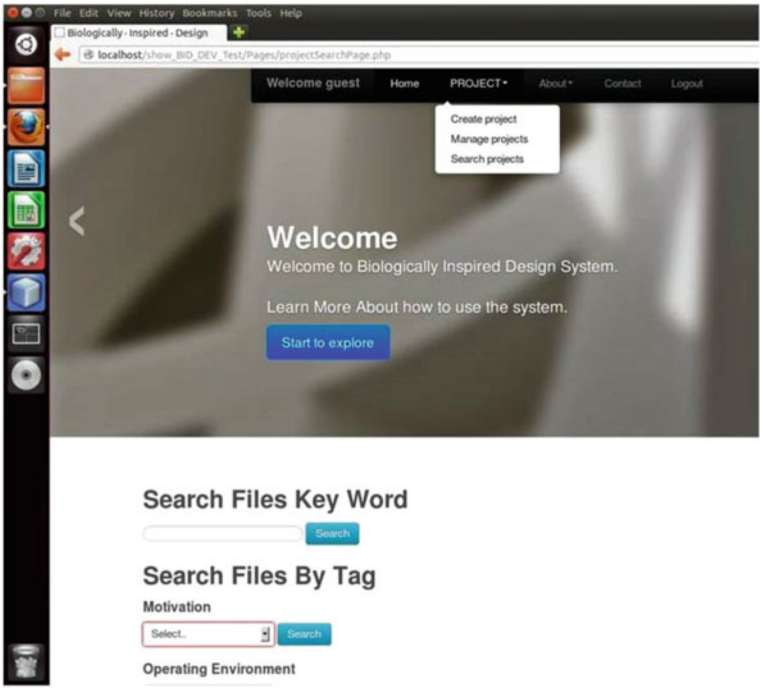

File Edit View History Bookmarks Tools Help
Biologically - Inspired - Design
localhost/show_BID_DEV_Test/Pages/projectSearchPage.php
Welcome guest
Home
PROJECT
About
Contact
Logout
Create project
Manage projects
Search projects
Welcome
Welcome to Biologically Inspired Design System.
Learn More About how to use the system.
Start to explore
Search Files Key Word
Search
Search Files By Tag
Motivation
Select..
Search
Operating Environment

Fig. 2 The search interface to DSL

describing analysis of sustainability. Thus, DSL enables users to go directly to the desired document rather than requiring the user to first start with the project, which would be inefficient.

Figure 2 shows a screenshot containing part of the search screen in DSL. In the Search Files Key Word method, a user can use any string as input. DSL will retrieve MS Word or PowerPoint documents that contain text that matches the input string. In the Search Files By Tag method, a user selects one of the following tag categories from the drop-down menu: Function, Structure, Principle, or Operating Environment [27]. Next, the user provides an input string representing a tag of the selected tag category. DSL will retrieve any document that has a tag of the selected tag category that exactly matches the tag inputted by the user. Because tags must match exactly in this search method, searching by tag may mostly be of use to find files that a user uploaded and tagged himself or herself. Finally, in the Search by Author method, a user selects an author name from the full list of authors of documents within DSL. After selecting an author from the list, DSL will retrieve all documents worked on by that author.

Uploading

DSL is an online library where users can share their design case studies. When uploading a document to DSL, we require a user to follow a two step process in addition to actually uploading the file. In the first step, the user may specify the project related to this document; in the second step, the user may add tags to the document.

Illustrative Examples

In this section, we present small parts of two case studies from DSL. As our examples show, the documents within DSL typically specify the following parts: motivation, problem statement, biological analogue, parts of the design process such as problem decomposition, and the final design.

Desert Chiller

Our first example comes from a project named "Desert Chiller". The following figures and quotes are from a slideshow document related to this project.

Motivation

The following passage from the document describes the motivation of the project.

> The motivation for the design of the Desert Chiller was to try and create a method of preserving food in arid environments without the use of electricity. One of the greatest difficulties associated with food aid is the lack of a good cold-chain, a path along which perishable goods can travel and be stored.

Problem Statement

The following passage from the document describes the problem statement of the project.

> Many nations in sub-Saharan Africa lack electricity. In fact, only 8 % of the population in that region have access to electricity. One of the necessities that nations without power cannot utilize is refrigeration of perishable foods. Residents of these locations are restricted to methods of food preservation that greatly decreases the nutritional value of foods, as well as increases the exposure of the residents to food-borne pathogens.

Biological Analogue

The following passage from the document describes the biological solution that acted as a source of inspiration for the design.

How does nature stay cool? Termite Mounds Provide One Solution, a simple but effective ventilation system. Mounds include flues venting from top and sides. Outside structure effectively captures breezes. And termites open and block tunnels to control air.

Problem Decomposition

Figure 3 is reproduced from the document and represents the problem decomposition.

Tagging the Document

To illustrate the kinds of tags that documents receive in DSL, we note that we tagged this document with the following tags when it was uploaded to DSL. For Structure, it was tagged as “termite mound like structure”. For Function, it was tagged as “reserve food in arid area”. For Principle, it was tagged as “Induced air flow in Ivory Coast termite mounds; The counter-current heat exchanger from Arterio-venous association in the Hind Limb of Birds; Wind capture in black-tailed prairie dog mounds.” For Operating Environment, it was tagged as “desert; outdoor.” We note that the semi-colon here represents a delimiter between tags.

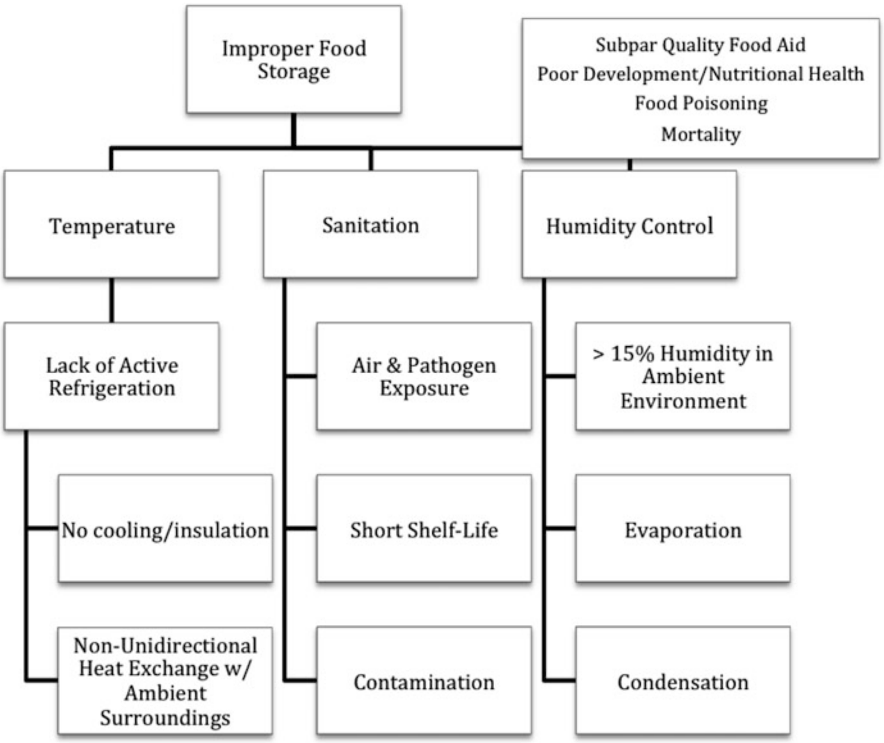


Fig. 3 The problem decomposition in “Desert Chiller” project

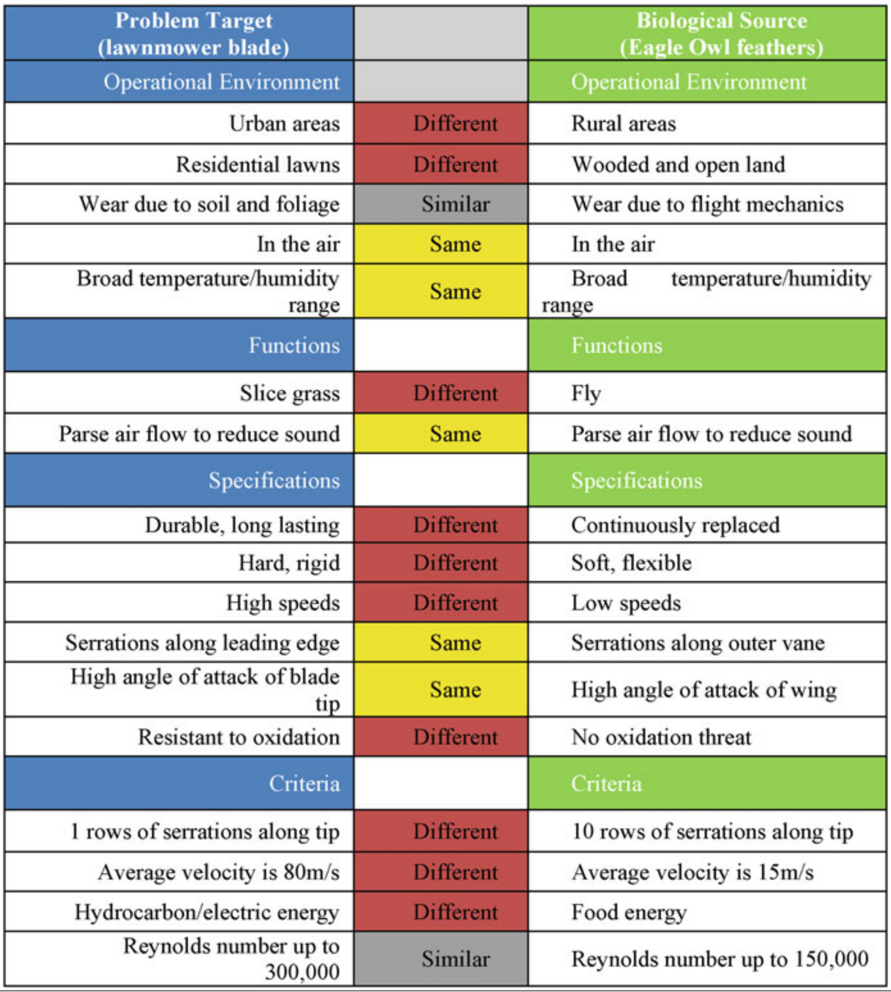
Owl Blade

Our second example is a project called “Owl Blade”. The following quoted passages and Table 1 are reproduced from the text document associated with the project.

Problem Statement and Motivation

The following passage represents both the problem statement and the motivation for this project.

Table 1 A T-chart created by the “Owl Blade” project designers, which compares a problem definition (on the left side) against a biology solution (eagle owl feathers, on the right side). The T-Chart is a method for evaluating biological analogues for design problems [27]



Problem Target (lawnmower blade)		Biological Source (Eagle Owl feathers)
Operational Environment		Operational Environment
Urban areas	Different	Rural areas
Residential lawns	Different	Wooded and open land
Wear due to soil and foliage	Similar	Wear due to flight mechanics
In the air	Same	In the air
Broad temperature/humidity range	Same	Broad temperature/humidity range
Functions		Functions
Slice grass	Different	Fly
Parse air flow to reduce sound	Same	Parse air flow to reduce sound
Specifications		Specifications
Durable, long lasting	Different	Continuously replaced
Hard, rigid	Different	Soft, flexible
High speeds	Different	Low speeds
Serrations along leading edge	Same	Serrations along outer vane
High angle of attack of blade tip	Same	High angle of attack of wing
Resistant to oxidation	Different	No oxidation threat
Criteria		Criteria
1 rows of serrations along tip	Different	10 rows of serrations along tip
Average velocity is 80m/s	Different	Average velocity is 15m/s
Hydrocarbon/electric energy	Different	Food energy
Reynolds number up to 300,000	Similar	Reynolds number up to 150,000

The average lawn mower generates noise between 85 and 90 dB and can be heard from more than a quarter of a mile away. Not only is this a source of annoyance for those in the vicinity, but prolonged exposure to noise levels 85 dB or higher can cause hearing damage.

The problem of excessively noisy lawn mowers is a big problem and a constant source of headache for many living in closely-packed neighborhoods. This problem is amplified by today's poor economy where more people are likely to maintain their own lawns. Hence, there is a good chance more people will perceive the noise as a problem, fueling the need for a genuinely superior solution. Lawn mowers have changed very little in the past few decades. The excessive noise made by these rather crude machines negatively affects not only those operating them, but also those in the general vicinity. Maintaining an aesthetically pleasing lawn is important to many people, and every neighborhood can potentially benefit from a design that reduces the noise emission of lawn mowers.

Operating Environment

The following passage describes the operating environment of the design in this project.

The operating environment of the design solution will be targeted towards neighborhood lawns for the greatest market potential. The majority of families in the United States have a lawn to maintain. The environment will most likely be moist depending on the time of day as grass transpires water vapor. There will be lots of debris flying about during operation, so the design must be durable enough to last for a certain lifespan. Also, the system will be operated by humans, so certain safety factors must be taken into account. For example, the system should be constructed using material(s) that do not shatter upon hard impacts which could potential endanger the user.

Process of Analogical Design

The following passage describes small part of the process of analogical design in this project.

The function is important because noise pollution in many environments, such as neighborhoods, is undesirable. No one wants excessive noise to disrupt their daily activities. As stated previously, lawn mowers can generate noise levels in excess of 90 decibels, which is at the very high end in terms of noise intensity. The function of noise reduction in lawn mowers can be approached from many angles including examining ways to suppress the noise generated by kinetic exhaust gases from the engine. Another possible way to approach the problem would be to investigate different biologically inspired materials for sound and energy absorption. Our particular design focused on controlling the aerodynamics of the blade to reduce noise emission. The solution for this particular function is driven entirely by the lack of existing systems in minimizing noise emission in lawn mowers. This new design just needs to have the additional function of noise reduction while maintaining the normal operating functionality of conventional lawn mower blades.

Biologically Inspired Design and Sustainability

One potential value of compiling case studies is they enable deeper, longitudinal analysis of biologically inspired design. For example, an open question in biologically inspired design is its relationship with sustainability: Is sustainability a primary driver of biologically inspired design?

In Table 2, we present the names of the final student designs from years 2006 to 2010 of the aforementioned biologically inspired design class organized by the year of the class in which they appeared. Also in Table 2, we show the results of our analysis of sustainability as a driver of biologically inspired design. For this analysis, we want to distinguish between two roles of sustainability in biologically inspired design: intentional and incidental. If, based on the documentation in a case study, we thought that sustainability was the primary goal of the design, we say that sustainability plays an intentional role in the design, and it is darkly shaded in the table. For an example of this kind of design, consider the WASP Paper project in 2008. Here, the students designed a paper production system that conserved more water and energy relative to existing methods. If, again based on documentation in a case study, we thought that the final design appeared sustainable compared to conventional designs, we say that sustainability plays an incidental role in the design, and it is lightly shaded in the table. For example, the AF (Antifouling) Armor project in 2007 designed a biofilm-resistant catheter to reduce the number of infections caused by conventional catheters. Although it was not the primary goal of the project, we infer from the documentation in the case study that the improved catheter would be longer-lasting and require less energy to clean than a conventional design. Finally, if we felt that the project fit in neither of these categories, it is unshaded in the table.

Looking at the 42 designs presented in Table 2, 26 of the designs appeared sustainable. Of those 26, 20 had sustainability as its primary goal and 6 incidentally addressed sustainability. However, we note that in the 2008 and 2009 versions, the class projects were specifically aimed at producing sustainable designs. If we consider only the 2006, 2007, and 2010 designs, 10 of the 25 designs appeared sustainable. Of those 10, 5 had sustainability as its primary goal and 5 incidentally addressed sustainability.

Learning About Biologically Inspired Design Processes

To determine if the Design Study Library can help novice designers learn the processes of biologically inspired design, we conducted a pilot study.

Table 2 Case Studies of biologically inspired design and our analysis of their relationship to sustainability

Year	Design name
2006	Abalone armor
	Ant inspired pheromone sensors for traffic control
	BIO_filter
	The eye in the sea
	iFabric: smart, adaptive fabric for thermoregulation applications
	Improving hip implants through protein surface treatment
	The InvisiBoard: a biologically inspired design
	RoboHawk: an aerial bomb detection device
	The shell phone: a biologically inspired design
2007	Applications for color-changing techniques found in nature
	AF (Antifouling) armor
	Biologically inspired designs using the focal adhesion complexes
	Cartilage
	Collision avoidance system
	Proposal for creating bio-inspired behavioral protocols for automatic oil spill cleaning robots
	Safer motorcycle helmet
	SQUID: satealthy quick underwater investigation device
	The xylem: a bioinspired problem based approach
2008	Ballon fog-water collectors
Sustainability-focused semester	The divine flush (Phi Flush)
	Everflo
	FoCoS: fog collection system
	FOOP: filtration of organic particles
	Foro
	PharmaFree: the ecological antibiotical annihilation station
	The urban pearl
	WASP paper: water-saving adhesively-bonded sustainable pulp-molded
2009	BioWat: passively actuated solar water heater/insulator
Sustainability-focused semester	The FireBox
	Flowfex
	From GRAY to GREEN
	Seal skin passive heat flow transfer system
	Septivent
	Solshield
	Tembo 2.0
2010	Desert chiller: passive refrigeration system for regions with limited access to electrical power
	Hydrospot
	Iron-bull bumper system
	Leverage building: modular, scalable, energy efficient building that leverages environment dynamics NOLA for NOLA: novel optimized levee architecture for New Orleans, Louisiana
	Shed-It
	SymPhonic

Method

The pilot experiment consisted of four Georgia Tech graduate students from different schools, none of whom had ever taken a course on biologically inspired design. Each participant followed the same procedure. First, we obtained informed consent from the participant. Next, we gave the participant a pre-test in the form of a questionnaire. After the participant completed the questionnaire, we gave the participant a ten page tutorial document and access to DSL. We then gave the participant 15 min to follow the tutorial and familiarize themselves with DSL.

We then gave the participant a written design problem description, verbally described the design problem to the participant, and told the participant to solve the design problem within 30 min. We told the participant that he or she could use DSL and the web; we also gave the participant scratch paper and a writing utensil to use.

We next asked the participant to verbally describe how he or she solved the design problem. After this step, we had the participant complete a post-test in the form of a questionnaire that had some questions that were identical to ones on the pre-test. This questionnaire asked the participant to reflect on his or her interaction with DSL and give feedback on it.

The Design Problem

All participants were given the same design problem: to fix the overheating problem in solar thermal collectors. Solar thermal collector technology harnesses solar energy by absorbing sunlight and generating heat, which is then used to increase the temperature of the water flowing through the device, providing hot water for residential use. Such a collector is usually composed of a dark flat-plate absorber of solar energy, a transparent cover that allows solar energy to pass through but reduces heat losses, a heat-transport fluid to remove heat from the absorber, and a heat insulating backing. The heat-transport fluid to remove heat from the absorber is usually a mixture of Glycol (55 %) and Water (45 %). Solar thermal collectors are exposed to high temperatures during peak summer and they tend to overheat. Beyond a certain temperature Glycol becomes unstable leading to the problem of degradation, flocculation, formation of solid residues etc. Without adequate protection, this problem could lead to high maintenance costs, damaged internal components, and reduced lifespan of the device. Current protection methods are primitive like blocking the absorber surface with wooden panels, which can lead to loss of efficiency.

Participants were asked to develop a bio-inspired heat regulation system that can fit onto an existing absorber to constantly maintain suitable temperature, and a bio-inspired feedback control system that regulates the temperature of glycol by operating the heat regulation system. We chose this problem because we had studied it earlier in a different context and thus were familiar with it [22].

Results

Our results indicate both opportunities and challenges regarding DSL's potential role as an interactive technology for teaching people the processes of biologically inspired design. On one hand, DSL did influence participants' own design processes. On the other hand, participants did not always seem to view DSL as a tool for learning about the processes of biologically inspired design.

To what extent does DSL influence a participant's knowledge of the biologically inspired design processes? In both the pre-test and the post-test, we asked the participants to define biologically inspired design. We posit that differences in their answers to this question indicate a change in their knowledge of the biologically inspired design processes.

In Table 3, we present a comparison between answers given by a participant for this question on the pre- and post-tests. We have tried to align the answers in terms of design steps (as depicted in the middle column).

As we indicated earlier, none of the participants in our study had any training or experience in biologically inspired design. Some participants had heard of the term biologically inspired design before and/or were able to infer something about the design methodology from the term itself. However, at least one participant had "no idea" about biologically inspired design. Table 4 indicates a substantial change in this participant's understanding of the processes of biologically inspired design as a result of using DSL in addressing a design problem.

Let us consider another question on the pre- and post-tests that also indicates a change in knowledge about the processes of biologically inspired design. The question was as follows: "How do you think "biological inspired design" differs from regular design?" Table 5 summarizes the answers provided by all four participants in the study. It suggests that the participants understanding of the processes of biologically inspired design increased as a result of using DSL to address a design problem.

Table 3 One participant's characterization of biologically inspired design on the pre- and post-tests. The middle column is to help compare the participant's answers

Pre-test answer		Post-test answer
		Compared with the answer I gave before read the reports, I think there should be one more step before design something
Observe the behavior of bio-system and investigate the benefits received by the bio-system from its structure	Step 1	Discover the problem, and understand what the strengths and drawbacks of other existing solutions are
Use the pattern to inspire our design to meet the demand of human beings	Step 2	Brainstorm if bio-system can give us some idea
Design the real product	Step 3	Explore the analogy between that bio-system with our problem
	Step 4	Carried out the design

Table 4 Answers of another participant for the same question as in Table 3

Pre-test answers		Post-test answer
I have no idea		
	Step 1	Identify the problem: limitation of existing design/needs (for new design)
	Step 2	Find similar processes from biology: morphological or functional
	Step 3	Evaluate the technological feasibility of mimicking the biological characteristics
	Step 4	Integrate the design

Table 5 Answers from all four participants to the question "How do you think "biological inspired design" differs from regular design?" This question was asked on both the pre- and post-tests. Each row represents a single participant's answers, and the middle column identifies the participant

Pre-test answers		Post-test answers
The bio-inspired design is more efficient because the bio-systems became the most efficient after thousands of years of evolution. On the other hand, regular design may not be as cost-saving as the bio-inspired design	Participant 1	They're different, but only slightly. The bio-inspired design borrows the concepts from bio-systems. But we're still using modern technique to realize that design
Sorry, I don't know	Participant 2	The biologically inspired design mimics the natural responses from animals or other species. They are more intelligent than regular system
I guess the biological inspired design is more human oriented and more easily to use than regular ones	Participant 3	Biologically inspired design is more easy than regular system because there already are some examples in biological world, if we know this well, we can take this and uses this in our design
It uses "emulation" from biological inspired design functionality, rather than just "sitting there and think"	Participant 4	It gets ideas from biological world to solve the problems that tends to be difficult

However, participant feedback on DSL also suggests that the users may perceive and use DSL only as a search engine for finding solutions to design problems, and not as a tool for learning about the processes of biologically inspired design. Table 6 summarizes the feedback on DSL that we collected during our pilot experiment. Although we had designed DSL as an interactive tool for learning about biologically inspired design, with searching as a function in the service of learning, Table 6 indicates that some participants viewed it only as a search engine. This is consistent of our observations of the participants' actual use of DSL during the experiment. We believe that some of these challenges derive from DSL's interface and the type

Table 6 Our categorization of data from participants on their experience with DSL suggests that some participants perceived DSL as a search engine for finding solutions to design problems rather than a library of design case studies useful for learning about biologically inspired design processes

As a search engine for finding solutions	As a library of case studies
Already been used. Not so complex	The way of DSL organizing things is fantastic
Database "paper" provides detail solution	The documents are well-written and easy to understand
Simple and intuitive	
To my understanding, the software only does one thing, so very basic, less functions make it easy to use	

of interaction participants had with DSL. Evidently we need to redesign DSL's interface and the user interaction with DSL before we conduct our next, larger scale, better controlled, more summative study.

Related Research

This research builds on a several large bodies of literature. The introduction to the paper already situates this work in the literature on biologically inspired design. In this section, we briefly situate the work in other bodies of literature. In particular, the term "case" has come to have different, if related, meanings in different streams of research; our work is related to at least three meanings of a "case."

Firstly, the case study method is a well known and well established research method in the social sciences and the life sciences [28, 29]. A case study is a descriptive study of persons, projects, policies and practices over an extended period of time. Our work clearly makes uses of case studies: each case study in our work pertains to an extended collaborative, interdisciplinary project in biologically inspired design.

Secondly, the case method is a well known and well established method of teaching and learning in business schools [30, 31]. In the case method, the student is given a decision-making problem and put in the position of the decision maker. The expert teachers act as discussants but do not provide any direct instruction. The method typically is supported by large repositories of decision-making cases. The case method is especially well suited for "soft" domains, such as management, for which "hard" rules and models typically are not available [32]. However, the case method is also used in "hard" sciences because the method situates the learning in an authentic context [33]. The case method of teaching and learning is related to the notion of situated learning in cognitive science [34], education science [35], and artificial intelligence [36]. Our work clearly is related to the case method of teaching and learning: instead of formal instruction in a classroom, DSL seeks to support informal cyberlearning.

Thirdly, case-based reasoning is a well known and well established method of analogical reasoning in artificial intelligence and cognitive science [37–39]. Artificial intelligence research on case-based design has entailed the development of digital libraries of design cases in many domains [40, 41]. In the early 1990s, we developed some of the first digital libraries of design cases. The ARCHIE system [42], for example, provided access to a digital library of design cases in the domain of building design in architecture. Each design case in ARCHIE contained a problem description, a solution description, and a critique of the solution; while the problem description and solution critique were represented verbally, the solution was represented both visually and verbally, in the form of annotated design drawings. Similarly, AskJef [43], provided access to a digital library of design cases in the domain of human-machine interface design as in the design of electronic menus in restaurants. Each design case in AskJef contained several versions in the evolution of a design including critiques of each version that contextualized guidelines for designing human-machine interfaces. AskJef was a multimodal system: it used not only verbal descriptions and graphical drawings, but also audio and animation to illustrate the design guidelines.

Finally, InteractiveKritik provided access to a digital library of design cases in the domain of simple mechanical systems. Each design case in InteractiveKritik contained a structure-behavior-function model that explained how the mechanisms of the system achieved its functions. InteractiveKritik also had an autonomous design agent called Kritik that could actually carry out analogical design [44].

Of course, case studies have also been used in biologically inspired design. For example, both Baumeister et al. [12] and Yen et al. [13, 14] use case studies for motivating and illustrating biologically inspired design in the classroom. However, most of the previous interactive tools for supporting biologically inspired design have focused on providing access to biological knowledge and not to the processes of biologically inspired design. Thus both IDEA-INSPIRE [17] and DANE [21] provides access to digital libraries of biological and technological systems, but neither captures much knowledge of biologically inspired design processes. In contrast, the case studies in DSL explicitly capture at least some elements and some aspects of the design process used in the case studies in the section titled Illustrative Examples.

Conclusions

In this paper, we described DSL (for Design Study Library), a digital library of about 70 case studies of biologically inspired design from 7 years of a course on biologically inspired design. Our fundamental hypothesis in this work is that compilation of digital libraries of case studies of biologically inspired design could lead to both new analyses of the paradigm as well as potentially novel techniques for learning about the methodology. We presented a preliminary analysis of about 40 case studies indicating that sustainability considerations in biologically inspired design are of two kinds: intentional, in which designer specifically takes sustainability into account, and incidental, in which sustainability is a byproduct of biologically inspired design. The analysis also indicates that sustainability was a major factor in about half of the case studies. This analysis is illustrative of a kind of analysis that could be done with DSL.

We also presented a preliminary, small-scale pilot study indicating that the use of DSL resulted in some improvement in novice designers' understanding of the process of biologically inspired design. The designers in our study found DSL to be both useful and usable, though they also critiqued DSL for both usefulness and usability. Future work will include expanding the scale of DSL, improving it for both usefulness and usability, and conducting larger scale, better controlled, more summative studies. Although preliminary, small scale, and formative, we believe that the pilot study nevertheless puts us on a path towards developing an interactive, educational technology for supporting asynchronous, distributed, informal cyberlearning about the processes of biologically inspired design.

Acknowledgements We are grateful to the instructors and students of the Georgia Tech ME/-ISyE/MSE/BME/BIOL 4740 class from 2006 to 2012. We thank the US National Science Foundation for its support of this research through a CreativeIT Grant (#0855916) titled "Computational Tools for Enhancing Creativity in Biologically Inspired Engineering Design," and a TUES Grant (#1022778) titled "Biologically Inspired Design: A Novel Biology-Engineering Curriculum." This work has benefited from many discussions with our colleagues at Georgia Tech's Design & Intelligence Laboratory and Center for Biologically Inspired Design, especially Michael Helms, Spencer Rugaber and Marc Weissburg. Also, Titilayo Craig helped with some of the experiments with DSL. Finally, we note that the analysis of the relationship between biologically inspired design and sustainability in this paper has been adapted from Goel et al. [45].

References

1. Bar-Cohen Y (2011) Biomimetics: nature-based innovation. CRC Press, Boca Raton
2. Benyus J (1997) Biomimicry: innovation inspired by nature. William Morrow, New York
3. Bhushan B (2009) Biomimetics: lessons from nature – an overview. Philos Trans R Soc A Math Phys Eng Sci 367(1893):1445–1486
4. French M (1988) Invention and evolution: design in nature and engineering, 2nd edn. Cambridge University Press, Cambridge
5. Gebeshuber I, Gruber P, Drack M (2009) A gaze into the crystal ball: biomimetics in the year 2059. Proc Inst Mech Eng C J Mech Eng Sci 223(12):2899–2918
6. von Gleich A, Pade C, Petschow U, Pissarskoi E (2010) Potentials and trends in biomimetics. Springer, Berlin
7. Shu L, Ueda K, Chiu I, Cheong H (2011) Biologically inspired design. Keynote paper. CIRP Ann Manuf Technol 60(2):673–693
8. Vincent J, Man D (2002) Systematic technology transfer from biology to engineering. Philos Trans R Soc A Math Phys Eng Sci 360(1791):159–173
9. Vogel S (2000) Cat's paws and catapults: mechanical worlds of nature and people. W.W. Norton and Company, New York

10. Fish F, Battle J (1995) Hydrodynamic design of the humpback whale flipper. J Morphol 225:51–60
11. Fish F, Weber P, Murray M, Howle L (2011) The tubercles on humpback whales' flippers: application of bio-inspired technology. Integr Comp Biol 51(1):203–213
12. Baumeister D, Tocke R, Dwyer J, Ritter S, Benyus J (2012) Biomimicry resource handbook. Biomimicry 3.8, Missoula
13. Yen J, Weissburg M, Helms M, Goel A (2011) Biologically inspired design: a tool for interdisciplinary education. In: Bar-Cohen Y (ed) Biomimetics: nature-based innovation. Taylor & Francis
14. Yen J, Helms M, Goel A, Tovey C, Weissburg M (2014) Adaptive evolution of teaching practices in biologically inspired design. In: Goel A, McAdams D, Stone R (eds) Biologically inspired design: computational methods and tools. Springer, London
15. Goel A, McAdams D, Stone R (eds) (2014) Biologically inspired design: computational methods and tools. Springer, London
16. Biomimicry 3.8 Institute (2008) AskNature. http://www.asknature.org/. Accessed on 29 May 2013
17. Chakrabarti A, Sarkar P, Leelavathamma B, Nataraju B (2005) A functional representation for aiding biomimetic and artificial inspiration of new ideas. AIEDAM 19:113–132
18. Vincent J, Bogatyreva O, Bogatyrev N, Bowyer A, Pahl A (2006) Biomimetics: its practice and theory. J R Soc Interface 3:471–482
19. Chiu I, Shu L (2007) Biomimetic design through natural language analysis to facilitate cross-domain information retrieval. AIEDAM 21:45–59
20. Nagel J, Stone R, McAdams D (2010) An engineering-to-biology thesaurus for engineering design. In: Proceedings of the ASME DETC Conference, Montreal, Aug 2010
21. Goel A, Vattam S, Wiltgen B, Helms M (2012) Cognitive, collaborative, conceptual and creative – four characteristics of the next generation of knowledge-based CAD systems: a study in biologically inspired design. Comput Aided Des 44(10):879–900
22. Vattam S, Goel A (2011) Foraging for inspiration: understanding and supporting the information seeking practices of biologically inspired designers. In: Proceedings of the 2011 ASME DETC Conference on Design Theory and Methods. Washington, DC
23. Vattam S, Helms M, Goel A (2008) Compound analogical design: interaction between problem decomposition and analogical transfer in biologically inspired design. In: Proceedings of the Third International Conference on Design Computing and Cognition, Atlanta, pp 377–396
24. Vattam S, Helms M, Goel A (2010) A content account of creative analogies in biologically inspired design. AIEDAM 24:467–481
25. Helms M, Vattam S, Goel A (2009) Biologically inspired design: process and products. Des Stud 30(5):606–622
26. Helms M, Goel A (2012) Analogical problem evolution in biologically inspired design. In: Proceedings of the Fifth International Conference on Design Computing and Cognition. Springer, College Station, July 2012
27. Helms M, Goel A (2014) The four-box method of analogy evaluation in biologically inspired design. In: Proceedings of the ASME DETC Conference on Design Theory and Methods, Buffalo, August 2014
28. Thomas G (2011) How to do your case study: a guide for students and researchers. Sage, Thousand Oaks
29. Yin R (2009) Case study research, 4th edn. SAGE Publishers
30. Hammond J (1976) Learning by the case method. Harvard Business School Press, Boston
31. Ellet W (2007) Case study handbook. Harvard Business School Press, Boston
32. Corey E (1980) Case method teaching. Harvard Business School Press, Boston
33. Herreid C (2005) Because wisdom cant be told: using case studies to teach science. Peer Rev 7:30–31

34. Lave J, Wenger E (1991) Situated learning: legitimate peripheral participation. Cambridge University Press, Cambridge
35. Greeno J (1998) The situativity of knowing, learning, and research. Am Psychol 53(1):5–26
36. Clancey W (1997) Situated cognition: on human knowledge and computer representations. Cambridge University Press, Cambridge
37. Aamodt A, Plaza E (1994) Case-based reasoning: foundational issues, methodological variations, and system approaches. AI Commun 7(1):39–59
38. Kolodner J (1993) Case based reasoning. Morgan Kaufmann, San Mateo, CA
39. Riesbeck C, Schank R (1989) Inside case-based reasoning. Lawrence Erlbaum, Hillsdale
40. Goel A, Craw S (2005) Design, innovation and case-based reasoning. Knowl Eng Rev 20 (3):271–276
41. Maher M, Pu P (eds) (1997) Issues and applications of case-based reasoning in design. Lawrence Erlbaum Associates, Mahwah
42. Pearce M, Goel A, Kolodner J, Zimring C, Sentosa L, Billington R (1992) Case-based decision support: a case study in architectural design. IEEE Intell Syst 7(5):14–20
43. Barber J, Bhatta S, Goel A, Jacobson M, Pearce M, Penberthy L, Shankar M, Simpson R, Stroulia E (1992) AskJef: integrating case-based and multimedia technologies for interface design advising. In: Proceedings of the Second International Conference on Artificial Intelligence in Design (AID-92), Boston, pp 457–476
44. Goel A, Bhatta S, Stroulia E (1997) Kritik: an early case-based design system. In: Maher M, Pu P (eds) Issues and applications of case-based reasoning in design. Erlbaum, Mahwah, pp 87–132
45. Goel A, Bras B, Helms M, Rugaber S, Tovey C, Vattam S, Weissburg M, Wiltgen B, Yen J (2011) Design patterns and cross-domain analogies in biologically inspired sustainable design. In: Proceedings of the AAAI Spring Symposium on AI and Sustainable Design. Stanford University, Palo Alto, March 2011, pp 45–51

A Comparison of Mechanical Engineering and Biology Students' Ideation and Bioinspired Design Abilities

Joanna Tsenn, Daniel A. McAdams, and Julie S. Linsey

Abstract Bioinspired design uses nature as a source of inspiration for creating solutions to engineering design problems. Nature evolves time-tested, efficient designs that may offer an innovative solution. However, it appears that one of the main obstacles to bioinspired design is the engineers' lack of biological knowledge, which causes difficulty in identifying analogous natural systems for the design problems. In this paper, we compare the ability of senior engineering and biology undergraduates to use nature as inspiration for concept generation. The two groups' solutions were analyzed for quantity of non-redundant ideas, quality, novelty, and variety of the solutions. The initial results indicate that there is not a statistically significant difference between the two groups. General trends are examined, and a qualitative study of the results is presented. The overall results suggest that biology coursework does not significantly aid students in identifying analogous biological systems or developing more creative solutions.

Introduction

For over 3,000 years, designers have looked to nature for inspiration to solve engineering design problems. Through billions of years of development, nature has evolved innovative designs. Popular examples of bioinspired designs inspired by natural systems include efficient turbine blades based on a whale's fin [1], self-cleaning paint based on the microstructure of a lotus plant [2], and highly adhesive tape based on a gecko's toes [3, 4]. Bioinspired design has traditionally been conducted on an ad hoc basis with a study of the biological phenomena. Currently a major issue in bioinspired design is that engineers generally lack the knowledge necessary to identify the biological systems relevant to their design problem [5]. By

J. Tsenn (✉) • D.A. McAdams
Texas A&M University, College Station, TX, USA
e-mail: joanna.tsenn@gmail.com

J.S. Linsey
Georgia Institute of Technology, Atlanta, GA, USA

© Springer International Publishing Switzerland 2015
J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_36

increasing the available biological knowledge base, the probability of finding a suitable source of inspiration should also increase.

In the study presented in this paper, we are interested in the effectiveness of a biologist in accessing their knowledge of nature to generate ideas and concepts to solve engineering-type problems. In short, does a general knowledge of nature have a significant impact on the usage of nature as inspiration for problem solving? To study the differences between mechanical engineers and biologists, a controlled experiment was run with senior-level undergraduate students from both classifications. The students participated in a design activity in which they were each given a design problem and 50 min to generate solutions.

The paper covers related work and positions this experiment within that context in the background section. It then describes the experiment in further detail and the hypotheses of the study are presented. The paper then examines and discusses the results, which were evaluated both quantitatively and qualitatively. The results were evaluated quantitatively using the ideation effectiveness metrics of quantity, quality, novelty, and variety of the solution sets. A qualitative comparison of the solutions developed by both groups of participants was also completed. The experiment allows us to look into the effects of general biological knowledge on the ability of finding analogous natural designs to engineering design problems.

Background

Biological systems are examined for potential sources of inspiration since natural systems often follow principles of efficiency, adaptability, and multi-functionality [6]. Looking to nature is often cited as a good source of inspiration, which is recommended by many design textbooks [7–10]. One important item to note is that the natural world should serve as inspiration for function or behavior through the study of aspects of the natural phenomenon, rather than mimicking or exactly copying a natural solution [11]. Since Benami et al. found that long-distance analogies result in more original solutions, bioinspired designs' usage of distant analogies can help produce more innovative solutions [12].

Cross-domain analogies are needed to bridge the biological and engineering domains to produce bioinspired solutions, which means that the designer needs to be familiar with both domains. Biological knowledge is required in order to find a relevant analogy, but most engineers do not have the necessary expertise. As such, biologists are sometimes recruited to aid in finding a solution [13]. This paper is interested in how effective a biologist would be in identifying potential biological analogies for an engineering design problem.

Differences in Design and Analogy Usage for Biologists and Engineers

An important part of bioinspired design is the use of a distant domain analogy. Biologists and engineers use analogies differently in real-world situations. A study was completed by Dunbar examining the use of analogy in different molecular biology labs [14]. He discovered that analogies are used frequently within lab meetings, but most of the analogies remained "within organism" or "other organism" rather than looking to more distant domains that were "non-biological". On the other hand, a study completed by Christensen and Schunn found that engineers use "within domain" analogies and the more distant "between domain" analogies a similar amount [15]. They completed their real-world study by studying product development meetings of design engineers at a major international company that does engineering design of medical plastics. A comparison of the two groups shows that distant domain analogies are used more frequently during engineering design than during biology lab work. The analogies were also used differently when comparing the biologists and the engineers. Almost half of the biologists' analogies were used to "provide an explanation", whereas only 32 % of the engineers' analogies were used for that purpose. Engineers use a majority of their analogies for solving problems.

In terms of general design perspectives, Vattam noted that biology and engineering students focus on different aspects when given a design challenge. The engineers focus more on the structural issues on a macro-scale level while the biologists consider the systems and interactivity among the systems and often think at a microscopic level [16]. These differences are observed even when considering the overall professions. Engineers try to understand structure and function in order to develop technological innovations and applications, while biologists gather information with the aim of better understanding the principles of living systems [17]. These different thought systems may influence how the groups access and develop analogies.

Bioinspired Methods

Other methods and techniques have been developed to aid a design engineer that lacks a strong biological knowledge base with bioinspired design. One of the simplest methods is the directed method and only requires that the designer consider how nature might solve a given engineering design problem. This method does require knowledge of biology, which makes it a popular method for biologists in biomimetic research. Anecdotal evidence suggests the directed method to be a viable option for bioinspired design [18, 19].

Other formalized methods include BioTRIZ, function-based bioinspired design, and the usage of dedicated search tools. BioTRIZ is an extension of TRIZ, which has a framework that allows an abstraction of a design in order to easily access solution principles from other disciplines [20]. BioTRIZ was developed following the study of about 500 biological phenomena and 2,500 biological conflicts and their resolutions in order to develop a new matrix similar too and derived from the standard TRIZ matrix [5, 21]. Another formalized method that aids in the development of bioinspired designs is function-based bioinspired design. Functional representation is often done with engineering design in order to decompose the problem in a form-independent manner and clarify the problem. It was found that natural systems can also be functionally modeled in the same manner when using the Functional Basis terms, as developed by Stone et al. [22]. Searches can be conducted on a functional level to aid a designer in identifying relevant natural solutions. In order to determine which biological search terms to use, Nagel et al. developed an engineering-to-biology thesaurus. Terms from the engineering Functional Basis are paired with biological terms in order to bridge the knowledge gap between the two domains [23]. Cheong et al. used a natural language analysis algorithm to also identify biologically meaningful keyword corresponding to the Functional Basis [24, 25]. Several systematic search tools have been developed to help engineers retrieve relevant biological information such as Shu's natural-language based search [13], Chakrabarti et al.'s IDEA-INSPIRE software [11], DANE [26], and AskNature [27].

The study presented in this paper will examine the effectiveness of the directed method when an individual has the required biological knowledge base. The ability of biology students and engineering students to identify potential biological analogies to engineering design problems will be studied. This will help determine if it is enough to have a designer with a good biological knowledge base in the concept generation phase of bioinspired design, or if formalized bioinspired methods are necessary to retrieve biological information and form analogies.

Experimental Structure and Methods

To study the differences in ability to make analogies with natural systems while generating design concepts, participants were needed from both engineering and biology. For the between-subject experiment, undergraduate students were recruited from a capstone mechanical engineering design course and a senior-level biology neuroscience class. Thirty-six students (31 male students and 5 female students) majoring in mechanical engineering participated in the study after completing approximately one-third of the semester-long engineering design course. A total of 11 students (3 male and 8 female) majoring or minoring in the biological sciences participated.

Upon the start of the experiment, the participants were given a consent form and survey materials to determine major, background information, and self-efficacy. The students then received a packet containing a design problem, shown in Fig. 1, blank pages in which to sketch their solutions, and a survey to verify that the problem was new and unfamiliar to the participants. The participants were given

Problem Description:
Alarm clocks are essential for college students, however often times they will wake up a roommate and those around them as well. Design an alarm clock for individual use that will not disturb others. The clock should be portable for use in a variety of situations such as on the bus, in the library, or in a classroom. Your goal is to create as many solutions to the problem as possible. **Imagine how nature would solve this problem.**

Customer Needs:

- Must wake up individual with no disturbance to others.
- Must be portable and lightweight.
- Electrical outlets are not available as a constant power source.
- Low cost.

Please sketch and describe with words one design solution per page.

Fig. 1 Problem statement given to participants in the study

50 min to generate solutions to the problem and were asked to generate as many solutions as possible. Participants were encouraged to generate solutions of high quality and variety. They were also asked to sketch and note all solutions generated even if the solutions were not technically feasible. The students were issued a notification when there were 5 min remaining. Following the concept generation period, the biology students were asked to note the inspiration sources for their solutions in their packets.

The problem given to the students asked for designs of a portable personal alarm clock, which was previously used by Glier et al. [28]. This problem was chosen because it does not require significant technical knowledge to understand. Thus, the mechanical engineers would not have an obvious advantage over the biologists. The participants were asked to "imagine how nature would solve [the] problem". Glier et al. found that the results from directing novice engineering designers to consider nature did not have a significant difference from the results with no formalized methods [28]. The lack of a significant difference may be due to the engineering participants' not having a substantial biological knowledge base to search within for analogous biological solutions. This problem was adapted from one used to determine the impact of a technical education that includes formalized concept generation methods on the concepts generated [29]. Thus, this problem is well suited for the study here.

Hypotheses

It is hypothesized that the biology students will identify more analogous biological systems for the design problem. With the increased number of bioinspired designs, the novelty and variety of the solutions developed by the biologists should be greater. Since the mechanical engineering students have the advantage of taking a design course and having a greater technical design knowledge base, they are expected to have a greater quantity of ideas and a higher quality (feasibility) score.

Metrics for Evaluation

There were four main metrics to quantitatively compare the two different groups based on their ideation effectiveness: quantity of non-redundant ideas, quality, novelty, and variety. The metrics were based on those developed by Shah [30] and refined by Linsey [31]. All four metrics' analysis begin on the solution level. A solution is a full concept or design developed by the participant. Most participants developed multiple solutions. The students were instructed to "sketch one solution per page" so raters could easily differentiate between separate solutions.

Quantity

The solutions were first analyzed for the quantity of non-redundant ideas. Here, an idea is a component that satisfies a term from the Functional Basis [22], and each solution is often composed of multiple ideas. To determine the participants' quantity score, all solutions are examined individually and all of the ideas are listed for each solution. The ideas are then compiled for each participant and the redundant ideas are removed. The resulting count is the participant's quantity score. A second independent rater, who is also a mechanical engineering design doctoral student, analyzed the data as well. A comparison of the ratings found an inter-rater agreement of 0.97 (using Pearson's correlation), indicating high reliability.

Quality

The quality metric used for the experiment is based on the three-point scale developed by Linsey et al. [31]. A solution receives a score of 0 if it is technically infeasible or does not solve the fundamental problem of the design statement. A solution receives a score of 1 if the solution is technically feasible, but would be technically difficult to develop. A score of 2 is given to technically feasible solutions that would be easy to develop or where technology already exists. After rating all of the solutions, the scores are averaged for each participant. Once again, the second rater analyzed the data to check inter-rater reliability. Cohen's kappa was used since the quality metric is on a 3-point scale and a value of 0.23 was found, showing fair agreement [32].

Novelty

Novelty is a measure of the originality of the solution and a participant's novelty score indicates how unique the solutions are compared to those of other participants. For the metrics of novelty and variety, a bin sort is needed. Bins are categories of solutions with similar attributes or features. The bins used in this study were previously developed using a separate set of data by Glier et al. [28] for the same alarm clock design problem. There are a total of 39 different bins for the alarm clock problem, including items such as a vibrating device, headphones, or a device that shocks the user. After completing the bin sort, the novelty score of a particular bin is calculated using:

$$Novelty = 1 - \frac{Number\ of\ solutions\ in\ the\ bin}{Total\ number\ of\ solutions}$$

A very unique concept would have few solutions in the bin and a high novelty score. Each of the most novel solutions would be developed by only one participant, and the solution would be the only one in its bin. In order to calculate a participant's novelty score, the novelty scores of his or her solutions' bins are averaged. The inter-rater agreement is 0.75 (Pearson's correlation).

Variety

The final metric calculated is variety, which is a measure of the different types of solutions investigated by a participant. The variety score is calculated following the bin sort using the equation:

$$Variety = \frac{Number\ of\ bins\ the\ subject\ used}{Total\ number\ of\ bins}$$

If a participant developed solutions that fall into many different bins, he or she would have a high variety of solutions. The Pearson's correlation of the two raters' analysis of variety was 0.98, demonstrating high reliability.

Results and Discussion

The participants all generated solutions for a personal alarm clock for 50 min. The biology and engineering students' solution sets were analyzed using the four metrics described. The results of each metric are presented in this section, along with a qualitative study of the solutions. A comparison of the biological sources of

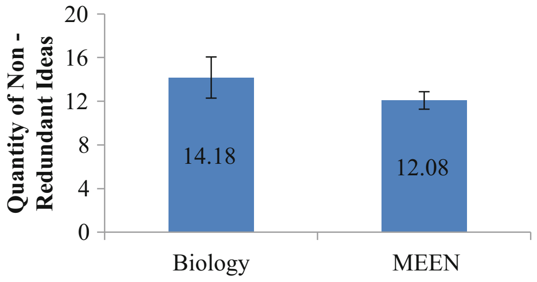


Fig. 2 The average quantity of non-redundant ideas generated by biology students and mechanical engineering students. Error bars show ±1 standard error

inspiration are also used to examine the effectiveness of a general biological knowledge base in finding natural analogues to engineering problems.

Quantitative Metric Comparison

As seen in Fig. 2, on average there was not a statistically significant difference in the quantity of non-redundant ideas produced by the biology students (M = 14.18, SD = 6.25) as compared to the engineering students (M = 12.08, SD = 4.81); t (45) = 1.18, *p* = 0.24. The quantity of non-redundant ideas, or functional components of a design, is related to the number of complete solutions that a participant generates, as long as the student is not simply repeating ideas. When examining the total number of solutions generated by the biology students versus the engineering students, there is a marginally significant result; t(45) = 1.81, *p* = 0.07. On average, the biology students are producing 2.30 more solutions than the engineering students (Fig. 3). However, the additional solution generated by the biology students must repeat many of the same ideas since there was not a significant difference in the quantity of non-redundant ideas.

There was also not a statistically significant difference in the quality of the solutions generated (Fig. 4); t(45) = 1.39, p = 0.17. The biology students had a mean quality score of 1.51 with a standard deviation of 0.37, while the engineering students had a mean quality score of 1.66 with a standard deviation of 0.27. Even when examining the number of high quality solutions (those receiving the maximum score of 2), there was not a statistically significant difference between the biology students (M = 5.27, SD = 4.38) and the engineering students (M = 3.72, SD = 2.25); t(45) = 1.57, p = 0.12 (Fig. 5). The lack of a significant difference in quality suggests that the biology students are able to develop feasible solutions that fit within the context of the posed problem as well as the engineering students can, which is contrary to the initial hypothesis.

The novelty scores for biology and mechanical engineering students are not statistically different with t(45) = 1.32, p = 0.19. On average, the biology students' solution sets have a mean novelty score of 0.91 with a 0.02 standard deviation while the mechanical engineering students' solutions sets have a mean novelty score of

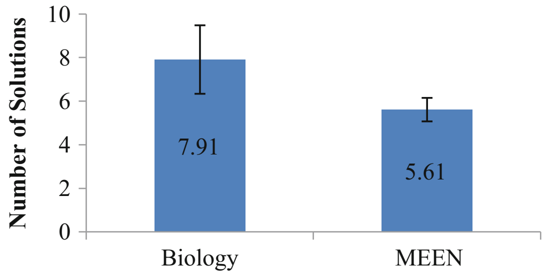


Fig. 3 A comparison of the average number of solutions generated by the biology and mechanical engineering students. Error bars show ±1 standard error

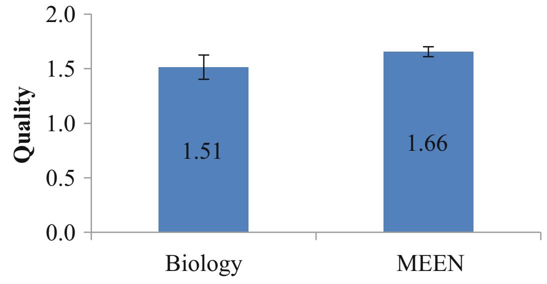


Fig. 4 A comparison of average quality scores between the biology and mechanical engineering students. Error bars show ±1 standard error

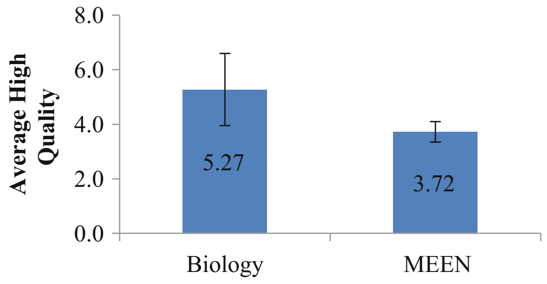


Fig. 5 A comparison of the average number of high quality solutions (solutions with a score of 2). Error bars show ±1 standard error

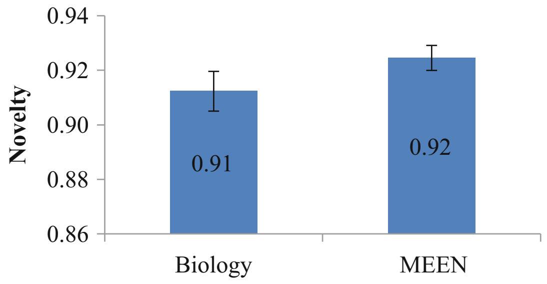


Fig. 6 A comparison of the average solution novelty for the biology students and the mechanical engineering students. Error bars show ±1 standard error

0.92 with a 0.03 standard deviation, seen in Fig. 6. However, of the eight bins that contain only one solution (very high novelty bins), seven of the solutions were developed by mechanical engineering students while only one of the highly novel solutions was developed by a biology student. This indicates that engineering

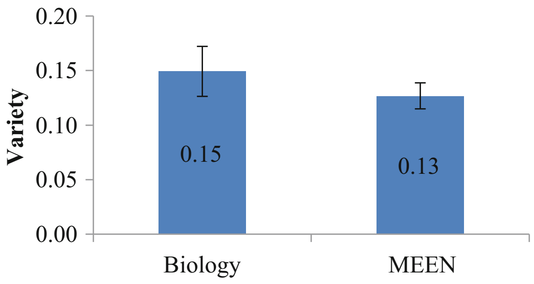


Fig. 7 A comparison of the variety of solutions produced by the students. Error bars show ±1 standard error

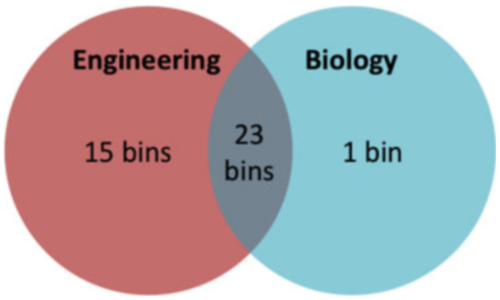


Fig. 8 Venn diagram presenting the number of bins used by only the engineering students, only the biology students, and by students of both groups

students may be able to generate more of the highly novel solutions, which conflicts with the hypothesized result that biology students would use more distant domain analogies from nature, leading to more novel solutions.

For the final metric of variety, the p-value from the t-test (t(45) = 0.90) comparing the two groups is 0.37, which indicates that there is no statistical significance. The biology students' mean variety score was 0.15 (SD = 0.08) and the engineering students' mean variety score was 0.13 (SD = 0.07), seen in Fig. 7. Individually, a biology student and an engineering student should both generate a similar variety of solutions, on average. However, when comparing the number of bins used only by the engineering students to those used only by the biology students (Fig. 8), it is evident that as a group, the engineering students generated a greater variety of solutions. This is opposed to the hypothesis that the biology students would develop a greater variety of concepts because they have greater biological knowledge from which to draw inspiration. The engineering students were able to develop solutions that fell within 15 bins that the biology students did not find, while only 1 biology student developed a unique solution that none of the engineers generated. This suggests that engineering students are developing different solutions from each other since the average engineering student's variety score is not high.

Based on the results of this study, none of the quantitative metrics resulted in a statistically significant result. This signifies that the mechanical engineering and biology students have relatively equivalent design competence for the given design problem. The mechanical engineering students were expected to generate a greater quantity and quality of ideas since they had completed 3 years of mechanical engineering coursework and were taking their first engineering design course, while the biology students were not formally trained in engineering or design. The similar metric scores between the two groups could be due to the alarm clock design prompt, which does not require excessive technical expertise so that taking engineering courses does not offer an obvious advantage. Another likely cause of the apparent equivalent design competence is that a majority of the engineering students did not employ the formal design methods they learned in class, but rather resorted to using general unstructured ideation.

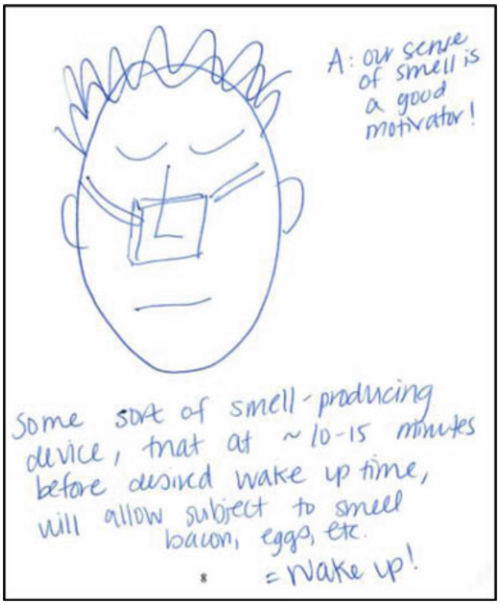
While the quantitative results were not statistically significant, the generated solutions still lead to interesting trends. Of interest here is the bioinspired nature of the solutions posed by the two subject groups. The specific and established metrics used above do not account for the source or type of inspiration used to generate the concepts.

Common Sources of Inspiration

There were four categories of solutions that were generated most often and over 50 % of the solutions fell into these categories. The most popular category for both the biology students and the mechanical engineering students related to wearable vibrating devices such as a watch, bracelet, or necklace, or a standalone vibrating device such as a pillow or phone. The other three common categories of solutions were wearing headphones or earbuds to contain the sound; getting a small shock as an alert; or wearing something that would light up when the specific time was reached.

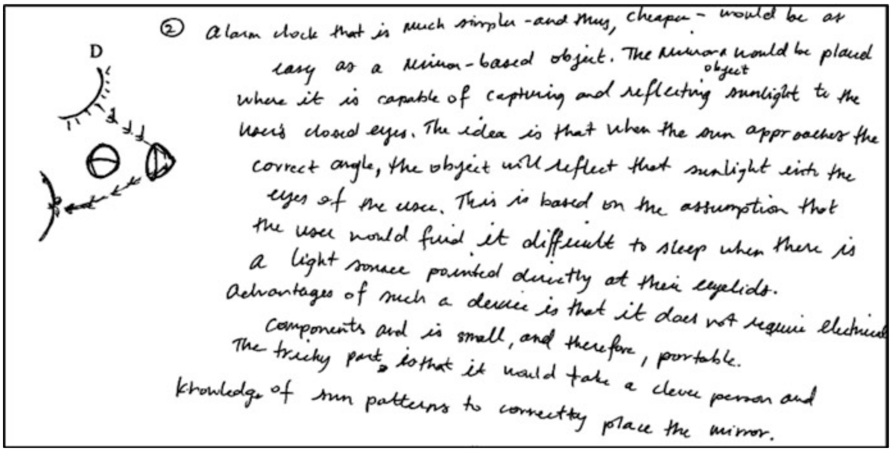
Using the notes students made, the sources of inspiration were reviewed. Many of the students specifically noted that they were looking to their own experiences when generating solutions. According to the subject self-identification, the vibrating device was inspired by students' cell phones set on vibrate as an alert. The headphones were mentioned as based on a phone or MP3 player since the devices are easily portable and allow an individual to listen to music privately. Solutions in the bin for "wearable lights" frequently cited the rising sun as inspiration since the light commonly wakes individuals in the morning. Other experiences noted by the students included being touched or hit, having water splashed in their face, experiencing a temperature change, or smelling food or smoke. The participants then generated solutions inspired by these approaches. For example, in Fig. 9, the participant noted that smelling breakfast is a good motivator to wake up so she mimicked that experience by adding a device to release the same scent at a predetermined time.

The problem statement used in the experiment asked to "imagine how nature would solve [the] problem". Using this guidance, several students explored the realm of "how does one naturally wake up" or how nature would wake a person.


A: our sense of smell is a good motivator!

Some sort of smell-producing device, that at ~10-15 minutes before desired wake up time, will allow subject to smell bacon, eggs, etc. = wake up!

Fig. 9 A solution generated mimicking an actual experience



② Alarm clock that is much simpler -and thus, cheaper- would be as easy as a mirror-based object. The mirror object would be placed where it is capable of capturing and reflecting sunlight to the user's closed eyes. The idea is that when the sun approaches the correct angle, the object will reflect that sunlight into the eyes of the user. This is based on the assumption that the user would find it difficult to sleep when there is a light source pointed directly at their eyelids.
Advantages of such a device is that it does not require electrical components and is small, and therefore, portable.
The tricky part, is that it would take a clever person and knowledge of sun patterns to correctly place the mirror.

Fig. 10 A concept that directly copies how nature would wake a person up

Surprisingly, all of the students that explicitly noted examining this line of thought were mechanical engineers. Natural methods of waking noted were the sun, insect stings, allergies, bird sounds, and hunger. However, not all of these ideas could be developed into a solution. While the students thought a great deal about how they are awoken or how nature affected them, they often copied these natural methods directly rather than trying to use the natural world as a source of inspiration. In Fig. 10, the participant explained that light makes it difficult for someone to sleep.

But rather than developing a device using a change in light as inspiration, he chose to design a mirror-based object that would just reflect the sunlight onto a person's eyes.

Biology and Engineering Solution Comparison

Based on observation by the experimenter, the attitude towards completing the experimental activity differed between the two subject groups. The biology students participating in the experiment appeared more nervous about the activity than the mechanical engineering students. After completing the study, about half of the biology students apologized for their work and stated that they were not sure if they had completed the activity correctly. Nevertheless, the solutions developed by the biology students did not differ greatly from the mechanical engineering students' solutions. Bins that contained at least 5 solutions were considered "popular", meaning they had low novelty, and 14 of the 39 bins fell into this group. While it was expected that the biology students' lack of technical design knowledge would hinder them in developing solutions, the biology students actually generated solutions that fell into all of the popular bins with only eleven participants. As Fig. 8 shows, the biology students developed solutions that fell into 24 different bins. On the other hand, the engineering students developed solutions for 38 of the 39 bins, with 15 of these bins unique to just the engineering students. This indicates that the biology students were confined within a smaller solution space.

The biology students did want to put their knowledge into use and developed solutions such as those based on REM cycles, sleeping pills with GABA inhibitors, changes in level of melatonin, and enhancements in the Circadian rhythm. Interestingly, the biology students were making direct changes to a person rather than creating some separate device that woke the sleeper. Some superficial similarity was found between solutions generated by both groups of test subjects. For example, the engineers suggested sleeping pills that would wake the person after a certain amount of time without knowing the detail about the usage of GABA inhibitors (see solutions generated by a biology and engineering student in Fig. 11). Another biology student suggested reducing levels of melatonin to help wake a person and suggested a release of caffeine as an aid, while mechanical engineering students simply suggested injecting people directly with caffeine.

While it was suspected that the greater proportion of female participants in the biology group as compared to the engineering group might affect the results, a comparison of each gender's solutions found that there were minimal gender effects. Both genders used similar approaches to generating concepts, such as using their knowledge of biology to make direct biological changes to the product user, and considering the five senses that could be used to waken a person. Additionally, both male and female biology participants generated solutions that fell within the four most popular categories of solutions.

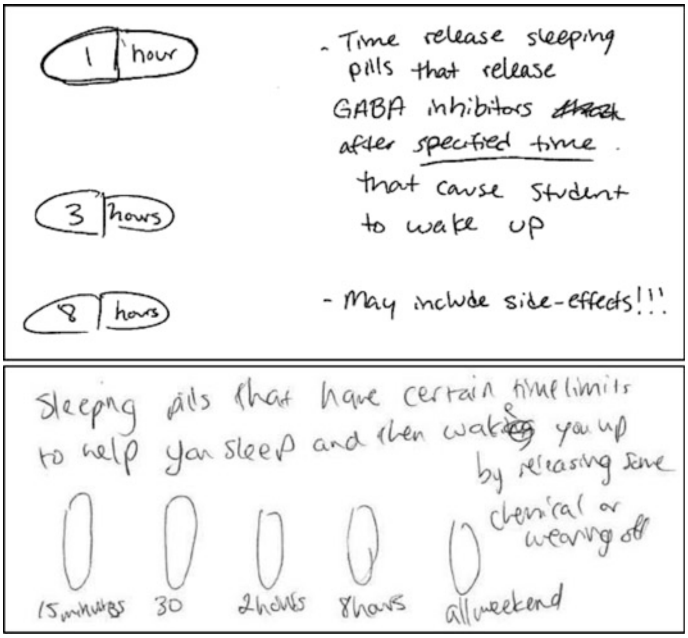
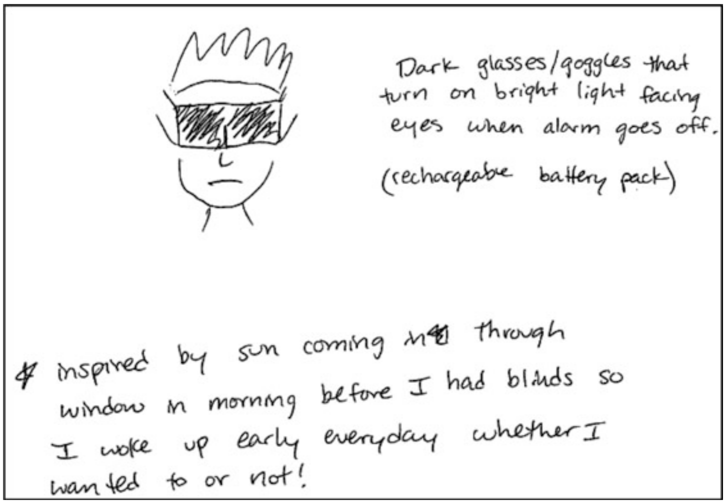

1 hour
3 hours
8 hours
- Time release sleeping pills that release GABA inhibitors ~~attack~~ after specified time that cause student to wake up
- May include side-effects!!!
Sleeping pils that have certain time limits to help you sleep and then wakes you up by releasing some chemical or wearing off
15 minutes
30
2 hours
8 hours
all weekend

Fig. 11 Solution generated by biology student making use of their biological knowledge (*above*). Similar solution developed by engineering student (*below*)

Identified Bioinspired Solutions

The main bioinspired solution common to both groups were glasses or eyemasks that were dark from the outside and lit up at a specified time, which is similar to how the sun rises in the morning. An example of one of these solutions produced by a participant can be seen in Fig. 12. Out of all 11 biology students, these solutions were the only ones in which students specifically noted a biological system as the source of inspiration. While the mechanical engineering students were not asked to note their sources of inspiration, many also referenced the sun in their solutions. One of the mechanical engineers did develop another bioinspired solution using personalized frequencies, which was based on how animals have frequencies unique to their own species. However, there were very few other obvious bioinspired solutions.

Overall, the biology undergraduate students were not able to produce any unique bioinspired solutions, although the students were able to produce a good variety of solutions in high quantity. However, almost all of the solutions generated by the biology participants were also found by the engineering students. For more technical problems, the engineering students would most likely prove more useful in generating solutions because of their background design knowledge, which would aid in understanding the problem and generating ideas.



Dark glasses/goggles that turn on bright light facing eyes when alarm goes off. (rechargeable battery pack)

* inspired by sun coming in through window in morning before I had blinds so I woke up early everyday whether I wanted to or not!

Fig. 12 Bioinspired solution based on the sun

Conclusion

This paper investigated the differences in the solutions generated by undergraduate biology students and mechanical engineering students. The students were directed to consider how nature might solve a design problem in order to influence the students into thinking about the natural world and developing bioinspired solutions. There was a sample size of 11 for the biology students along with the 36 mechanical engineering students that participated. There was no statistically significant difference in the average solution sets from the two groups for the metrics of quality and novelty. The biology students did generate a greater number of solutions, but they repeated functional components, which resulted in similar quantities of non-redundant ideas. The mechanical engineering students were able to generate a greater variety of solutions as a whole compared to the collective biology students. However, on average, the participants from each group have similar variety scores.

The participants were asked to design a portable alarm clock that would not disturb others and the students looked to their own experiences when generating solutions. Most tried to directly copy different techniques that had woken them in the past rather than analogically mapping from other domains. The only bioinspired solution noted by the biology participants was designing a device that would light up at the specified time, similar to how a sun rises to mark the start of a new day. The biology students almost exclusively looked at "within organism" domains by thinking about how humans wake up, rather than exploring "other organism" analogies, as was also found by Dunbar [14]. The mechanical engineering students found the same sun-inspired solutions and only one of the students noted an additional biological analogy. While, only one additional biological analogy was developed, the engineering students did attempt to venture to more distant "other organism" domains since several noted their attempts to consider how nature or animals wake up. Similar to the results of Christensen and Schunn's study, the engineers were much more likely to look for distant domain analogies than the biologists [15].

The biology students did try to use what they had learned in their classes to develop solutions. However, the concepts they generated were still conventional enough that engineering students were able to find similar solutions. This indicates that the biology students are searching the same area of their memory as the engineering students, but they are applying some of their biological knowledge within their design. As Shu suggested, the biology students were biased towards a certain area of biology [13]. The biology students were recruited from a senior-level undergraduate neuroscience course and many of their biological solutions related to neuroscience material, such as utilizing REM cycles or creating sleeping pills with GABA inhibitors. It would be expected that engineers learning more biology would also be biased towards their specific area of study when developing bioinspired designs.

Since the students with the biological knowledge base were not able to develop any unique bioinspired solutions, having basic biological knowledge is likely not enough for bioinspired design. Designers need help in identifying analogous biological systems and transferring the systems into the engineering domain. One issue is that superficial similarity is often used to find an analogy, which causes a transfer between two distant domains to be more difficult due to a variety of small differences between the domains [33]. Further study may also need to be conducted to more deeply understand the cognitive bases of design by analogy, which can help to determine other issues that plague designers during the analogical reasoning process. Additional designer studies should also be conducted on the formal bioinspired methods and search tools that are currently in development. These products are superior to the directed method because they do not require engineers to possess a significant biological knowledge base. Another advantage is that the formal methods and search tools can assist with analogical retrieval, which is known to be a major hurdle in the analogical reasoning process. Some of the methods and tools also help with the mapping and abstraction analogical processes. It is possible that designers can utilize these products to overcome various cognitive issues of the design by analogy process. As such, the continued development of these methods and tools can be beneficial in aiding designers' identification of potential natural systems to use for analogies, which also allows the designers to produce bioinspired designs without extensive biological knowledge.

Acknowledgment The authors would like to acknowledge the support provided by the National Science Foundation NSF EEC 1025155/1338383. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Van Nierop EA, Alben S, Brenner MP (2008) How bumps on whale flippers delay stall: an aerodynamic model. Phys Rev Lett 100(5):54502, 1-4
2. Barthlott W, Neinhuis C (1997) Purity of the sacred lotus, or escape from contamination in biological surfaces. Planta 202:1–8
3. Geim A, Grigorieva SVDIV, Novoselov K, Zhukov A, Shapoval SY (2003) Microfabricated adhesive mimicking gecko foot-hair. Nat Mater 2:461–463
4. Autumn K, Sitti M, Liang YA, Peattie AM, Hansen WR, Sponberg S, Kenny TW, Fearing R, Israelachvili JN, Full RJ (2002) Evidence for van der Waals adhesion in gecko setae. Proc Natl Acad Sci 99:12252–12256
5. Vincent JFV, Mann DL (2002) Systematic technology transfer from biology to engineering. Philos Trans R Soc Lond Ser A Math Phys Eng Sci 360:159–173
6. Yen J, Weissburg M (2007) Perspectives on biologically inspired design: introduction to the collected contributions. Bioinspir Biomim 2:1–4
7. Dym C, Agogino A, Eris O, Frey D, Leifer L (2005) Engineering design thinking, teaching, and learning. J Eng Educ 94:103–120
8. Hyman BI (1998) Fundamentals of engineering design. Prentice Hall, Upper Saddle River
9. Otto KN, Wood KL (2000) Product design. Prentice Hall, Englewood Cliffs
10. Ulrich KT, Eppinger SD, Goyal A (2011) Product design and development. Irwin/McGraw-Hill, New York
11. Chakrabarti A, Sarkar P, Leelavathamma B, Nataraju B (2005) A functional representation for aiding biomimetic and artificial inspiration of new ideas. Artif Intell Eng Des Anal Manuf 19:113–132
12. Benami O, Jin Y (2002) Creative stimulation in conceptual design. IDETC 2002:251–263
13. Shu L, Ueda K, Chiu I, Cheong H (2011) Biologically inspired design. CIRP Ann Manuf Technol 60:673–693
14. Dunbar K (1997) How scientists think: on-line creativity and conceptual change in science. In: Ward TB (ed) Creative thought: an investigation of conceptual structures and processes. American Psychological Association, pp 461–493
15. Christensen BT, Schunn CD (2007) The relationship of analogical distance to analogical function and preinventive structure: the case of engineering design. Mem Cognit 35:29–38
16. Vattam S, Helms ME, Goel AK (2007) Biologically-inspired innovation in engineering design: a cognitive study. Technical report. Georgia Institute of Technology
17. Gebeshuber I, Drack M (2008) An attempt to reveal synergies between biology and mechanical engineering. Proc IME C J Mech Eng Sci 222:1281–1287
18. The Biomimicry Institute (2013) What is biomimicry? http://biomimicryinstitute.org/about-us/what-is-biomimicry.html
19. Biomimicry Guild (2008) A conversation with Janine Benyus
20. Altshuller G (1999) The innovation algorithm: Triz, systematic innovation and technical creativity. Technical Innovation Center, Worcester
21. Vincent JFV, Bogatyreva OA, Bogatyrev NR, Bowyer A, Pahl AK (2006) Biomimetics: its practice and theory. J R Soc Interface 3:471–482
22. Stone RB, Wood KL (2000) Development of a functional basis for design. J Mech Des 122:359–370
23. Nagel JKS, Stone RB, Mcadams DA (2010) An engineering-to-biology thesaurus for engineering design. IDETC 2010:117–128
24. Cheong H, Chiu I, Shu L, Stone R, Mcadams DA (2011) Biologically meaningful keywords for functional terms of the functional basis. J Mech Des 133:021007-1-11
25. Cheong H, Shu L, Stone RB, Mcadams DA (2008) Translating terms of the functional basis into biologically meaningful keywords. IDETC 2008:137–148
26. Vattam S, Wiltgen B, Helms M, Goel A, Yen J (2011) Dane: fostering creativity in and through biologically inspired design. Des Creat 2010:115–122

27. The Biomimicry 3.8 Institute (2008–2013) Asknature. http://www.asknature.org/
28. Glier MW, Tsenn J, Linsey JS, Mcadams DA (2012) Evaluating the directed method for bioinspired design. IDETC 2012:403–413
29. Genco N, Hölttä-Otto K, Seepersad C (2012) An experimental investigation of the innovation capabilities of undergraduate engineering students. J Eng Educ 101:60–81
30. Shah JJ, Smith SM, Vargas-Hernandez N (2003) Metrics for measuring ideation effectiveness. Des Stud 24:111–134
31. Linsey J, Tseng I, Fu K, Cagan J, Wood K, Schunn C (2010) A study of design fixation, its mitigation and perception in engineering design faculty. J Mech Des 132:041003, 1-12
32. Landis JR, Koch GG (1977) The measurement of observer agreement for categorical data. Biometrics 33:159–174
33. Johnson-Laird PN (1989) Analogy and the exercise of creativity. In: Vosniadou S (ed) Similarity and analogical reasoning. Cambridge University Press, pp 313–331

A Structural Equation Modeling Approach to Product Innovation

Christopher Hoyle, Irem Tumer, and Brady Gilchrist

Abstract This paper aims to take a new approach to quantifying innovation using a structural equation model (SEM). The SEM consists of a measurement model using survey data to model the perception of innovativeness for a given population, a structural model using product attribute data to quantify product innovativeness in terms of product attributes, and a binary choice model to relate product innovativeness and other latent characteristics to its selection as an innovative product by the target population. The SEM is estimated using Hierarchical Bayes methods. This model approach is capable of differentiating the innovativeness among a set of products and identifying products perceived as most innovative. A case study involving sets of innovative and common consumer products is provided to illustrate the SEM approach. In the SEM, three latent variables are identified, product innovativeness, product usability, and company profile, which form the basis for characterizing a product as innovative. The results of the model and its utility in the design process are discussed.

Introduction

Innovation is relatively easy to sense in a product, but has proven to be difficult to quantify for the design process. Experts can look at a product or process and determine whether or not it is innovative and why [1]; however, this approach is highly subjective and does not provide design guidance. To date, there is no universally accepted method to quantify the "innovativeness" of a given product or rank products with respect to innovativeness. The research presented herein attempts to (1) demonstrate that the "innovativeness" of a product can be quantified using a structural equation model (SEM); and, (2) show that this metric for innovativeness, based on customer perceptions, can be related to physical product parameters to estimate the innovativeness of a given product design based upon its product attributes. SEMs utilize latent variables: variables that are abstract and

C. Hoyle (✉) • I. Tumer • B. Gilchrist
Oregon State University, Corvallis, OR, USA
e-mail: chris.hoyle@oregonstate.edu

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1_37

difficult to quantify directly but are related to measurable indicators [2] which are used to estimate them.

It is necessary for the purposes of this study to define innovation. To best align with the goal of this study, a definition of innovation is taken from [3] as "providing novel value to the customer that produces growth and profit." Innovation has typically been discussed as a means to gauge technological progress; however, this paper measures innovation in the context of product innovativeness for use as a method to select preferred design concepts. The literature supports the use of models of innovation on the national level and the corporate level, but has yet to delve into measuring the innovativeness of products [3–7]. Innovation measurements at the firm level are used for comparisons between different companies, using different models and indicators, such as patent count and R&D expenditures. Measuring innovation at the national level involves comparing innovative output between countries. These metrics are used to measure competitiveness between nations, or to gauge technological progress [7]. In the marketing domain, work exists in quantifying the impact of innovation, but not specifically in quantifying innovativeness of a product [8–10].

In the product domain, methods exist to judge the creativity of a set of designs during the concept generation phase. One method, introduced by Shah et al. measures four different aspects of a concept. In this method, quality, quantity, novelty, and variety are calculated for each concept, based on the different ways each product-related function is defined, and used to select a concept to pursue in detailed design [11]. Oman et al. extended this approach and isolated the metrics most closely associated with creativity (novelty and variety) and developed the comparative creativity assessment and multipoint creativity assessment method [12].

The research in this paper contributes a quantitative measure of the innovativeness of products. This measure can then be used to screen concepts, or select components and help designers to better understand the perceptions of the products they develop. The approach presented first uses a psychometric survey to gather information about the perception of the characteristics and traits of four pairs of products. The pairs are presented using one product identified as innovative and an associated product not identified as innovative by an outside source (e.g. magazine, website). Next, the SEM is used to create a model which links the product design attributes controlled by a product designer to the choice of a given product as innovative. Potential applications of its use are then presented with their benefits to conceptual design. Finally, conclusions that can be drawn from the results are offered along with future work.

Background

To design innovative new products, it is necessary to estimate how innovative these products will be viewed by potential users. If innovativeness can be assessed earlier in the design process, designers can better focus their efforts to create more innovative products. Current methods of identifying and characterizing whether a product is innovative are based on expert opinions. Qualitative, i.e. less subjective, methods for calculating the innovativeness of products do not exist. It is hypothesized that innovativeness is a latent, or unobserved, variable which cannot be directly quantified but can be estimated through the use of measured indicators.

This section first introduces currently used methods of quantifying innovativeness at both the national and firm levels. Next, methods of uncovering latent variables are described with their applications in behavioral and social sciences, as well as engineering and product development.

Methods for Quantifying and Characterizing Innovation

Metrics for quantifying innovation vary widely. Some are single measures, such as R&D expenditures or patent propensity. Others are a combination of indicators, which give a more complete description of innovation activity. These measures have their drawbacks though, and none of the measures of innovation are formulated at the *product level*: they are used to track trends in innovation activity.

Indicators used to measure innovation are typically based on input to the innovation process and output measures of success and growth [5]. One measure of firm innovation is patent propensity, or the number of patentable inventions that are patented [4]. However, this metric has drawbacks that limit its accuracy in that not all products or innovations are patented, so the result is contingent upon whether the company has patents for their products. Some firms use secrecy to protect their ideas. Arundel and Kabla surveyed over 600 companies in Europe to determine the rate of patented technology and found on average only 35 % of innovations are patented [4]. In addition, the ratio of patents to R&D has declined in the past 40 years, showing a decrease in the use of patents to protect innovations [13]. Patent propensity is just one aspect of a larger model of innovation at the firm level.

R&D input, or investment, is used as a measure of innovation output in Mairesse and Mohnen's work [14]. Data was collected in France based on new products introduced in the market. Indicators of innovation include quantities such as R&D per employee, share of sales by new products, and share of sales of products protected by patents. It was also noted in this work that the indicators were highly subjective, further strengthening the need for a less subjective measure. Their work aimed to determine the effect of R&D on various innovative indicators, and resulted in a positive correlation for all indicators measured. As is the case with patent propensity, R&D expenses do not give a complete view of innovation. In addition, it is simply another measure of firm innovation, not product innovation.

Shapiro proposed a new measure of innovativeness based on the percentage of revenue from "new platforms" in [3]. A platform is essentially a base design from which product variants are developed. This is combined with a measure of revenue from new products to give more insight into the successful development of innovative new products.

As with the previously introduced measures of innovation, the following are metrics of either national innovation or firm innovation that combine multiple indicators to show areas of strength or weakness. The European Union publishes the Innovation Union Scoreboard (IUS) yearly. This report outlines the innovative performance of European Union nations based on several indicators, including: percentage of population completing tertiary education, number of international scientific publications, venture capital investments, and R&D expenditures [7]. Measuring innovation at the national level, Mairesse and Mohnen created a framework using metrics such as innovation intensity, propensity of innovation, and size of the firm [6]. Another example of measuring firm innovativeness is the Composite Innovation Index developed by Carayannis and Provance. The index is based on three major factors: innovation posture, propensity and performance [5].

Finally, looking specifically at innovative products, Saunders and Seepersad surveyed 197 products identified by outside sources as innovative. By comparing to a competing product and isolating the characteristics that made those products innovative, they were able to compile a list of five major categories of innovation: functionality, architecture, external interactions, user interactions, and cost [15]; however, they did not create a model of innovativeness for a given product.

Structural Equation Models (SEMs)

SEMs are used to create models in which there is a combination of observable, and hence, *measured* variables combined with indirectly observed "hidden", or *latent*, variables. The idea behind latent variables is that there are observable and quantifiable factors that can be used to estimate the unobserved latent variables using a *measurement model*. A latent variable measurement model is based on the idea that the number of driving forces on a given variable is less than the number of available measurements [16].

Behavioral and social scientists have used SEMs and general latent variable models to describe unobservable quantities of interest. Qualities such as preferences, attitudes, ambition, intelligence, personality traits, and behavioral intentions have been quantifiable using latent variable models [17, 18]. More recently, the technique has been applied to other areas, for example, to model the occurrence of diseases such as alcoholism [19]. Varying degrees of alcohol dependence can be measured using *indicators* such as amount, frequency, frequency of high consumption, and selected demographic information. Alcoholism, like other latent variables, is not something that is directly observable, but it can be estimated using other observable indicator variables [19].

Recent research exists in applying the latent variable concept to industrial processes [16, 20, 21]. Kim et al. used the concept to model the forging process and to identify defects and problems without necessitating visual inspection, saving time and money [22]. Robustness of a membrane manufacturing process has also been modeled as a latent variable problem in [23].

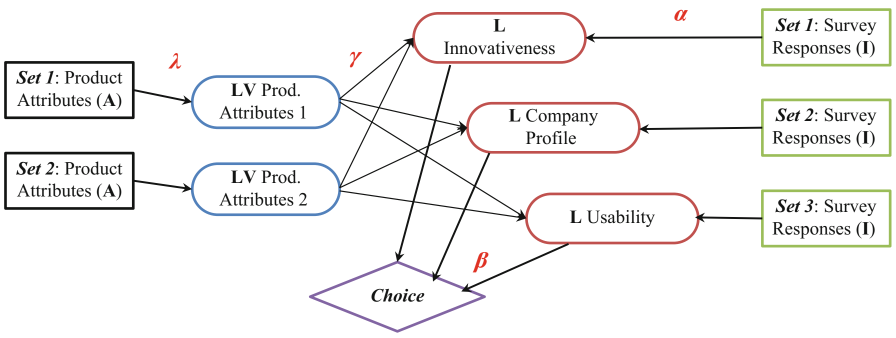
Applying the latent variable approach to customer preference, Wassenaar et al. created an integrated model of passenger vehicles [24]. It was proposed that consumers do not necessarily purchase cars based on measureable engineering attributes such as transmission design, number of seats, engine power, or other engineering attributes, but that they are mainly concerned with perceived attributes such as safety, comfort, and drivability. From this model, the authors were able to gain a better understanding of how customer desires relate to vehicle choices.

Little research exists linking the concept of "innovation" to latent variables. Innovation is a good example of a latent variable: it cannot be directly measured, yet everyone has some sense of what it is. Leder and Carbon considered "innovativeness" to be an aspect of vehicle interior design. Showing customers a series of vehicle interiors with different attributes, such as the shape of the steering wheel, they were able to create a model of customer perceptions. They also used a questionnaire to uncover the customer's perception of innovativeness [25]. Trigo and Vence measured innovative cooperation between companies in the service sector using latent class analysis, a type of discrete latent variable analysis, to measure how companies cooperate to innovate new ideas [26]. Innovative attitude was used as an indicator in a latent variable model for "level of technology" as it was applied to agriculture and the adoption of new technology by Esposti and Pierani [27]. The "innovativeness" of IT firms has also been measured using latent variables. Lopes created an Innovation Index based on the results of the model created in [28].

While innovativeness has been previously characterized and measured on the national and firm level, it has yet to be quantified at the product level. The goal in this work is to use a SEM to determine what products people *perceive* as innovative and why. It is hypothesized that the SEM will uncover attributes of products that lead people to think of them as innovative. Innovation, being inherently abstract and not directly discernible fits the criteria for applying a latent variable approach, such as the SEM. The method of applying the concept of a SEM model to measuring the innovativeness of individual products is presented next.

Structural Equation Modeling Methodology for Innovativeness

The overall SEM for innovativeness is shown in Fig. 1. The model consists of three sub-models consisting of the (1) measurement model, (2) structural model, and (3) binary choice model. The *measurement model* relates the indicators $\mathbf{I}$ from survey data to the unobserved latent variables $\mathbf{L}$. To measure the indicators, a psychometric survey is administered to a set of respondents to determine their perceptions of the characteristics of innovative and common products. The measurement model is estimated based on the indicator values derived from the psychometric survey: a set of coefficients $\boldsymbol{\alpha}$ is estimated which relates the latent



Set 1: Product Attributes (A)
Set 2: Product Attributes (A)
λ
LV Prod. Attributes 1
LV Prod. Attributes 2
γ
L Innovativeness
L Company Profile
L Usability
α
Set 1: Survey Responses (I)
Set 2: Survey Responses (I)
Set 3: Survey Responses (I)
β
Choice

Fig. 1 Innovative Dyson Air Multiplier™ (*left*) with its common product, Holmes® Fan (*right*)

variables **L** to the indicators **I**. The *structural model* relates the measurable product attributes to the latent variables, estimated using the measurement model. This model shows which product attributes are correlated with the latent variables. The structural model relates the product attributes **A** to the latent variables **L** through sets of coefficients $\boldsymbol{\gamma}$ and $\boldsymbol{\lambda}$. The structural model is a actually a two-stage model in which product attributes **A** are first linked to intermediate latent variables **LV**, representing latent product characteristics, through model parameters $\boldsymbol{\lambda}$, and then the latent product characteristics are related to the user-perceived latent variables **L** through model parameters $\boldsymbol{\gamma}$. The binary *choice model* is derived from both the measurement model and the structural model. This model relates the latent variables **L**, and ultimately the product attributes **A** to choices made by respondents regarding which product is innovative, through model parameters $\boldsymbol{\beta}$. A binary logit choice model is used for this model.

The number of responses needed to estimate the three models shown in Fig. 1 can be determined from a number of empirical relationships. To estimate a choice model, a minimum of 150 responses is generally recommended [29]. To estimate the latent variables from the indicators, a good practice is to have at least four indicators per latent variable [30]. Additionally, a general rule for identification of latent variable models is that the number of unknown parameters should be less than the number of variances and covariances in the observed covariance matrix of the data [2]. In this work we have 20 observed variables, so we can have no more than 210 unknown model parameters.

Estimating a SEM with several sub-models is computationally challenging and it can be difficult to achieve model convergence. Maximum likelihood Estimation (MLE) or Hierarchical Bayes (HB) methods can be used to estimate these models [31]. In this work, the model is estimated using the Hierarchical Bayes approach. The model is solved using Gibbs sampling implemented in WinBugs [32]. The advantage of HB is that the more difficult problem of finding an optimal solution using MLE is replaced with the problem of finding the posterior distribution of the model coefficients, which can be achieved using numerical methods such as Gibbs

Table 1 Products used for latent variable survey

Innovative product	Common product
Dyson Air Multiplier™ [28, 29]	Holmes® Fan
Powermat® [33]	Journey's edge dual powered charging station
Oliso® Smart Iron [34]	Proctor Silex® Iron
KidSmart Vocal Smoke Detector [35]	First alert basic smoke alarm

sampling in the HB method. This makes model convergence easier for complex models.

A detailed description of the three models used to estimate the Innovativeness SEM is described as follows.

Measurement Model

Survey data is required to estimate the measurement model: survey questions are indicators **I** of each latent variable **L**. The use of a psychometric survey data provides a representation of the customers' perceptions of the products used in the study. The survey used in this paper consisted of 20 questions that pertained to the products listed in Table 1 and range from questions about the specific product functionality to the company's influence around the world. The set of indicators was determined through consultation with both researchers in the area of survey design and in the area of product innovation. Because the goal is to have at least four indicators for each latent variable, it was theorized that 20 question would allow discovery of three to four latent variables (assuming some of the indicators would not prove to be significant). Each question has a scale between one and seven with one indicating very low agreement and seven being high agreement with the question. Each participant was asked to respond to one set of survey questions for an *innovative* product, as well as its associated *common* product.

Sources such as Popular Science's "Best of What's New," IDSA's "IDEA" awards, and TIME Magazine's "Best Inventions of..." were used to identify the innovative products used in the study. The common products were chosen based on the absence of claims that the product was innovative in their product descriptions. The surveys responses were collected in pairs of one innovative product and one common product. An example of a product pair, the Dyson Air Multiplier™ and the Holmes® Fan, can be seen in Fig. 2. The two products have the same black box functionality, i.e. to move air, but use different means to achieve it. The Dyson has an impeller in the base which channels air around the multiplier ring where it uses the process of inducement to draw air into the flow and accelerate it forward [36].

A list of the innovative features of the innovative products used in the study can be seen in Table 2.

Some of the indicators (i.e., survey questions) used were based on modified selection criteria from the innovative products lists use for this study. One of the

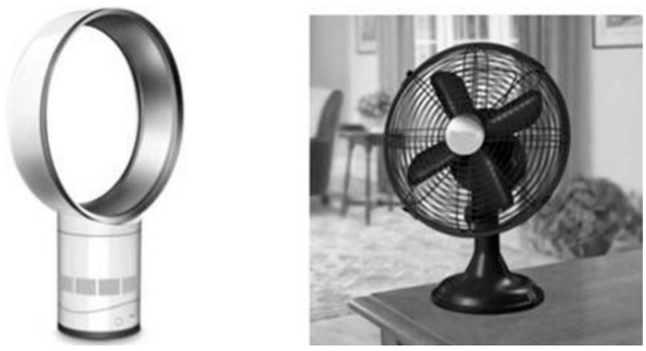


Fig. 2 Innovativeness structural equation model (SEM)

Table 2 Innovative features of the innovative products

Product	Innovative feature
Dyson Air Multiplier™s	Bladeless fan. Air pulled in through the base and pushed out using an impeller around a circular airfoil [28, 29]
Powermat®	Magnets align the device with the pad and using magnetic induction, the pad charges the device [33]
Oliso® Smart Iron	When the handle is released, the iron automatically lifts itself an inch above the ironing board [34]
KidSmart Vocal Smoke Detector	Uses a parents recorded voice to wake children and provide evacuation instructions [35]

sources, Popular Science, uses several criteria to judge innovative products, including significance of design, quality of design, and originality [37]. Selection criteria from the sources used for the innovative products in the study have been adapted and expanded to form the product innovativeness survey.

To find the indicator values for each of the products, a psychometric survey was used. They survey was administered during the 2012–2013 school year in junior level mechanical design courses at Oregon State University. The subject population was targeted because of their knowledge of engineering principles and product design, as well as the convenience of using local students; however, this population group would generally not be representative of a product consumer group and therefore this study is treated as a pilot study. The physical products listed above were available for the participants to manipulate and use during the survey to better understand their functionality. See Appendix for an example of the survey.

A total of 133 surveys were collected in pairs of one innovative and one common product for a total of 266 responses. Using this data, exploratory factor analysis was implemented on the responses. This follows the approach outlined by Loehlin in [30] to determine the number of latent variables represented by the data collected. This is done to describe the covariance relationships among all of the random variables in terms of a few underlying variables. To determine which factors are significant, the Kaiser-Guttman rule was applied to the factor correlation matrix,

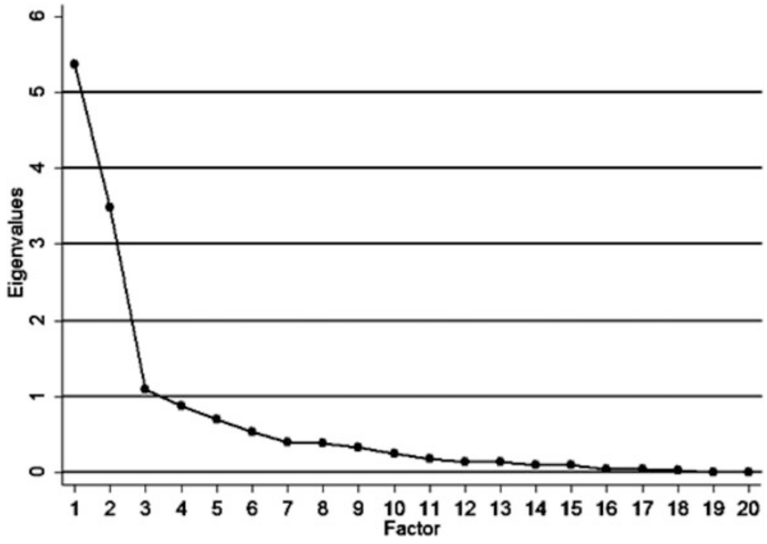


Fig. 3 Scree plot of all potential factors

where an eigenvalue of greater than one indicates significance [30]. As seen in the Scree Plot in Fig. 3, the first three factors have eigenvalues above one, indicating significance. Using the scree test to confirm factor significance, where the curve of decreasing eigenvalues changes from rapidly decreasing to a flat gradual slope, it was determined that the first three factors are significant [30].

These 20 survey questions are reduced to three significant factors. Once these factors were identified, the factor pattern matrix was rotated using the Varimax rotation developed by Kaiser [38]. Varimax performs an orthogonal rotation of the factor matrix by an arbitrary factor such that a principal factor matrix results. Finally, only factor loadings with absolute values greater than 0.35 (after Varimax rotation) are included in the measurement model. This is because a level of 0.35 corresponds to about 12 % of the variance in the indicator being explained by the factor; therefore, values of indicators less than 0.35 are not well explained by a given factor.

Based on the indicators that are significant for each of the factors, the three factors identified represent *innovativeness*, *usability*, and *company profile*. The positive indicator coefficients of product *innovativeness* include product complexity, originality, aesthetics, and perceptions of "high tech" and "trendiness." Interestingly, this factor had several negative loadings that would make practical sense. How successful the product was, global success of a product, influence of the product on society, as well as familiarity with its design and features all have a negative relationship with innovativeness. Intuitively, as a product is more prevalent across the world and the public becomes more familiar with its use, its perception of innovativeness decreases.

Table 3 Factor loadings for each indicator variable

Indicator	Innovativeness α_1	Usability α_2	Company profile α_3
Product success	1	1	
Product performance		1.023	
Global success	−0.962		
Global influence	−0.747		
Product efficiency		0.852	
Company reputation			1
Company influence			0.607
Product quality		0.438	
Product complexity	1.556		
Product originality	1.006		1.421
User benefits		0.775	
Product aesthetics	1.023		1.117
Product familiarity	−0.230		
Ease of use		0.577	
Satisfaction w features		2.963	
High tech	1.013		
Trendy	1.026		1.513
Product adjustability		2.832	

A second factor is determined to be product *usability,* based on the inclusion of the indicators of performance, efficiency, quality, benefits to the user, ease of use, satisfaction with the products' features, and adjustability of the product. These variables are determined to be most associated with how the product is used, and thus the factor is labeled usability.

The final significant factor uncovered is *company profile*. This factor relates to the reputation of the company that produced the product, the influence that company has on the market, how original the product is, and how trendy it is. It is assumed that this latent variable is a measure of brand equity, or the value of having a familiar company.

With the exploratory factor analysis completed to identify the **I** to **L** relationship, Eq. 1 is used for the measurement sub-model.

$$\boldsymbol{I} = \alpha \boldsymbol{L} \tag{1}$$

While the entire SEM was estimated "all at once" using the HB approach, the model coefficients $\boldsymbol{\alpha}$ for the measurement sub-model are shown in Table 3.

Structural Model

With latent variables identified, the structural model linking the measurable product attributes **A** to the latent variables **L** can be generated. First, a list of product attributes that are potentially related to the latent variables is compiled based on the literature. Data for each product is collected from the physical product itself and

Table 4 Factored product attribute coefficients

	(LV_1) Design λ_1	(LV_2) Performance λ_2
Power usage		1
Number patents	1	
Product materials		0.898
Product parts	0.902	0.503
Product features		1.015
Product sustainability	1.041	
Product functions	0.757	0.696
Product flows	0.977	
Product cost	1.015	

Table 5 Coefficients associated with each latent variable by product characteristic

	Design (LV_1) γ_1	Performance (LV_2) γ_2
Innovativeness	0.874	−0.065
Usability	−0.239	−0.115
Company profile	1.032	−0.119

product information available online. Attributes such as power usage, part count, product cost, number of different materials, and number of patents are used as the **A** in the structural model.

An exploratory regression of the structural sub-model demonstrated high collinearity between the model coefficients $\boldsymbol{\gamma}$. To diminish the issues caused by collinearity of the observed variables, an additional level of latent variable model is used to relate the product attribute **A** to intermediate product characteristic latent variables **LV** [39] through coefficients $\boldsymbol{\lambda}$. Following the same method used to generate the latent variables in the measurement model, factor analysis is performed on the product attributes. This results in two factored attributes, $\mathbf{LV_1}$ and $\mathbf{LV_2}$, as seen in Table 4. Using this method, a total of nine product attributes **A** are incorporated into the model.

Factor 1 ($\mathbf{LV}_1$) represents the *design* of the product. Its significant variables include number of company patents, total product part count, product sustainability, product functions, flows, and cost. Factor 2 ($\mathbf{LV}_2$) represents the *performance* of the product. Important factors include power usage, total number of different materials used, total part count, number of different features, and number of different functions.

The structural sub-model can now be formulated. The model shown in Eq. 2 estimates each latent variable, **L**, based on the factored product attributes, **LV**, and each attribute variable coefficient, γ. Because of the homogeneity of the participants of the survey, socio-demographic attributes, **S**, are omitted from the model.

$$\mathbf{L} = \gamma_1 \mathbf{LV} + \gamma_2 \mathbf{S} \tag{2}$$

Table 5 shows the resulting model coefficients, γ, for the structural submodel.

Binary Choice Model

With the structural and measurement model, the binary logit choice model that models the relationship between the qualitative latent attributes of the product (**L**) and a product's selection as an innovative product is created. The respondents were asked to choose one of the products as the innovative product as part of the survey. Logistic regression is used because there is a binary choice of innovative product or not which measures the relationship between a categorical variable (i.e. choice) to a continuous variable (i.e. latent variable). This choice model models choice as a function of product *innovativeness*, product *usability*, and *company profile* in customer selection of an innovative product. The model provides a probabilistic model of choice using an intermediate observed *utility* function (W_i) as shown in Eq. 3.

$$\text{Utility} = W_i = \beta_1 \mathbf{L}_{1,i} + \beta_2 \mathbf{L}_{2,i} + \beta_3 \mathbf{L}_{3,i} \tag{3}$$

The choice model coefficients from the "all at once" HB model estimation are shown in Table 6. Using the latent attributes, the choice model assumes that the innovativeness of a product is explained by perceived attributes of the products, such as its *innovativeness* or *usability*, as opposed to quantitative measures, such as the product's weight or number of parts. This model links product attributes, the latent variables, and the selection of the product as innovative. Here, it is assumed that customers make choice decisions that maximize their utility, and that utility is a function of key customer desired attributes, **L**, and product attributes, **A**.

With the utility function W, the probability of a product i selected as innovative is a function of the observed utility, given by Eq. 4.

$$Pr(i : [1,2]) = \frac{e^{W_i}}{\sum_{j \in [1,2]} e^{W_j}} \tag{4}$$

Results and Discussion

Using the coefficients for each latent variable found in the choice model, and the values for each latent variable found in the structural model, observed utility can be calculated to estimate the probability a given product i is selected as innovative according to Eq. 4. To verify the model, the probability that the innovative product was chosen as innovative compared to the common product is computed. The probability of choice for each product is given in Table 7.

Table 6 Binary choice model coefficients

	Innovativeness β_1	Usability β_2	Company profile β_3
Utility (W)	3.916	−2.28	2.494

Table 7 Model estimation of probability of chosen as innovative

	Product	Pr(i:[1, 2])
Innovative	Dyson Air Multiplier™	**0.936**
Common	Holmes® Fan	0.064
Innovative	Powermat®	**0.607**
Common	Journey's Edge Charging Station	0.393
Innovative	Oliso® Smart Iron	0.381
Common	Proctor Silex® Iron	**0.619**
Innovative	KidSmart Vocal Smoke Detector	**0.541**
Common	First Alert Basic Smoke Alarm	0.459

As seen in the table, the model assigns a higher probability of innovativeness to the innovative product vs. the common product, with the exception of the iron. In this case, the model estimates that the common Proctor Silex Iron has a higher probability (0.619) of being selected as innovative than the Oliso Smart Iron. A reason for this could be that while the Oliso was theorized to be innovative, actual survey respondents did not view it as innovative. Also, the survey respondents may have not scored this iron highly (or consistently) on the indicators used in the measurement model. With this one exception, the model generally provides expected estimates. The SEM can be used by design teams as part of a trade study to understand the influence of design decisions on the perception of innovativeness for their product, as compared to a common product of the same type. To conduct the study, the designer quantifies the new product attributes **A** which, through the intermediate **LV**s, enter the choice utility function. The "innovativeness" probability can then be computed for the new product design and multiple design alternatives compared. A limitation of this approach is that the model predicts choice probabilities based upon only the set of respondents who completed the survey, in this case engineering students. A larger population study is required to achieve a model capable of making predictions over a more general population.

In terms of model fit, statistics can be computed to determine if the model adequately explains the data [40]. Generally 200 observations are recommended for fitting an SEM; however, fewer observations may be acceptable based on the fit statistics. The first measure is the model chi-square test to determine if the model explains the correlation in the data; a good model fit would provide a chi-square value less than a 0.05. This model meets the criteria with a chi-square value of effectively 0. The pseudo-R^2 statistic is a goodness of fit statistic, with the best possible model achieving a value of 1 and the worst possible model achieving a value of 0. This SEM has a pseudo-R^2 value of 0.63, which is classified as an acceptable fit (but not ideal). It is concluded that the estimated SEM fit is acceptable for making predictions, but with room for improvement. The fit could be improved by increasing the sample size, revising the indicators, or exploring different model structures.

Conclusions and Future Work

The model developed in this paper is a structural equation model (SEM) for use in estimating the innovativeness of a product. It incorporates the latent variables: product *innovativeness*, *usability*, and *company profile* of the manufacturer. The measure of product innovativeness is time variant in that as the technology is more understood and widespread, and as product success and influence increases, the perception of a product's innovativeness decreases. More unknown, less understood products are considered more innovative. The usability latent variable incorporates the indicators that are associated with the user's interaction with the product such as efficiency, quality, ease of use, etc. The company profile latent variable captures the influence a company's innovative identity plays on the perception of its products. Products from companies that are seen to be innovative receive a boost in their perception of innovation based on the company's innovative attitude. The Dyson Air Multiplier™ for example is perceived to be innovative partly because of the innovative nature of the Dyson Company. Their products use new technology, or utilize technology in a different way to create new products.

The usefulness of the product innovativeness model is for product development. The model presented herein results in a methodology to quantify the innovativeness of a new product, or design, based on physical attributes of that product for a given population of users. With components with different scores for innovativeness, and the connections between those components known, it is possible to develop conceptual designs with different target levels of innovativeness. These conceptual designs can be generated by a computer, and presented to the designer for further consideration.

In terms of future work, the usefulness of this method needs to be evaluated based on a real world design problem. Using target values for each of the latent variables, it is possible to calculate the design attributes needed to achieve those levels, and proceed with a detailed design accordingly. Additionally, a larger scale study is required to test the feasibility of the approach with a larger set of survey data and models containing more product design attributes. Convergence of the hierarchical Bayes approach with larger models must be studied to ensure that the approach is scalable to real design problems.

Acknowledgments The authors would like to thank the National Science Foundation for their support under Grant Nos. 0928076 and 0927745 from the NSF EDI Program. Any opinions or findings of this work are the responsibility of the authors, and do not necessarily reflect the views of the sponsors.

Appendix: Survey Questions Administered

The following survey questions were approved by the Oregon State University IRB board and were used as part of the survey developed for this research. Responses were on a scale of 1–7 with 1 being in very low agreement, 4 being somewhat in agreement, and 7 being in high agreement. The participants were given a purchased product for each of the products surveyed. The products were presented in their full assembled state.

1. What is your perception on how successful the product has been?
2. How successfully does the product function?
3. What is your perception of how successful the product has been on a global scale?
4. How influential is the product to society?
5. How efficiently does the product perform its task?
6. How well known is the company that made the product?
7. How influential is the company that produced the product?
8. How well was the product made?
9. How complex is the product?
10. How original is each product? i.e. is it based on a product that already exists?
11. How beneficial is the product to its user?
12. How aesthetically appealing is the product?
13. How familiar are you with the product's design and functionality?
14. How easy is this product to use?
15. How satisfied are you with the amount of features present in the product?
16. How "high tech" is the product?
17. How "trendy" is the product?
18. How adjustable is the product?
19. How sustainable is the product?
20. How risky does the product appear to be?

References

1. Moreau CP, Lehmann DR, Markman AB (2001) Entrenched knowledge structures and consumer response to new products. J Mark Res 38:14–29
2. Chen W, Hoyle C, Wassenaar HJ (2013) Decision-based design. Springer, London
3. Shapiro AR (2006) Measuring innovation: beyond revenue from new products. Res Technol Manag 49:42–51
4. Arundel A, Kabla I (1998) What percentage of innovations are patented? Empirical estimates from European firms. Res Policy 27:127–141
5. Carayannis EG, Provance M (2008) Measuring firm innovativeness: towards a composite innovation index built on firm innovative posture, propensity and performance attributes. Int J Innov Reg Dev 1:90–107
6. Mairesse J, Mohnen P (2002) Accounting for innovation and measuring innovativeness: an illustrative framework and an application. Am Econ Rev 92:226–230

7. European Commission (2012) Innovation Union Scoreboard 2011. www.proinno-europe.eu
8. Hurley RF, Hult GTM (1998) Innovation, market orientation, and organizational learning: an integration and empirical examination. J Mark 62:42–54
9. Alegre J, Lapiedra R, Chiva R (2006) A measurement scale for product innovation performance. Eur J Innov Manag 9:333–346
10. Danneels E (2002) The dynamics of product innovation and firm competences. Strateg Manag J 23:1095–1121
11. Shah J, Kulkarni SV, Vargas-Hernandez N (2000) Evaluation of idea generation methods for conceptual design: effectiveness metrics and design of experiments. J Mech Des 122:377–384
12. Oman SK, Tumer IY, Wood K et al (2012) A comparison of creativity and innovation metrics and sample validation through in-class design projects. Res Eng Des 24:1–28
13. Lanjouw JO, Schankerman M (2004) Patent quality and research productivity: measuring innovation with multiple indicators. Econ J 114:441–465
14. Mairesse J, Mohnen P (2004) The importance of R&D for innovation: a reassessment using French survey data. National Bureau of Economic Research, Cambridge, MA
15. Saunders MN, Seepersad CC, Holtta-Otto K (2011) The characteristics of innovative, mechanical products. J Mech Des 133:21009
16. Garcia-Munoz S, Settell D (2009) Application of multivariate latent variable modeling to pilot-scale spray drying monitoring and fault detection: monitoring with fundamental knowledge. Comput Chem Eng 33:2106–2110
17. Everitt BS (1984) An introduction to latent variable models. Chapman and Hall Ltd, New York
18. Vermunt JK, Magidson J (2005) Factor analysis with categorical indicators: a comparison between traditional and latent class approaches. New developments in categorical data analysis for the social and behavioral sciences. Psychology Press, London
19. Muthen BO (1992) Latent variable modeling in epidemiology. Alcohol Health Res World 16:286–292
20. Jaeckle CM, MacGregor JF (1998) Product design through multivariate statistical analysis of process data. AIChE J 44:1105–1118
21. MacGregor JF, Yu H, García Muñoz S et al (2005) Data-based latent variable methods for process analysis, monitoring and control. Comput Chem Eng 29:1217–1223
22. Kim J, Huang Q, Shi J (2007) Latent variable based key process variable identification and process monitoring for forging. J Manuf Syst 26:53–61
23. Yacoub F, MacGregor JF (2011) Robust processes through latent variable modeling and optimization. AIChE J 57:1278–1287
24. Wassenaar HJ, Chen W, Cheng J et al (2004) An integrated latent variable choice modeling approach for enhancing product demand modeling. In: Proceedings of the 2004 ASME Design Engineering Technical Conference, American Society of Mechanical Engineers, New York
25. Leder H, Carbon C-C (2005) Dimensions in appreciation of car interior design. Appl Cogn Psychol 19:603–618
26. Trigo A, Vence X (2011) Scope and patterns of innovation cooperation in Spanish service enterprises. Res Policy 41:602–613
27. Esposti R, Pierani P (2000) Modelling technical change in Italian agriculture: a latent variable approach. Agric Econ 22:261–270
28. Lopes LF (2010) Innovation as a latent variable: an alternative measurement approach. Universidade Nova de Lisboa and CEFAGE, Lisbon
29. Orme B (1998) Getting started with conjoint analysis: strategies for product design and pricing research. Research Publishers LLC, Madison
30. Loehlin JC (1987) Latent variable models: an introduction to factor, path, and structural analysis. Lawrence Erlbaum Associates, Hillsdale
31. Walker J, Ben-Akiva M (2002) Generalized random utility model. Math Soc Sci 43:303–343
32. Spiegelhalter DJ, Thomas A, Best NG et al (2003) WinBUGS version 1.4. http://www.mrc-bsu.cam.ac.uk/bugs
33. Best of What's New 2010. http://www.popsci.com/bown/2010

34. Best Inventions of 2006. http://www.time.com/time/specials/packages/0,28757,1939342,00.html
35. IDEA 2006 Gallery. http://www.idsa.org/content/panel/idea-2006-gallery
36. Air Multiplier Technology. http://www.dyson.com/fans/fansandheaters/air-multiplier-technology.aspx
37. 2011 Best of What's New. https://www.popsci.com/bown2011/html/index/judging
38. Kaiser HF (1958) The varimax criterion for analytic rotation in factor analysis. Psychometrika 23:187–200
39. Song X-Y, Lee S-Y (2012) Basic and advanced Bayesian structural equation modeling: with applications in the medical and behavioral sciences. Wiley, Chichester
40. Hooper D, Coughlan J, Mullen MR (2008) Structural equation modelling: guidelines for determining model fit. Electron J Bus Res Methods 6:53–60

Index

J.S. Gero, S. Hanna (eds.), *Design Computing and Cognition '14*,
DOI 10.1007/978-3-319-14956-1