The Roles of Principles in Enterprise Architecture

H.A. (Erik) Proper[1,2] and D. (Danny) Greefhorst[3]

[1] Public Research Centre Henri Tudor, Luxembourg
[2] Radboud University Nijmegen, Nijmegen, The Netherlands
[3] ArchiXL, Amersfoort, The Netherlands

Abstract. Key concepts in enterprise architecture include concerns, principles, models, views and frameworks. While most of these concepts have received ample attention in research, the concept of principles has not been studied much yet. In this paper, we therefore specifically focus on the role of principles in the field of enterprise architecture, where we position enterprise architecture as a means to direct enterprise transformations.

In practice, many different types of architecture principles are used. At the same time, principles are referred to by different names, including architecture principles, design principles, and IT policies. The primary goal of this paper is, therefore, to arrive at a conceptual framework to more clearly clarify and position these different types.

The paper starts with a discussion on enterprise architecture as a means to govern enterprise transformation. This provides a framework to position the different types of principles, and highlight their roles in enterprise transformations.

1 Introduction

As discussed in [1], key concepts in the field of enterprise architecture include *concerns*, *principles*, *models*, *views* and *frameworks*. Ample research has been conducted on architecture frameworks, architecture modelling languages [2, 3], model analysis [4, 5], as well as viewpoints and concerns [6, 7, 8]. In this paper we turn our focus to the concept of *principles* and its role in the field of enterprise architecture. Given that principles have not received a lot of research attention [9], there is a need to better understand their essense.

Several approaches to enterprise architecture position principles as an important ingredient [10, 11, 1, 12, 13], while some even go as far as to position principles as being the essence of architecture [14]. At the same time, initial case studies [15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25] indicate there to be a wide variation in the actual use of principles. The primary aim of this paper is therefore to arrive at a first version of a conceptual framework which more clearly identifies and positions the different types of principles.

The framework presented in this paper is the first iteration in a design science [26] driven research effort in which we aim to more clearly define the concept of architecture principles, and develop an associated methodology for defining and describing architecture principles. This first iteration, provides a synthesis of existing views on enterprise architecture and enterprise engineering [27, 1, 14, 13].

E. Proper et al. (Eds.): TEAR 2010, LNBIP 70, pp. 57–70, 2010.

The remainder of this paper is structured as follows. Before we are able to sensibly explore the different types of principles, and their roles in enterprise transformations, Section 2 offers a review of our understanding of the fundamental purpose of architecture as a means to direct enterprise transformation. In Section 3, we then provide a conceptual framework of the different types of principles that can be discerned within our field.

2 Architecture as a Means to Govern Enterprise Transformations

In line with [1], we take the perspective that enterprise architecture should play a pivotal role in governing the continuous improvement process of an enterprise. In order to better understand the governing role of enterprise architecture, this section positions architecture as a means to govern enterprise transformations. As we will see, principles are the key means to govern the direction of the transformation of an enterprise.

In our view, governing enterprise transformations first and foremost entails the perspective on an enterprise as a purposely designed and implemented artefact. This enables the governing system to govern the enterprise transformation in terms of a clear goal, its current state, and the desired future states of the enterprise. Doing so, implies a perspective on properly governed enterprise transformation as being a form of *engineering*. This gives rise to the field of *enterprise engineering* [28, 14], which is an emerging discipline that regards the design and implementation of enterprises from an *engineering* perspective. Two key paradigms underpin this discipline. The first paradigm states that enterprises are purposefully designed and implemented systems. Consequently, they can be re-designed and re-implemented if there is a need for change. The second paradigm of enterprise engineering is that enterprises are primarily social systems, supported by technical systems. This means that the dominant system elements are social individuals, and that the *essence* of an enterprise's operation lies in the entering into and complying with commitments between these social individuals, while the implementation of this *essence* involves the design of an orchestrated collaboration between social beings and technical artefacts.

In line with [29, 1], the governance of an enterprise transformation process is regarded as involving a force-field between *enterprise strategy*, *programme management* and *enterprise architecture*. When only considering the typical project parameters, one runs the risk of conducting "local optimisations" at the level of specific projects. For example, when making design decisions which have an impact that transcends a specific project, projects will still aim for solutions that provide the best cost/benefits trade-off within the scope of that specific project while not looking at the overall picture. Such local optimisations are likely to damage the overall quality of the result of the transformation [1]. *Enterprise architecture* is concerned with an operationalisation of the direction in which the enterprise aims to transform itself, in terms core properties of the enterprise being engineered. This operationalisation allows the different change projects to be assessed whether they contribute to the realisation of the strategy, while guarding the properties that transcend specific projects.

In this paper we focus on the position of enterprise architecture in relation to enterprise engineering, and the potential roles of principles within this. From that context it is usefull to see architecture as: *the normative restriction of design freedom* [14], allowing enterprise architecture to exert its governing role towards the enterprise transformation. It stresses an important goal of architecture: *to restrict design freedom*, which can also be interpreted as "*to reduce design stress*". This does not exclude architecture as a means for other goals. Indeed [2, 1] classifies architecture viewpoints into *designing*, *deciding*, *contracting* and *informing* viewpoints. Furthermore, in [1] enterprise architecture is positioned explicitly as a means for informed governance of enterprise transformation, requiring indicators and controls to govern enterprise transformations.

The desire to restrict design freedom implies normative instruments with which such restrictions can be made. We believe architecture principles are key instruments in this [30], and we are certainly not alone in doing so. Several approaches position principles as an important ingredient [31, 32, 10, 11, 1, 13, 33, 34], while some even go as far to position principles as being the essence of architecture [14]. Architecture principles fill the gap between high-level strategic intentions and concrete designs. They ensure that the enterprise architecture is future directed, and can actually guide design decisions, while preventing *analysis paralysis* by focussing on the essence. Furthermore, they document fundamental choices in an accessible form, and ease communication with all those affected. They are formulated based on drivers such as strategy, goals and risks. Potential undesired impact on the goals of stakeholders can be reduced by formulating architecture principles.

3 A Conceptual Framework for Architecture Principles

As argued before, we take the perspective that architecture principles are a cornerstone of enterprise architecture. The goal of this section, is to provide a conceptual framework for architecture principles. As mentioned before, the framework presented in this paper is the first iteration in a design science [26] driven research effort in which we endeavour to more clearly define the concept of architecture principles, and develop an associated methodology for defining and describing architecture principles. The first iteration as presented in this paper, provides a synthesis of existing views on enterprise architecture and enterprise engineering [27, 1, 14, 13].

3.1 History

The term *principle* is said to originate from the Latin word of *principium* [35], which means "origin", "beginning" or "first cause". Vitruvius, an architect in ancient Rome, used principles to explain what is true and indisputable, and should apply to everyone. Vitruvius considered principles as the elements, the laws of nature that produce specific results. For instance, he observed how certain principles of the human body, such as symmetry and proportion, ensure "perfection". The human body was a great source of inspiration to him. He even believed that the principles of the human body should also be applied in the design of gardens and buildings because it would always lead to a perfect result: an ultimate combination of beauty, robustness and usability.

When using principles in the sense of *beginning*, they generally provide insight into the causes of certain effects. These causes can be *laws of nature*, *beliefs* or *rules of* *conduct*. *Laws of nature* simply *are*, and influence the things we do. Examples of such principles are the *law of gravity* and the *Pauli exclusion principle*. The latter is a quantum mechanical principle formulated by Wolfgang Pauli in 1925. It states that no two identical fermions may occupy the same quantum state simultaneously. Another example, more directly relevant to enterprise engineering, is the principle of *requisite variety* from general systems theory, which states that a regulating system should match the variety of the system that should be regulated [36].

Beliefs are typically founded in moral values. Examples of such principles are Martin Luther King's *principles of nonviolence*, that were to guide the civil rights movement. In an enterprise engineering context, examples of such principles would be: *No wrong doors* (suggesting that clients should be helped at whichever office/desk they approach the enterprise) and *The customer is always right*.

Rules of conduct are explicitly defined to influence behaviour, and are typically based on facts and beliefs. General examples include the *Ten Commandments* from the Bible, e.g. "*You shall not murder*" and "*You shall not commit adultery*". In our enterprise engineering context, examples would be: *Clients can access the entire portfolio of services offered by any part of the government by way of all channels through which government services are offered* and *Before delivering goods and services to external parties, we must hold receipt of the associated payment*.

The remainder of this section will show various dimensions in which principles can be positioned. We distinguish scientific principles from normative principles, positioning architecture principles as normative principles. We divide normative principles into credos and norms, in which the latter form is needed in order to provide enough restriction of design freedom. We show how principles relate to requirements and instructions. Finally, we position architecture as a form of essential design, focusing on the fundamental and essential aspects [37].

3.2 Scientific Principles versus Normative Principles

The American Engineers' Council for Professional Development [38] states that engineering concerns "*the creative application of* **scientific principles** *to design or develop structures, machines, apparatus, or manufacturing processes, or works utilising them* ...". Principles are used in a wide range of engineering disciplines such as industrial engineering, chemical engineering, civil engineering, electrical engineering and systems engineering. They can be seen as a form of design knowledge that should be shared, in order to increase the quality of designs. In line with [38], we will refer to these principles as *scientific principles*.

Scientific principles are likely to be cross-disciplinary in the sense that they will be applicable in various design disciplines. Lidwell [39] provides a list of 100 "universal principles of design", consisting of laws, guidelines, human biases, and general design considerations. The principles can be used as a resource to increase cross-disciplinary knowledge and understanding of design, promote brainstorming and idea generation for design problems, form a checklist of design principles, and to check the quality of design processes and products. Examples of principles described by Lidwell that fall into the category of scientific principles are the "*exposure effect*" and "*performance load*". The first principle states that "*repeated exposure to stimuli for which people* *have neutral feelings will increase the likeability of the stimuli*". The latter states "*the greater the effort to accomplish a task, the less likely the task will be accomplished successfully*".

Principles have always played an important role in civil engineering, a professional engineering discipline that deals with the design, construction and maintenance of the physical and naturally built environment, including works such as bridges, roads, canals, dams and buildings. Principles from general systems theory, such as the earlier mentioned law of *requisite variety* [36], are examples of scientific principles that are applicable in an enterprise engineering context.

Architecture principles are commonly seen as *normative principles*. TOGAF [13] states that "principles are general rules and guidelines, intended to be enduring and seldom amended, that inform and support the way in which an enterprise sets about fulfilling its mission". The use of principles in the context of enterprise architecture can be traced back to a multi-year deep dive research project led by Michael Hammer, Thomas H. Davenport, and James Champy, called the Partnership for Research in Information Systems Management (or PRISM) [40], which was sponsored by about sixty of the largest global companies (DEC, IBM, Xerox, Texaco, Swissair, Johnson and Johnson, Pacific Bell, AT&T, et cetera). It is a principles based architecture framework, also involving core terminology of, what was at that stage, a novel paradigm. In this context, principles were defined as "*simple, direct statements of an organisation's basic beliefs about how the company wants to use IT in the long term*". Note that in this definition, the operative word is *wants*. It refers to the fact that fundamentally, such principles are used to express a *normative desire*. Even more, it also expresses how these principles will aim to bridge the communication gap between top management and technical experts. The PRISM model, being from 1986, is among the first published enterprise architecture framework, and as such actually precedes the Zachman framework [41] (published one year later). PRISM's concept of principles as well as how they guide the definition and evolution of architectures was its most salient and widely accepted contribution.

The PRISM model has strongly influenced other enterprise architecture standards, methods and frameworks. The earliest publications referring to the concept of principle, in an enterprise architecture context, can indeed be traced back to the PRISM project [31, 32]. Furhermore, the HP Global Method for IT Strategy and Architecture [34, 42], which is based on work at Digital Equipment Corporation starting in 1984, was almost completely based on the PRISM model and the concept of principles. Many years later, the PRISM report [40] also influenced the IEEE definition of architecture, as many of the IEEE 1471 [27] committee members (Digital included) were employed by the original sponsors of this early work. The concept of *architecture principle* as it is defined in TOGAF [13] today, is also inspired by the PRISM model.

Normative principles do not exist in isolation. They are based on all sorts of other artefacts, such as the strategy, issues, the existing environment and external developments. On the other hand, they also influence all sorts of other artefacts, such as guidelines, requirements, designs and implementations. One can regard the *normative principles* as bridging between strategy and operations; they are primarily an alignment instrument. They are formulated based on knowledge, experience and opinions of all sorts of people in the organisations; senior management, as well as the people that do the actual work. This mixture of people is also the target audience of normative principles. In that sense, the definitions of normative principles also provides a common vocabulary for the organisation.

3.3 Credos versus Norms

In practice we see normative principles at various levels of precision. In [24] we have made the distinction between *architecture principles* and *guidelines*, where guidelines are more specific than architecture principles. NORA [43] distinguishes between *fundamental principles* and *derived principles*, where fundamental principles are the basis for derived principles. The level of precision influences the ability to assess the compliance of a design or architecture to the principle. When considering the role of principles bridging between strategy, via architecture to design, this is quite natural. At first, a principle will be formulated rather informally and refined later on in order to use it as a means to restrict design freedom. The definition of the word *principle* in the dictionary [35] suggests multiple forms of principles:

- *1a: a comprehensive and fundamental law, doctrine, or assumption b (1): a rule or code of conduct (2): habitual devotion to right principles <a man of principle> c: the laws or facts of nature underlying the working of an artificial device,*
- *2: a primary source: origin,*
- *3a: an underlying faculty or endowment <such principles of human nature as greed and curiosity> b: an ingredient (as a chemical) that exhibits or imparts a characteristic quality,*
- *4: Christian Science: a divine principle: god.*

In terms of the general definition, *scientific principles* refers to the interpretation of principles as *laws or facts of nature underlying the working of an artificial device*, *normative principles* refers to principles in the sense of a *a comprehensive and fundamental law, doctrine, or assumption* or a *rule of conduct* that guide changes in the enterprises by influencing/directing the design of these changes.

At the start of their life-cycle, normative principles are just statements that express the enterprise's fundamental belief of how things ought to be. At this stage, their exact formulation is less relevant. This is in line with intentions behind TOGAF [13] and the Zachman [41] framework, where the architecture process starts with the creation of an architecture vision. In this phase, architecture is very future-oriented and mostly a creative process. The principles can be used as a means to express a vision, which is mostly based on personal beliefs of the stakeholders involved in the envisioning. They can be seen as normative principles in their initial stage. They are not yet specific enough to actually use them as a norm. In other words; assessing compliance of architectures and designs to these principles is not feasible. They are primarily used as a source of inspiration. Examples of principles in this phase, taken from practical cases, are:

- *We should follow citizen logic.*
- *Work anywhere; anytime.*
- *Reuse as much as possible.*
- *Applications should be decoupled.*

Principles in this phase can best be referred to as being a *credo*. The dictionary [35], defines credo as: "*a set of fundamental beliefs; also: a guiding principle*". This is very close to the definition of principle by Beijer [34]: "*A fundamental approach, belief, or means for achieving a goal...*". In terms of the dictionary definition of principle, we consider this to correspond to its interpretation as *a comprehensive and fundamental law, doctrine, or assumption*. As such, *credos* are things an enterprise consciously chooses to adopt. They represent the fundamental beliefs or assumptions underpinning further architectural decisions. This allows enterprises to provide a first elaboration of an enterprise's strategy towards the desired design of the enterprise.

When enterprises want to use normative principles as a way to actually *limit* design freedom, the principles need to be more specific. This is when the exact formulation of the principle becomes important. They need to be formulated in such a way that compliance to them can be assessed. This starts with a reformulation of the principle statement, but extends to other properties. The full specification will a.o. need to contain definitions of terminology used, as well as a definition of how to assess the compliance of a design to the principle. The examples given previously could be reformulated as follows to make them more specific:

- *The status of customer requests is readily available inside and outside the organization.*
- *All workers are able to work in a time, location and enterprise independent way.*
- *Before buying new application services, it must be clear that such services cannot be rented, and before building such application services ourselves, it must be clear that they can not be purchased.*
- *Communication between application services will take place via an enterprise-wide application service bus.*

Once normative principles have been (re)formulated specific enough to use them to restrict design freedom, we can refer to them as a *norm*. The dictionary [35], defines a norm as: *a principle of right action binding upon the members of a group and serving to guide, control, or regulate proper and acceptable behaviour*. In terms of the dictionary definition of principle, we consider this to correspond to its interpretation as *rule of conduct*. Norms can also be regarded as a tactic by which a *credo* can be enforced. To indeed enable the normative effect of norms, they are required to be *specific*, *measurable*, *attainable* and *relevant*.

When considering TOGAF's [13, Section 3.17] definition of principle:

> *A qualitative statement of intent that should be met by the architecture. Has at least a supporting rationale and a measure of importance.*

and more specifically the purpose it attributes to such principles [13, Section 36.2.4]:

> *Principles are general rules and guidelines, intended to be enduring and seldom amended, that inform and support the way in which an enterprise sets about fulfilling its mission.*

we take the stance that TOGAF requires/presumes architecture principle to be in the form of *norms*.

3.4 Principles versus Requirements and Instructions

Normative principles limit design freedom. They are, however, not the only statements which limit design freedom. Requirements also limit design freedom. However, requirements state *what* (functional or constructional) properties a (class of) system(s) should have, and *why* the stakeholders want the (class of) systems to have these properties (also see [34]). Normative principles provide policies on *how* the design of the (class of) system(s) will ensure that the actual implemented system(s) will meet the requirements. Requirements are the basis for solutions, expressing their required characteristics. Fisher [9] states that architecture principles refer to the construction of an enterprise while requirements refer to its function.

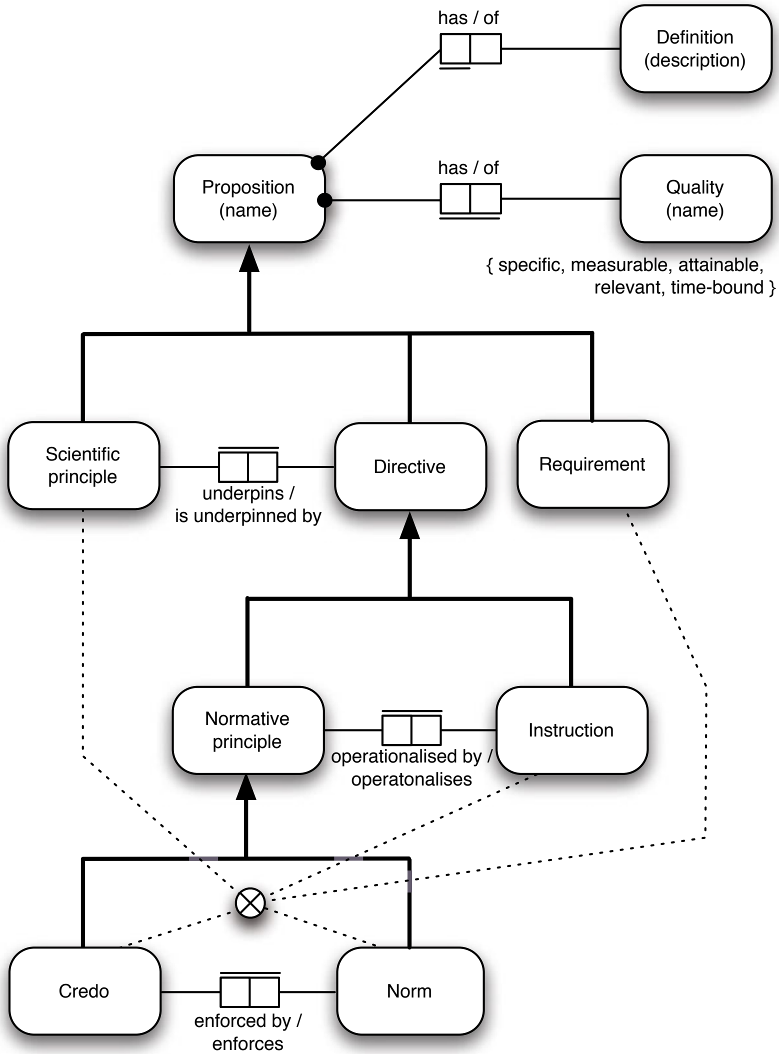
Generally, enterprise architectures are not only specified in terms of *normative principles*, but also in terms of more instructive statements, such as models and detailed descriptions on how to apply these in a specific situation. We will refer to these statements as *instructions*, since they tell designers specifically what to do and what not to do. Instructions will refer to the concepts used in the actual construction of the enterprise, such as: value exchanges, transactions, services, contracts, processes, components, objects, building blocks, et cetera. Enterprises typically use languages such as UML [44], BPMN [45], TOGAF's [13] content framework, ArchiMate [3], or the language suggested by the DEMO method [28], to more explicitly express their architectures in terms of concrete modelling concepts. *Instructions*, provide a more operational and tangible refinement of the *normative principles*. Due to their tangible nature, in terms of actual concepts used in the construction of the enterprise, architecture models enable enterprises to study/analyse the effects of different options for the future, as well as analyse problems in the current situation [2]. Just as normative principles, instructions are required to be *specific*, *measurable*, *attainable* and *relevant*

Collectively we will refer to *normative principles* and *instructions* as *directives* to express the fact that they both direct the design of the enterprise (albeit at different levels of specificity) and both involve a choice by the enterprise to direct their transformation. The dictionary [35] defines directive as: *serving or intended to guide, govern, or influence*, while the OMG's business motivation model [46] also uses the notion of *directive* as the most general form of guidance/regulation. In terms of the NAF definition of architecture [14], these two flavours of *directive* collectively cover its role as a normative restriction of design freedom.

Figure 1 provides (in the style of Object Role Modelling [47]) a domain model positioning *credos*, *norms*, *normative principles*, *instructions*, *requirements* and *scientific principles*. In the ORM diagram, the encircled cross is used to signify the fact that *credos*, *norms*, *scientific principles*, *instructions* and *requirements* are mutually exclusive. The general notion of *proposition* is used as a further generalisation of *scientific principles*, *requirements* and *directives*. Each *proposition* must have a *quality* and a *definition* (signified by the black dot in the diagram), while they have at most one *definition* (signified by the short bar on the fact type).

3.5 Architecture Principles versus Design Principles

Regarding an architecture as a *normative restriction of design freedom*, raises the question what the difference is between architecture and design. More operationally, *What*



A Credo **must** have Qualities: { attainable, relevant, time-bound }
A Norm **must** have Qualities: { specific, measurable, attainable, relevant, time-bound }
A Instruction **must** have Qualities: { specific, measurable, attainable, relevant, time-bound }
A Scientific principle **must** have Qualities: { specific, measurable }
A Requirement **must** have Qualities: { specific, measurable, attainable, relevant, time-bound }

Fig. 1. Core terminology

should be included in an architecture, and thus restrict the freedom of ensuing design activities, and what should indeed be left to designers? As suggested by the IEEE [27] and TOGAF [13] definitions of architecture, the architecture level should focus on fundamental aspects. An enterprise architecture should provide an elaboration of an enterprise's strategy, while focussing on the core concerns of the stakeholders. As such, an architecture is typically positioned at a level concerned with a class of systems. A design focuses on the remaining requirements and design decisions pertaining to a specific system being developed, which will typically have a limited impact on the key concerns of the stakeholders.

Fehskens [37] states that architecture should explicitly address alignment, relating the role of architecture to the mission. He redefines architecture as "*those properties of a thing and its environment that are necessary and sufficient for it to be fit for purpose for its mission*". In his view, architecture should focus on what is essential, on "the stuff that matters". This equates to those properties that are necessary and essential. This is also what distinguishes architecture from design. A different architecture implies a different mission, whilst different designs may address the same mission.

Rivera [42] acknowledges that architecture is about the essense. He adds that generally speaking, design work seeks to find optimal solutions to wellunderstood problems. It's more science than art, algorithmic in nature, and deals mostly with a system's measurable attributes. Architecting deals primarily with nonmeasurable attributes using nonquantitative tools and guidelines based on practical lessons learned. In his view, the architecture uses a heuristic approach. Whereas design and engineering work is primarily deductive in nature, architecture work is primarily inductive.

The distinction between design and architecture, also allows us to distinguish between *architecture* and *design* versions of *normative principles*, *instructions* and *requirements* respectively:

– *Architecture principles* are normative principles that are included in the architecture of a class of systems.
– *Architecture instructions* are instructions that are included in the architecture of a class of systems.
– *Architecture requirements* are requirements that pertain to the architecture of a class of systems.
– *Design principles* are normative principles included in the design of a specific system.
– *Design instructions* are instructions that are included in the design of a specific system.
– *Design requirements* are requirements that pertain to the design of a specific systems.

The role of these concepts is made even more explicit in Figure 2. In this diagram, concepts from Figure 1 are shown in their architecture and design variants, and guided by scientific principles. The diagram illustrates the flow from strategy via requirements that should already be addressed at the level of the architecture, the actual architecture, the requirements to be met by the design of the system, the design of the system, to the implementation of the system implementation. The scientific principles can be applied during the entire engineering process to motivate design decisions. Note that principles

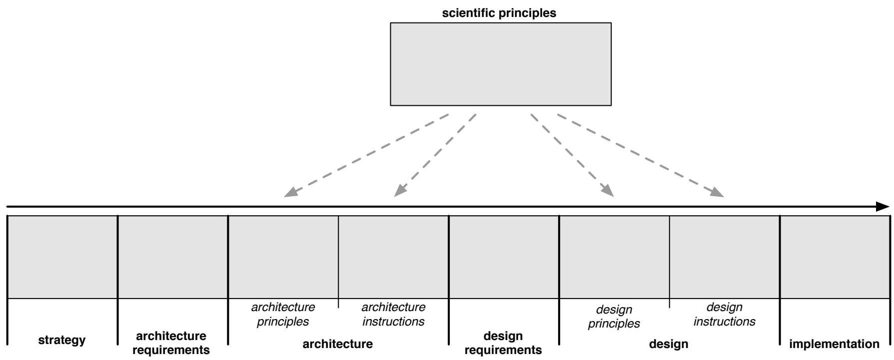


Fig. 2. From strategy to implementation

are much less common in designs than in architectures. Rivera [42] even states that "*One of the key differences between the content of descriptions generated from architecture work and that generated from design work is in their use of principles*".

As an illustration of the flow from strategy to implementation, we use a fictitious insurance company. Their strategy is to become the provider of the cheapest insurances in the country. To this end they have formulated the objective to cut costs with 20% within two years, which can be considered an architectural requirement. Based on this requirement they have defined an architecture principle which states that "business processes are standardized and automated". Although they could not find any scientific principles to support this, they had good experiences with process standardization in other organizations. The architecture principle is translated to specific architectural instructions on the claims handling process. These instructions standardize the process by defining the specific activities which must be present in all claims handling processes. A new claims handling system is designed to support the standardized claims handling process. A design requirement for this system is that it integrates with the recently developed customer portal. The lead designer strongly believes that business rules should be defined and implemented separately from other application functionality in this claims handling system and therefore defines the design principle that "business rules are defined in a business rules engine". He also provides more specific design instructions on how to actually define these business rules, by prescribing the specific constructs in the business rules engine that should be used. These design instructions are used by the developers that use the rules engine to implement the system.

4 Conclusion

In this paper we have explored the concept of *principle* in relation to enterprise transformations, leading to a conceptual framework more clearly defining *principle* and associated terminology.

The presented framework is the first iteration in a design science [26] driven research effort in which we aim to more clearly define the concept of architecture principles, and develop an associated methodology for defining and describing architecture principles. We have produced a domain model of the concepts involved, taking into account established definitions as well as practical experiences. While the proposed framework is a synthesis of existing theoretical perspectives as well as empirical insights, in line with the design science approach, the necessary next step is to validate this framework in terms of additional practical cases and experiments. With the current conceptual framework in place, we can indeed endeavour to do so.

References

1. Op't Land, M., Proper, H., Waage, M., Cloo, J., Steghuis, C.: Enterprise Architecture – Creating Value by Informed Governance. Springer, Berlin (2008) ISBN-13: 9783540852315
2. Lankhorst, M., et al.: Enterprise Architecture at Work: Modelling, Communication and Analysis. Springer, Berlin (2005) ISBN-10: 3540243712
3. Iacob, M.E., Jonkers, H., Lankhorst, M., Proper, H.: ArchiMate 1.0 Specification. The Open Group (2009) ISBN-13: 9789087535025
4. Johnson, P., Ekstedt, M.: Enterprise Architecture: Models and Analyses for Information Systems Decision Making. Studentlitteratur (2007) ISBN-13: 9789144027524
5. Iacob, M.E.I., Jonkers, H.: Quantitative analysis of service-oriented architectures. International Journal of Enterprise Information Systems 3, 42–60 (2007)
6. Proper, H., Hoppenbrouwers, S., Veldhuijzen van Zanten, G.: Communication of Enterprise Architectures. In: [2], pp. 67–82, ISBN-10: 3540243712
7. Lankhorst, M., Torre, L.v.d., Proper, H., Arbab, F., Steen, M.: Viewpoints and Visualisation. In: [2], pp. 147–190, ISBN-10: 3540243712
8. Buckl, S., Ernst, A., Lankes, J., Matthes, F.: Enterprise Architecture Management Pattern Catalog - Version 1. Technical Report TB 0801, Technische Universität München, Garching bei München, Germany (2008)
9. Fischer, C., Winter, R., Aier, S.: What is an Enterprise Architecture Design Principle? Towards a consolidated definition. In: Proceedings of the 2nd International Workshop on Enterprise Architecture Challenges and Responses, Yonezawa, Japan (2010)
10. Tapscott, D., Caston, A.: Paradigm Shift – The New Promise of Information Technology. McGraw–Hill, New York (1993) ASIN 0070628572
11. Wagter, R., Berg, M.v.d., Luijpers, J., Steenbergen, M.v.: Dynamic Enterprise Architecture: How to Make It Work. Wiley, New York (2005) ISBN-10: 0471682721
12. Capgemini: Enterprise, Business and IT Architecture and the Integrated Architecture Framework. White paper, Utrecht, The Netherlands (2007)
13. The Open Group – TOGAF Version 9. Van Haren Publishing, Zaltbommel, The Netherlands (2009) ISBN-13: 9789087532307
14. Dietz, J.: Architecture – Building strategy into design. Netherlands Architecture Forum. Academic Service – SDU, The Hague (2008), `http://www.naf.nl` ISBN-13: 9789012580861
15. Lindström, A.: On the Syntax and Semantics of Architectural Principles. In: Proceedings of the 39th Hawaii International Conference on System Sciences (2006)
16. Lindström, A.: An Approach for Developing Enterprise-Specific ICT Management Methods – From Architectural Principles to Measures. In: IAMOT 2006 – 15th International Conference on Management of Technology, Beijing, China (2006)
17. Lee, C.: Aerospace Logistics architecture program: Action Research at Air France Cargo – KLM Cargo. Master's thesis, Delft Technical University, Delft, The Netherlands (2006)

18. Go, A.: Implementing Enterprise Architecture: Action Research at Air France Carge – KLM Cargo. Master's thesis, Delft Technical University, Delft, The Netherlands (2006)
19. Kersten, J.: Propositions. Master's thesis, Radboud University Nijmegen, Nijmegen, The Netherlands (2009) (in Dutch)
20. Boekel, K.v.: Architectuurprincipes: Functie en Formulering (Architecture Principles: Function and Formulation. Master's thesis, Radboud University Nijmegen, Nijmegen, The Netherlands (2009) (in Dutch)
21. Tillaart, M.v.d.: Propositions into a Framework. Master's thesis, Radboud University Nijmegen, Nijmegen, The Netherlands (2009)
22. Ramspeck, B.: Formulating Principles in an Effective Way. Master's thesis, Radboud University Nijmegen, Nijmegen, The Netherlands (2008)
23. Greefhorst, D., Proper, H., Ham, F.v.d.: Principes: de hoeksteen voor architectuur – Verslag van een workshop op het Landelijk Architectuur Congres 2007 (Principles: The Cornerstone of Architecture – A report of a workshop held at the Dutch National Architecture Congres 2007. Via Nova Architectura (2007) (in Dutch), `http://www.via-nova-architectura.org`
24. Greefhorst, D.: ICT bibliotheek. In: Ervaringen met het opstellen van architectuurprincipes bij een verzekeraar (Experiences with the formulation of architecture principles at an insurance company), vol. 35, ch. 2, pp. 53–62. Academic Service – SDU, The Hague (2007) (in Dutch) ISBN-13: 9789012119511
25. Bouwens, S.: DYA Architectuurprincipes – Deel 1: Basics (DYA Architecture Principles – Part 1: Basics). White paper, Sogeti, The Netherlands (2008) (in Dutch)
26. Hevner, A., March, S., Park, J., Ram, S.: Design Science in Information Systems Research. MIS Quarterly 28, 75–106 (2004)
27. The Architecture Working Group of the Software Engineering Committee, Standards Department, IEEE: Recommended Practice for Architectural Description of Software Intensive Systems. Technical Report IEEE P1471:2000, ISO/IEC 42010:2007, The Architecture Working Group of the Software Engineering Committee, Standards Department, IEEE, Piscataway, New Jersey (2000) ISBN-10: 0738125180
28. Dietz, J.: Enterprise Ontology – Theory and Methodology. Springer, Berlin (2006) ISBN-10: 9783540291695
29. Rijsenbrij, D., Schekkerman, J., Hendrickx, H.: Architectuur, besturingsinstrument voor adaptieve organisaties – De rol van architectuur in het besluitvormingsproces en de vormgeving van de informatievoorziening, Lemma, Utrecht, The Netherlands (2002) (in Dutch) ISBN-10: 9059310934
30. Op't Land, M., Proper, H.: Impact of Principles on Enterprise Engineering. In: Österle, H., Schelp, J., Winter, R. (eds.) Proceedings of the 15th European Conference on Information Systems, University of St. Gallen, St. Gallen, Switzerland, pp. 1965–1976 (2007)
31. Davenport, T., Hammer, M., Metsisto, T.: How executives can shape their company's information systems. Harvard Business Review 67, 130–134 (1989)
32. Richardson, G., Jackson, B., Dickson, G.: A Principles-Based Enterprise Architecture: Lessons from Texaco and Star Enterprise. MIS Quarterly 14, 385–403 (1990), `http://www.jstor.org/stable/249787`
33. Hartman, H., Hofman, A., Stahlecker, M., Waage, M., Wout, J.v.h.: The Integrated Architecture Framework Explained. Springer, Berlin (2010) ISBN-13: 9783642115172
34. Beijer, P., de Klerk, T.: IT Architecture: Essential Practice for IT Business Solutions. Lulu (2010)
35. Meriam–Webster: Meriam–Webster Online, Collegiate Dictionary (2003)
36. Beer, S.: Diagnosing the System for Organizations. Wiley, New York (1985)
37. Fehskens, L.: Re-Thinking architecture. In: 20th Enterprise Architecture Practitioners Conference, The Open Group (2008)

38. The Engineers' Council for Professional Development. Science 94, 456 (1941)
39. Lidwell, W., Holden, K., Butler, J.: Universal Principles of Design. Rockport Publishers, Inc., Massachusetts (2003)
40. CSC Index, Inc., Hammer & Company, Inc., Cambridge MA.: PRISM: Dispersion and Interconnection: Approaches to Distributed Systems Architecture, Final Report. Technical report, CSC Index, Inc. and Hammer & Company, Inc. Cambridge, MA (1986)
41. Zachman, J.: A framework for information systems architecture. IBM Systems Journal 26 (1987)
42. Rivera, R.: Am I Doing Architecture or Design Work? It Professional 9, 46–48 (2007)
43. ICTU: Nederlandse Overheid Referentie Architectuur 2.0 – Samenhang en samenwerking binnen de elektronische overheid (2007) (in Dutch), http://www.ictu.nl
44. OMG: UML 2.0 Superstructure Specification – Final Adopted Specification. Technical Report ptc/03–08–02, OMG (2003)
45. Object Management Group: Business process modeling notation, v1.1. OMG Available Specification OMG Document Number: formal/2008-01-17, Object Management Group (2008)
46. BMM Team: Business Motivation Model (BMM) Specification. Technical Report dtc/06–08–03, Object Management Group, Needham, Massachusetts (2006)
47. Halpin, T., Morgan, T.: Information Modeling and Relational Databases, 2nd edn. Data Management Systems. Morgan Kaufman, San Francisco (2008) ISBN-13: 9780123735683