Hermann Kühnle
Editor

Distributed Manufacturing

Paradigm, Concepts, Solutions and Examples

Springer

Distributed Manufacturing

Hermann Kühnle
Editor

Distributed Manufacturing

Paradigm, Concepts, Solutions and Examples

Springer

Editor
Hermann Kühnle, Prof. Dr.
Otto-von-Guericke-Universität
Institut für Arbeitswissenschaft,
Fabrikautomatisierung und Fabrikbetrieb (IAF)
Universitätsplatz 2
39106 Magdeburg
Germany
hermann.kuehnle@ovgu.de

ISBN 978-1-84882-706-6 e-ISBN 978-1-84882-707-3
DOI 10.1007/978-1-84882-707-3
Springer London Dordrecht Heidelberg New York

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

Library of Congress Control Number: 2009936771

Cover design: eStudioCalamar, Figueres/Berlin

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Foreword

In recent last years it has become obvious to many companies that, to succeed, they need more effective support for operations and to implement better organisational principles and practices. It is clear that the new industrial world is more complex and more difficult to manage. We need novel organisational and management paradigms in order to uncover and exploit new thinking. Innovation pressures have forced the emergence of Networked Organizations seeking to access innovation resources globally and to secure lasting efficiency competitive advantages.

The networked economy has been enabled and has developed as a result of information and communication technology (ICT) for interorganisational collaboration. Market environments have become much more agile and turbulent and organisations more open under the influence of the rapid advancement of information processing devices and network technologies. Therefore, business strategic and process management, which oversee the overall value creation of products, are the focus of many companies.

Nevertheless, companies have taken rather different approaches to respond to the changes. Some have formed strategic alliances to capture new market opportunities, some have improved existing relationships with their supply network members to enhance the entire value creation supply chains, and some are planning to make changes but in a rather slow manner. Actual cases of industrial implementation of Distributed Manufacturing have shown its effectiveness in responding to the new challenges of the current turbulent and competitive global market, although a number of various barriers still prevent their wider application.

The aim of this book is to promote the adoption of Distributed Manufacturing solutions already experienced by industry players as part of the realisation of the concurrent enterprise (CE) vision, established by the European CE community and represented by CE-NET, the concurrent enterprising network of excellence (CE-NoE), which aims at promoting top-notch European CE expertise from research, academy and industry. The CE-NET community, facilitated by the European Society of Concurrent Enterprise (ESoCE), has identified new paradigms and approaches to enable the adoption of concurrency principles at the level of distributed product development, networked enterprise and user driven innovation, under the new framework of concurrent innovation. For systematic approach to an examination of the whole body of CE knowledge, the issues have been organised into five main areas:

- Human aspects,
- Business model & organisation,
- ICT infrastructure,
- Product/service development and
- Policy and regulations.

This book provides a substantial contribution to the establishment of a concurrent innovation scientific base and thoroughly covers the first four areas with a stronger focus on technologies, platforms and standards characterising the Distributed Manufacturing context. This book addresses the main barrier that has so far limited the full adoption and exploitation of the emerging paradigms i.e. the issue of managing the complexity of Distributed Manufacturing, and it intelligently suggests how the theory of complex adaptive systems can provide some of the answers. It also sheds new light on the modes by which complex networks can manage parallelism, emergence, behaviour, iteration and encapsulation and in particular how the creative power of individuals and users can be integrated into the distributed processes with a multidisciplinary approach. Although enabled by technology, the initiators of the networked economy are people. Rapid and easy communication brings obvious benefits, but not all the increased communication is value-adding and people have less time to think. It is clear that there are many challenges for companies developing products and services. Some of these have been met; many more require additional efforts. In this book the editor has extracted and synthesized a carefully selected bundle of project results from the European projects PABADIS and PABADIS PROMISE and from closely related research. The invited authors present a number of theories and approaches that, adequately combined, may create suitable concepts for implementing highly workable solutions for Distributed Manufacturing.

In the new context of global industrial networks many concepts have been suggested to improve the base of support for geographically distributed networks and its effective collaboration. Mainly the proposals cover the socio-organisational field; some discuss information and communication technology impacts and options. This book draws from both fields and adds game theoretic and evolutionary elements pulling together principles, aspects and attributes that, even in industry, are mostly considered separate fields and entities that are difficult to unite.

Pick any random spot and you'll find inspiring ideas in this book. The book will serve as more than a detailed record of complex projects. It will provide an invaluable resource for all those wishing to enhance their understanding of changes in the manufacturing world in general and the introduction of new principles in particular. For that reason, I recommend that this book join the set of ready references available to you as together we practice and improve our profession of advanced manufacturing and extended enterprise.

Rome, April 2009

Roberto Santoro
President, European Society of Concurrent Enterprising (ESoCE)
Chairman, European Network of Living Laboratories (ENoLL)

Preface

> *"In anything at all, perfection is attained not when there is no longer anything to add, but when there is no longer anything to take away".*
> *(Antoine de Saint Exupery)*

Profound changes have already occurred in manufacturing within the last decades and the competitive environment for manufacturing will again be significantly different in the next 10 or 15 years. Major developments will occur in a number of different areas of manufacturing such as organisation, collaboration and globalisations resulting in Distributed Manufacturing in many cases. Distributed Manufacturing was originally focused on manufacturing architecture and control within single plants; later it was extended to the virtual manufacture of products and the networked organisation and includes all issues surrounding industrial networks. Key driving forces may be seen in all developments and trends in the fields of information and communication technology (ICT).

The gap between manufacturing automation and social actors' communication should be overcome.

This book represents a synthesis of selected key outcomes from the projects plant automation based on distributed systems (PABADIS) and PABADIS, based product oriented manufacturing systems for re-configurable enterprises (PABADIS'PROMISE), funded by the European Commission. The work on these projects was done through international collaboration over 8 years involving leading researchers as well as leading companies and renowned institutions in manufacturing systems control, embedded systems and network organisation world wide. The results have been consolidated with engineering communities and standardisation bodies.

The volume seeks to anticipate broadly emerging manufacturing structures and the respective information and communication technologies for organisations, their leaders and ICT strategists as well as researchers and technologists facing the challenges of their enterprises' geographical dispersion and network partners' dependencies. To this end, theoretical and application-oriented contributions have been included with a view to achieving the optimum breadth and depth of the relevant subject matter.

The book begins with on overview of methods and systems appropriate for concurrent product development in distributed structures. As many multisite companies and enterprise networks face competition in local contexts while having to keep the enterprises' advantages of common platforms and standards, co-evolution thinking has been chosen as a suitable new theoretical background and idea generator to cope with this growing challenge. The next part discusses new concepts of manufacturing management and novel ICT applications which may be unfamiliar to readers and challenge the status quo. As ICT advances are evidently occurring more rapidly at the machinery and equipment level with the respective execution systems, the next large next is devoted to multi-agent systems (MAS) as the central part of the project. The final part, outlining the most recent project results, links the world of agents to products and flexible manufacturing technologies, leading back to the first part and giving substantial clues to further developments as well as hot research topics.

More and more enterprises are faced with the huge, and thus far unseen, challenges of doing manufacturing efficiently in collaborative networks and distributed structures, and operating beyond the consolidated state of the art. For their support and to provide insight into recent developments and emerging concepts, this volume presents a number of ideas, concepts and solution approaches that, when combined in the right way, gives considerable help in responding to those challenges.

Stuttgart, May 2009

Hermann Kühnle

Contents

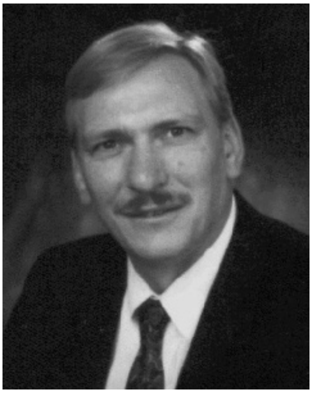
Acronyms

AARIA	autonomous agents at Rock Island Arsenal
AB	Ability Broker
ABMEI	agent-based manufacturing enterprise infrastructure
ADRENALIN	advanced fractal companies use information supply chain
AOA	Agent-Oriented Architecture
AR	augmented reality
BPM	business process modelling
BSCW	basic support for cooperative work
CAMAC	computer-automated measurement and control
CARE	c-agent runtime environment
CBA	component-based automation
CCITT	Comité Consultatif International Télégraphique et Téléphonique
CE	Concurrent Enterprise
CENELEC	Comité Européen de Normalisation Electrotechnique
CMU	Co-Operative Manufacturing Unit
CNO	collaborative network organisation
CSCW	computer-supported collaborative work communications
CSMA	carrier sense multiple access
CVE	collaborative virtual environment
CWE	collaborative working environment
DBMS	database management system
DCS	Distributed Control System
DEDEMAS	decentralised decision making and scheduling
DF	Device Function
DFP	Device Function Proxy
DIS	distributed interaction simulation
DM	Device Manager
DMU	digital mock-up
DMZ	demilitarised zone
DO	Device Observer
DP	Device Proxy
EA	expectation awareness
EDDL	Electronic Device Description Language

EM	Execution Manager
EN	events notification
ERP	Enterprise Resource Planning
FB	function block
FIPA	Foundation for Intelligent Physical Agents
FMS	Flexible Manufacturing System
GDP	gross domestic product
GPS	global positioning system
HLA	high-level architecture
HMI	human machine interface
HMS	Holonic Manufacturing System
I/O	Input/Output
IC	integrated circuit
IC	information collector
ICT	information and communication technology
IEC	International Electrotechnical Commission
IM	instant messaging
IMS	Intelligent Manufacturing System
IRC	internet relay chat
IRT	isochronous real time
ISO	International Organisation for Standardisation
JAKOBI	Java und komponentenbasierte Industriesteuerung
JVM	Java virtual machine
MAC	media access control
MAP	manufacturing application protocol
MAS	Multi-Agent System
MASCADA	managing production change and disturbance
MDE	Model-Driven Engineering
MES	Manufacturing Execution System
MMS	manufacturing message specification
MOM	manufacturing operations management
MRP	material requirements planning
MWC	mobile wearable computing
NO	node operator
NTP	network time protocol
OA	Order Agent
OAS	Order Agent Supervisor
OO	object orientation
OSI	open systems interconnection
P&P	plug & participation
PABADIS	Plant Automation Based on Distributed Systems
PABADIS' PROMISE	PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises

PABADIS' PROMISE-CD	PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises Control Device
PAHT	product and agent host tag
PDM	product data management
PDR	Product Data Repository
PID	proportional integral differential (continuous controller)
PIT	product identification tag
PLC	Programmable Logic Controller
PLM	product life cycle management
PO	Production Order
PROSA	Product-Resource-Order-Staff Architecture
PS	Process Segment
PTP	precision time protocol
PVC	professional virtual community
R&D	research and development
RA	Resource Agent
RAS	Resource Agent Supervisor
RFID	radio frequency identification
RFIT	radio frequency information technology
RMS	Re-Configurable Manufacturing System
SG	serious gaming
SIFB	Service Interface Function Block
SN	social networking
SOA	Service-Oriented Architecture
SW	shared workspace
TORERO	Total Life Cycle Web-Integrated Control
VAR	virtual and augmented reality
VR	virtual reality
WIP	Work In Progress
WO	Work Order

About the Editor



Hermann Kühnle holds a doctoral degree in mechanical engineering as well as a master's degree in mathematics, both from the University of Stuttgart. He joined the Otto-von-Guericke University Magdeburg, Germany, in 1994 as a full university professor for factory operations and production systems and as executive director of the Institute for Ergonomics, Manufacturing Systems and Automation, having previously served as lecturer, head of national applied research and responsible regional CIM coordinator and consultant. From 1994 to 2001 he was also foundation and executive director of the Fraunhofer Institute for Factory Operation and Automation, IFF, Magdeburg. Since 1995, Hermann Kühnle has been the spokesman for research on advanced production systems in Saxony-Anhalt. He is a member of boards of international journals, of companies and of venture capital groups.

0 Distributed Manufacturing: Paradigms, Concepts, Solutions and Examples

0.1 Introduction

A simultaneous presence in several regions and in different regional markets has become more and more essential for suppliers and manufacturers alike. These configurations are enforced by volatile market demands, fierce competition, and high innovation pressure in order to capture lasting advantages in efficiency. In particular companies that have experienced rapid international growth through mergers and acquisitions are suddenly faced with the challenges of structuring, managing and operating effectively a network of geographically dispersed factories with worldwide transfer of assembly and manufacturing operations for similar products between multiple production sites in different countries. This competitive global environment imposes the continuous need to identify and exploit new manufacturing paradigms, adapted methods and cutting edge technologies. Little attention has been paid so far to the fact that distributed manufacturing structures and their full advantages may be exploited best if concurrency of information flows and operations is strived for. The competitive power of distributed structures lies in their abilitiy to put entities all together and make the net concurrent customer-driven, involving organisation, processes and business models. As competition starts pressuring whole networks, fast linking and interoperability as well as adaptation abilities have become crucial attributes for manufacturing companies.

As production systems “disperse” their value chains, engaging more and smaller units all over the globe, value creation increasingly appears as a result of geographically distributed networked operations and services, representing in total at least the sum of all necessary resources. As the responsibilities for operations are strongly tied to organisational units and their socio-technical nature, Distributed Manufacturing also has all the features of human-influenced complex network building (e.g. trust, individual preferences) as well as the planning and execution of efficient processes within networks, fully engaging the scope of information and communication technologies (ICT) for repetitive process routines and standardised functionalities. The optimum basis for collaboration using the least amount of resources and time is to take significant steps towards parallelism of all actions and operations. Distributed Manufacturing in enterprise networks dynamically combines core competencies and knowledge of different entities to fully meet specific, narrowly defined market opportunities. All activities in distributed, temporary alliances of independent, co-operating manufacturers, customers and suppliers use systematic approaches, methods and advanced technologies for increasing efficiency in the design and manufacture of products and services by concurrency, integration, standardisation and teamwork for achieving common goals in global markets.

Driving technology spheres are adaptable, integrated equipment and systems that can be readily reconfigured, technologies that convert information into knowledge for effective decision making, enhanced human-machine interfaces as well as software for intelligent systems for collaboration provide for completely new opportunities. Once manufacturing structures are distributed in this sense, all successful set-ups definitely point to the emergence of a strongly ICT supported networked manufacturing world. Adequate ICT applications for modelling and simulation are also considered as extremely important to be able to quickly innovate, design and produce the 'right product right the first time'.

This volume summarises the most important results of the EU projects Plant Automation Based on Distributed Information Systems (PABADIS) and the successor for implementation PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises (PABADIS'PROMISE), which represent substantial advances in the field of control of Distributed Manufacturing network using concurrency principles. The consortia that have been led by the editor had involved 27 institutions and more than 150 researchers and experts from EU Europe, Switzerland and Canada. The project had been extended to the international context by joining the EU–Intelligent Manufacturing Systems program.

The project's key statement is clear: Distributed Manufacturing is a new pattern of interfirm relationships evolving networkwide integration by creating different forms of interentity processes. The underlying paradigms are a valid frame for next-generation planning systems and management procedures.

The major impacts on manufacturers enforcing these organisation patterns have been the higher availability of resources as low-cost labour and manufacturing capacity, increasingly compelling companies to move towards sourcing parts and components globally. Key driving forces have also been shortening product life cycle, placing a premium on speed to market, as well as rapidly declining costs of transportation and communications, atomising all resources for manufacturing. Within simple settings of collocated operations, the challenge of managing can still be achieved by conventional planning and decision mechanisms. For networks, control becomes much more complicated, as the involved units and their respective roles are not stable but evolve dynamically. However, these properties activate to incorporate changing external partners as well as varying capabilities and knowledge, enormously increase a company’s adaptabilities and strongly amplify differentiations and uniqueness.

On the long path to such enterprise networks a number of more advanced manufacturing principles and management approaches have been presented. The most important manufacturing philosophies that have been introduced may be considered Lean Manufacturing (Ohno 1988), Agile Manufacturing (Kidd 1994), Holonic Manufacturing (VanBrussels et al. 1998), and Fractal Factory (Kühnle 1995). More comprehensive interorganisational structures are supply networks, Virtual Enterprises, extended enterprises (E2), professional virtual communities (PVC) and collaborative network organisations (CNO) (Camarinha–Matos and Afsamanesh 2005). Emphasis has been placed on the renewal of companies’ culture, organisation and management, making use of growing ICT possibilities, where human creativity and improvisation are given higher decision-making power to better meet the new objectives. However, the emerging enterprise nets are more than just the amalgamation of a number of entities. Such networks consist of numerous independent and geographically dispersed entities as well as subnets of complex behaviour; therefore, the manufacturing principles, addressed above, face difficulties in implementation amid this complexity.

As a consequence, the last few years have witnessed a resurgence of interest in the nature of networks and complexity theory as possible sources of solutions. Complex adaptive systems theory is increasingly concerned with the understanding of intrinsic interactions and non-linear dynamics of distributed systems with many entities. It has proven to be a highly suitable framework that accounts for the complex interactions among the various entities of manufacturing networks, which give rise to complex behaviour that cannot be attributed to a single unit as it is a collective effect. Analysing Distributed Manufacturing networks through the lens of complex adaptive systems is helpful due to the fact that contemporary operation set-ups rather resemble dynamic, complex, interdependent and globally distributed webs than the static well-determined systems which have traditionally dominated our thinking. Furthermore, understanding Distributed Manufacturing as complex networks reveals new potentials for improving decision-making for the management of processes and value chains. Network principles in manufacturing replacing hierarchical management undoubtedly give competitive advantages, as the “certainties” of command and control approaches evidently no longer seem to hold true. A company has to see itself primarily as a unit in a network, getting value out of this loosely coupled enterprise (Norri and Lee 2006) by focusing on distinct process segments and by attracting the maximum network resources towards its visions and objectives.

Entirely new devices from ICT have enhanced adaptabilities not only at the information-integration level but also at the resource and process levels. Working environments may consist of hybrid spaces composed of virtual and actual features. Moreover ICT devices applied for concurrent and intensive participation – plug & participate (P&P) – are available wordlwide. Mobile and wireless networks, seamless interconnection and the wide spread use of powerful systems applications have brought about advanced types of individual, organisational and management use. Emerging collaborative working environments (CWEs), making extensive use of telepresence, offer ubiquitous computing infrastructure composed of resources providing a new blend of activity-oriented, context-aware flexible software services supporting patterns of human interactions and human-machine interactions. The efficient use of these devices requires continuous re-visiting of applied functionalities as well as product bundles as the ability to provide immersion facilities by powerful 3D (ambient) virtual and augmented reality systems is considered one of the most important manufacturing skills. A hybrid virtual real environment is another advanced infrastructure for creative collaborative work. Concurrent visualisations of products and processes allow effective collaborative analyses of new ideas, experiments to test different ideas, collaborative problem solving and instant distribution of tasks.

Concurrent enterprising is the term used to describe these contexts including such interactions as "interactive computer networks will link workers for all aspects of the business". It will not only make use of significant new technologies in communication and processes by which products are produced, but also of new organisational set-ups. This is particularly true for Distributed Manufacturing networks. Distributed Manufacturing builds on the fundamental characteristics of complex systems and decision-making in network design and process engineering (Kühnle 2007). The contributions in this book implicitly exploit five modes for network structuring, linking and improving:

1. The parallelism mode exploits the option of achieving shorter execution times by performing multiple stages in parallel or with some overlap; it includes event-driven or real-time updates and evaluations of models (Chaps. 2 and 4).
2. The emergence mode expresses the proper composition of the overall network and value chain from smaller segments that are concurrently controlled and managed quasi-independently (Chaps. 7 and 8).
3. The behaviour mode defines the dynamics of the synthesised networks and the dependencies on event-driven data and logics as well as interactions of operations (Chaps. 2 and 8).
4. The iteration mode emphasises the fact that there is an inherent, evolving nature to structuring. Iteration results in changes that must propagate through the structure's stages, requiring continuous process rework (Chap. 6).
5. The encapsulation mode enables one to build networks and processes by combining elements for creating new entities or for atomising entities to obtain elements respectively as well as to bundle or decompose data networks with the methods that operate on that data (Chaps. 3 and 5).

Oriented by these five fundamental modes, this book outlines the important research results of a comprehensive project. It highlights key parts of the project's outcomes and draws attention to the following points:

- Distributed product development

- Co-evolution
- Software technology paradigms
- Immersion and wearable computing
- Multi-agent systems.

0.1.1 Overview

All contributions highlight specific areas in Distributed Manufacturing. The huge number of approaches and attempts to implement new ICT solutions in manufacturing networks underpin the importance of interoperability, security, linkages and interconnectedness issues in distributed industry structures. Another topic in Distributed Manufacturing that is arousing much interest is the field of collaboration in all its patterns.

0.1.1.1 Distributed Product Development

Engineering quite early on supplemented the macro level of organisational design and coordination with specific product development and project management methods. These methods orchestrate specific activities towards common goals. However, product development methods have only to a small extend been extended by methods addressing their specific requirements such as distributed, but interdependent and collaborative, planning (Gassmann and von Zedtwitz 2003), systematically implementing parallelism, iteration and emergence. Again, new communication technologies are the key enablers and the main drivers of new solutions. However, companies and researchers have very limited understanding on how to organise and manage product development in this context successfully as the underlying principles seem to be ill understood, leading to increased risks. Solution approaches, methods and routines of distributed collaboration and their embedding in Distributed Manufacturing networks to support the development of bodies of knowledge must be discussed. Organising and managing concurrent product development utilising new communication technologies must be implemented and sustained as interdependent and dynamic structures of organisational set-ups, management methods and technology applications.

0.1.1.2 Co-evolution

Co-evolution is a dynamic process that accelerates the path from concepts to concurrent capabilities development, analysis and evolution across enterprises. It is the prototypical illustration of parallelism, behaviour, emergence and concurrent (multistream) iteration and interaction. Examples of multistream interaction are ant swarms, the Internet, traffic patterns and economic markets. In co-evolution, the results of an evolutionary process define their own fitness functions, because they are selected for their success in competing with each other. The principally cited example is the evolution of a predator species alongside the evolution of a prey species. In a broad sense, biological co-evolution is "the change of a biological object triggered by the change of a related object". In Distributed Manufacturing, co-evolution may therefore be envisioned as a multidimensional exploration of units' 'effectiveness landscapes' to identify and achieve (co-evolve) combinations of characteristics with acceptable levels of utility.

For Distributed Manufacturing this co-evolution metaphor includes rapid development techniques, proof-of-concept and learning prototypes as well as continuous review and experimentation to continually support and adjust the development process and its products. By experimentation and continuous evaluation by the user community, it also ensures that the capabilities developed in Distributed Manufacturing networks are relevant, even in a continuously changing landscape.

0.1.1.3 Software Technology Paradigms

As distributed structures in general and Distributed Manufacturing in particular are strongly influenced by trends in ICT as well as in software engineering, these trends have a strong impact on concepts and solutions aimed at supporting upcoming set-ups in manufacturing. Systems and software developers address these set-ups as challenges for their packages' interoperability, interfaces and functionalities. Mirroring the challenges to the most recent technology platforms generates substantial insights concerning how planning and control software should be built to fully integrate all levels of the automation pyramid.

0.1.1.4 Immersion and Wearable Computing

Another key concern of Distributed Manufacturing is to facilitate and encourage interactions between individuals from different entities. One way to connect people and concepts together, and make the collaboration and the exchange of information more fluent and easier to handle, is through the use of a virtual space of networked individuals, based on virtual and augmented reality (VAR) or mixed augmented reality (MAR), forming ad-hoc groups on the basis of common tasks and objectives. Important solutions for linking collaborative resources together or for engaging new resources within the structures of Distributed Manufacturing are possible within such VAR environments. Taking a look into collaborative environments and their benefits by providing much faster and broader access to existing knowledge and people know-how gives ideas about the emerging potential of

ICT for distributed value creation. These technologies provide enhanced environments for supporting individual and group collaboration work. The framework is unwrapped and options are analysed covering interaction and collaboration issues. Expected benefits include providing more supportive, appealing and communicative applications by applying various design theories and practices to distributed manufacturing networks contexts. A novel approach classifying and eliminating "collaborative distances" is proposed as a solid base for company-specific implementation projects. Mobile VAR may be envisioned as a concrete example what organisational and educational implications are to be faced in the Distributed Manufacturing contexts.

0.1.1.5 Multi-Agent Systems

Multi-agent systems (MAS) with options as well as with standard features are an appropriate approach for modelling complex networks which can bring in a platform of important functions for generating the decisions of network structures (Wooldrige 2002). When adopting an agent-oriented view, it soon becomes apparent that most problems require or involve multiple agents to be able to represent the decentralised nature of the problem, the multiple loci of control, the multiple perspectives, the competing interests or the distribution of control responsibilities. An agent is an autonomously operating software program which contains a representation of its environment and has its own objectives.

The need for both flexibility and aggregate planning in distributed manufacturing environments, resulting from today's market demands, requires the development of control methods that are able to combine the advantages of both heterarchical (flexibility) and hierarchical (aggregate planning) features. Keywords are self-configuration, co-operation, modification and high flexibility. Different methods have recently been developed that aim at achieving these goals, which are often indicated as evolution-based control concepts.

Moreover, the agents will need to interact with one another to achieve their individual objectives, manage the dependencies that ensue from being situated in a common environment, and provide an optimal solution for the overall control problem. These interactions can vary from simple semantic interoperation (information passing), to traditional client–server-type interactions, to rich social interactions (the ability to cooperate, coordinate and negotiate about a course of action). A multi-agent architecture allows decisions to be taken in a decentralised way. In production control systems, intelligent agents are introduced to represent the physical resources, parts and jobs in a manufacturing system. MASs in Distributed Manufacturing networks as well as in process control strongly rely upon the parallel update of data and iteration for achieving efficient operation plans. This parallelism assumption in MASs is so evident that it is not even mentioned. For distributed computing and the operating systems introduced there, encapsulation might be self evident for manufacturing systems in general and distributed manufacturing in particular, it is not. It is extremely important that all concerned units' behaviours and restrictions be updated; the redundancy in data is not an issue since the computing power, even of small devices, is sufficient in all cases. Updates are executed event-driven, which is an important and inevitable step overcome the time slot planning procedures of contemporary Enterprise Resource Planning (ERP) and Manufacturing Execution System (MES) systems, so forward planning and simulation turn out to be identical.

0.1.2 Impacts on Manufacturing Industry

In Distributed Manufacturing, control and decision-making decentralise, and the „collaborative character" of the network is emphasised. As management becomes more complex a new generation of instruments is called for.

This book gives descriptions of important paradigms and their unfolding within Concurrent Enterprising, Distributed Manufacturing, and industrial control. It covers important aspects from the points of view of marketing, engineering, technology, environmental protection, etc. and synthesises technical to organisational solutions.

Mainly the MAS approach has been successfully implemented in manufacturing networks in MESs. The decision-making problems there can be clearly structured, and therefore the MES (including plug and produce, etc.) may be fully decentralised and automated. For parallelism of operations in manufacturing, the research on industrial networks must aim deeper into the dynamic forms of communication and coordination. Resources and capabilities inside and outside of an industrial organisation are vast resources for innovative ideas and knowledge. In order to take advantage of these resources, Distributed Manufacturing also focuses on linking industrial processes throughout the distributed enterprise by exploiting the collection of all available knowledge and innovative ideas. Manufacturing enterprise networks emerge where all aspects of manufacturing are networked, so informed decisions concerning a specific activity can be made, based on knowledge and experience, concerning the total network. Elaborate concepts resulting from Distributed Manufacturing might revolutionise the ways people interact at all levels of organisations.

One major conclusion of the contributions is that the prevailing trends in the structuring and engineering of Distributed Manufacturing will come from information sciences and its applications to mechatronic units. The permanent need to restructure and relink obviously brings about successful practical solutions that strongly involve the principles of complexity. However, the focus of such control efforts is still on the design of units' configurations as well as the activation of process sequences, where standards may provide for open-loop controls. For efficient adaptations as well as for continuous improvements in manufacturing networks, important control loops should be closed. The instruments suggested may be seen as possible next steps on the way to the closed-loop control of Distributed Manufacturing networks.

As a substantial contribution to excellence in production, Distributed Manufacturing obviously builds up competitive power for everyday operations. Instead of trying to ignore or even eliminate structural behaviours of networks, Dispersed Manufacturing successfully exploits these network properties. In unwrapping the Distributed Manufacturing paradigm, solutions have been developed for the generation of control decisions and evaluations, e.g. on the task levels by software agent technology. One resulting generic concept for distributed control and order execution on the MES level, based on the described framework, is the PABADIS architecture.

Exploiting ideas of the application of mechatronic units and its control, as is considered in the PABADIS approaches, has actually leaded manufacturing and control system companies to develop technologies for efficient manufacturing system engineering with an emphasis on open industry standards (AutomationML 2007).

Comparable efforts are going on for further ERP developments too. As the distributed ICT world is about to be established on company levels as well, important vendors of ERP systems are intensively working on corresponding software packages, also making extensive use of these frameworks and standards. With these achievements complexity approaches have, contrary to some harsh critics of managers, successfully generated important additional insights which are novel and instructive. Moreover, many thoughtful managers in Distributed Manufacturing networks will appreciate them as an enrichment of their production worldview in turbulent times.

References

AutomationML. 2007. Available from: http://www.automationml.org/.

Camarinha-Matos, L. M. and H. Afsarmanesh. 2005. Collaborative Networks: a new scientific discipline. *Journal of Intelligent Manufacturing,* 16: 439-452.

Gassmann, O. and M. von Zedtwitz. 2003. Trends and determinants of managing virtual R&D teams. *R & D Management,* 33 (3): 243-262.

Kidd, P. T. 1994. *Agile Manufacturing: Forgoing New Frontiers.* Addison-Wesley.

Kühnle, H. 1995. L'entreprise fractale. *In:* Braesch, C. and A. Haurat, eds. *La modélisation systémique en entreprise.* Paris: Pôle productique Rhône-Alpes.

Kühnle, H. 2007. A system of models contribution to production network (PN) theory. *Journal of Intelligent Manufacturing,* 18 (5): 543-551.

Norri, H. and W. B. Lee. 2006. Dispersed network manufacturing: Adapting SMEs to compete on the global scale. *Journal of Manufacturing Technology Management,* 17 (8):1022-1041.

Ohno, T. 1988. *Toyota Production System: Beyond Large-Scale Production.* Productivity Press.

VanBrussels, H., J. Wyns, P. Valckenaers, L. Bongaerts and P. Peeters. 1998. Reference architecture for holonic manufacturing systems – PROSA. *Computers in Industry*, 37.

Wooldridge, M. 2002. *Introduction to MultiAgent Systems.* Chichester: John Wiley & Sons.

1 The Concurrent Product Development Process

Antonio Hidalgo[1], **Fernando Aldana**[2], **Darius Singh**[3]

[1] Dept. Business Administration, School of Industrial Engineering, Universidad
Politécnica de Madrid, C/José Gutiérrez Abascal, 2, 28006 Madrid (Spain),
e-mail: ahidalgo@etsii.upm.es

[2] Dept. Business Administration, School of Industrial Engineering, Universidad
Politécnica de Madrid., C/José Gutiérrez Abascal, 2, 28006 Madrid (Spain),
e-mail: f.aldana@upm.es

[3] Centre for Advanced Manufacturing Technology, AUT University
24 St. Paul, Auckland 1142 (New Zealand), e-mail: darius.singh@aut.ac.nz

Abstract Trade barriers have largely been removed through free trade agreements and economic blocs, not only in Europe but also worldwide. Manufacturers are finding that their distinguishable markets are swiftly evolving into a single, global marketplace. This can create new business opportunities but, on the other hand, the domestic markets are no longer protected as more and more companies are competing internationally. To compete in this new scenario companies are increasingly focused on covering all aspects of the product development process as their core business and decentralising other tasks using business units which operate more or less independently. The product development process in extended enterprises is increasingly organised in networks of suppliers, manufacturers and users, evolving towards service companies.

1.1 Principles of Product Development

The old proverb the customer is king is even more true in the current environment where, instead of buying whatever the manufacturers decide to produce, customers require products tailored to their particular needs and tastes. European enterprises are expanding globally and now have different worldwide sites for developing products, manufacturing and marketing. Growing global competition as well as new regulations and laws concerning environmental issues, quality issues, etc., have led to dramatic structural and technological changes within industry. In this context, extended enterprises are becoming an extremely important organisational concept.

An understanding of the attributes of successful new products in extended enterprises is central to obtaining an effective new product management. It provides insights for executing new product projects (i.e. are certain best practices strongly linked to success?) and yields clues to new product selection (what is the profile of a winner?). These success factors can be approximately divided into two groups (Cooper 1993, 2001):

- Process attributes – those factors that capture the nature of the new product process and how the project is undertaken. These are often controllable factors (i.e. doing projects right).
- Selection attributes – those factors that describe the new product project and its situation. These tend to be outside of the control of the project leader and team but are useful in project selection (i.e. doing the right projects).

The keys to new product success (critical factors of process attributes) are based on numerous research studies into why new products succeed, why they fail, and comparisons of winners and losers. The most revealing of these studies have been the large-sample, quantitative studies of successful versus unsuccessful new products. They began with Project SAPPHO in the early 1970s (Rothwell et al. 1974), followed by the NewProd series of studies, the Stanford Innovation Project, and more recent studies (Maidique and Zirger 1984; Cooper 1995; Montoya-Weiss and O´Driscoll 2000; Daneels and Kleinschmidt 2001; Calantone et al. 2003; Dröge et al. 2008). To summarise, a winning product is superior to competing products in terms of meeting users' needs, offers unique features not available from competitive products, solves a problem the customer has with a competitive product, provides excellent relative product quality, reduces the customer's total costs (high value in use) and boasts excellent price/performance characteristics.

Not only must the product be superior, but it must also be launched, marketed and supported in a proficient manner. These elements include brand name or company reputation, superior marketing communications (advertising and promotion), a good sales force or distribution channel, superior technical support and technical service, or simply product availability. The limited evidence available, however, suggests that the impact of non-product advantage pales in comparison to the impact of product advantage (Crawford 1992).

Product development is very much a team effort. Do a post-mortem on any bungled new product project and invariably you will find each functional area doing its own piece of the project, with very little communication between players and functions, and no real commitment of players to the project. This derives from a typical pattern of inadequate human resources devoted to the project with players having numerous other functional tasks going on at the same time. Product development in extended enterprises must be run as a multidisciplinary endeavour and must have a good organisational design which means (Cooper et al. 2002):

- The project is organised as a cross-functional team with members from research and development (R&D), product engineering, manufacturing operations, marketing and sales, operations, and so on.
- The team is dedicated and focused (i.e. devotes a large percentage of its time to this project, as opposed to being spread over many projects).
- The team members are in constant contact with one another via frequent but short meetings, interactions and project updates.
- There is a strong project leader who leads and drives the project.

A second organisational success ingredient is climate and culture. A positive climate is one that supports and encourages intrapreneurs and risk-taking behaviour, where new product successes are rewarded and recognised (and failures not punished), where team efforts are recognised, rather than individuals, and where resources and time are made available for creative people to work on their own unofficial projects. Idea submission schemes (where employees are encouraged to submit new product ideas) and open project review meetings (where the entire project team participates) are other facets of a positive climate.

Finally, there are four factors that describe the new product project and its setting – critical factors of selection attributes (Cooper 2005). Unlike the factors above, which are process related, those below are less controllable by the project team and tend to be more useful as project selection criteria. These factors are market potential, competitive situation, product life cycle, and synergy or leveraging core competencies.

1.2 Methodology of New Product Development in Extended Enterprises

The product development literature has given increasing attention to firm-level considerations of an organisation's new product efforts. In extended enterprises the actors of new product development are members of a variety of firms, with different cultures, different systems, speaking different languages and using differently named concepts. At this level, new product development may be defined as the aggregate pattern of product introductions that emerges from an organisation over time. In that perspective the methodology of new product introductions in extended enterprises is based on five specific approaches: product strategy, advanced product planning, product cost management, market analysis and process coordination.

1.2.1 Product Strategy

Product strategy provides the focus for the extended enterprise's new product efforts and is manifest in its pattern of sequential product introductions. The purpose of an extended enterprise's product strategy is to link its products to its overall objectives and to assist in the search for new products. To characterise a new product, usually three key dimensions are suggested that hold significant strategic implications for firms: newness of embodied technology, newness of market applications and innovativeness in the market (Meyer and Lehnerd 1997; Wonglimpiyarat 2004; Lawley 2007). Nowadays the new trend in strategies for innovation is linked to the enrichment of the product itself adding and embedding in it some services or knowledge that make it more significant and attractive for the customer. An example comes from the automotive sector where the car is becoming a knowledge-intensive product offering many services to the customer (i.e. satellite-based navigation system, more passenger entertainment, etc).

Basically, four generic design functions of a product, each emphasising a dominant attribute, can be identified: technology-centred, marketing-centred, image-centred and user-centred. The precise combination of these will vary according to product, but all should be closely integrated. Divisions often exist between technology-centred aspects of design products and manufacturing, marketing demands and ideas about styling. All too often, user needs are subordinated to available technology or attempts to create a superficial image for purposes of differentiation. This push to production and marketing has obvious limitations; one research programme revealed that, in over half the projects studied, products needed redesigning by the time final prototype testing took place since customer requirements were found not to match those originally planned after concept evaluation (Bonnet 1986; Khan 2004).

In terms of its essential contribution to product development in extended enterprises, three major functions of industrial design can be identified:

- Giving a product concept tangibility, which is a vital stage in translating from an abstract idea to an actual form as perceived by users, thus enabling decisions on the feasibility of ideas to be more firmly grounded.
- The form of a design has important implications for manufacturing feasibility and therefore cost. Identifying any incompatibilities or the need for new equipment or supplemental processing (machining, polishing, coating, rework, etc.) at an early stage can be a vital element in costing and decision-making on a project.

- The reality of a design and its value as perceived by users is the ultimate determinant of market success and should be the core focus of any development process.

Product models and product families, long and widely accepted in practice as basic units of analysis, have recently begun to receive attention. Model distinctions and family relationships are fairly easily established for sophisticated technical products, particularly assembled systems (Rothwell and Gardiner 1988; Burgelman et al. 2003). Having established product models and families as our units of analysis, it is appropriate to analyse each in terms of variety and rate of change in order to arrive at an integrated picture of an extended enterprise's pattern of product competition. In Fig. 1.1, which presents the sales history for a family of related models, each model has its own model life cycle. An extended enterprise's product family life cycle is the aggregation of these model cycles. The model variety available to customers at *t*, for example, is 12. Thus, we can estimate model variety only if the boundary of the product family is known, which reinforces the need to attend to industry criteria for family membership.

Model lifetimes can be used to generate a rough estimate of the rate of serial model change, that is, the rate at which a model is being replaced. This can be approximated by the reciprocal of the model's lifetime. Models with 2-year lifetimes, for example, will be replaced at a rate of 0.5 models per year if existing model variety is to be preserved.

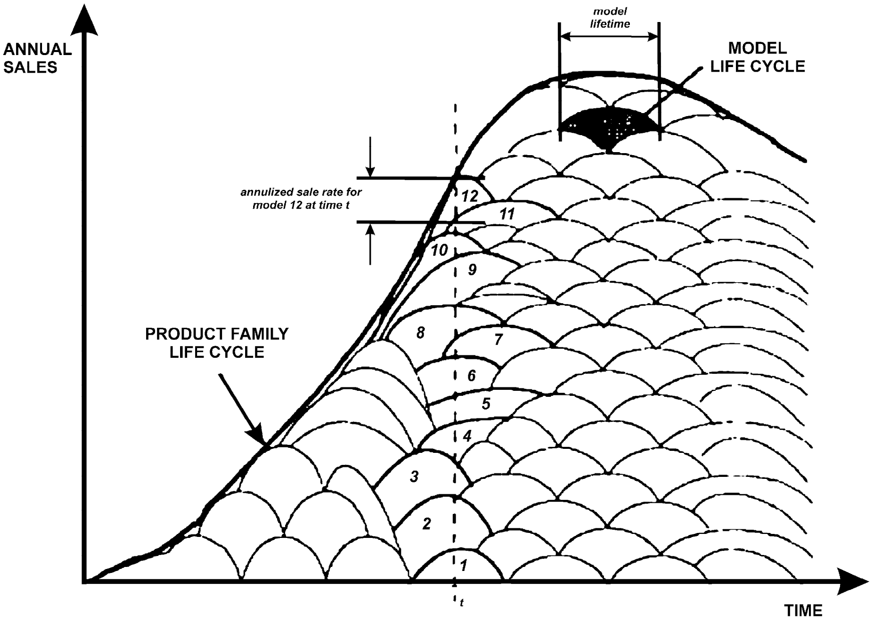

ANNUAL SALES
model lifetime
MODEL LIFE CYCLE
annulized sale rate for model 12 at time t
12
11
10
9
8
7
6
5
4
3
2
1
PRODUCT FAMILY LIFE CYCLE
t
TIME

Fig. 1.1 Product family life cycle

Beyond the competitive patterns within individual product families, extended enterprises have a larger opportunity to develop multiple product families simultaneously. These can be plotted in terms of variety and rate of change. For simple product families, the product model and family life cycles are essentially the same. For most other products, patterns of family evolution are independent of, and can be quite different from, patterns of model evolution. Product families that replace one another in rapid serial fashion exhibit a generational pattern of evolution. One might find in successive generations of a product family, for example, any or all of the four patterns of product model evolution. The logic advanced thus far suggests three criteria for generational product evolution:

- Powerful and persistent market demand for continuous improvement (without which major change will not occur).
- More than one technological way to satisfy market need (given only one technological approach, the generational model evolution will dominate).
- Strong market resistance to the simultaneous existence of more than one product family (without such resistance, turbulent product family competition will result).

1.2.2 Advanced Product Planning

Accelerated product development is particularly important to extended enterprises that are committed to pioneering. Pioneers are the first entrants into a market with a new product or a new generation of product. A common characteristic is that they have a high tolerance for risk. As the first entrants, they have a monopoly position until a rival emerges. This position may lead to a leadership reputation and enable premium pricing and may lead to the pioneering product's establishing itself as standard. For products with high switching costs, pioneers can secure their position by creating a large installed base before significant rivals emerge. The experience gained from the pioneer's lead may also translate into a cost advantage or a sustainable lead in technology development. Finally, the first entrant often has first access to actual experiential customer feedback. Having the ability to rapidly translate this new market knowledge into the next generation of product means that the pioneer is more likely to continue to satisfy market needs.

There are seven characteristics of businesses that have achieved relatively short product development cycles. These characteristics have been identified in recent studies (Zirger and Hartley 1996): a market-oriented product definition process, dedicated and cross-functional project teams, predevelopment planning, overlapping development phases, focusing on core competencies, incremental product development based on reuse and leverage, and accessible organisational memory. Managing the cycle time for the development of new products should not be viewed in isolation from the larger issues and concerns confronting the business. The long-term success of an extended enterprise depends on a stream of new products – some replacing older ones, others pioneering new markets, and all satisfying customer needs. It is this stream of new products, exploiting advances in both product technologies and technologies used to manufacture, distribute, and provide support, that provides the fuel for corporate growth and renewal. But the most important indicator of success in terms of cycle time is not the schedule to slip rate of any single product but the ability of the firm to introduce a stream of exciting, value-rich products over time.

Successful product-developing extended enterprises must create robust product families. Product families do not have to emerge one product at a time. In fact, they can be planned so that a number of derivative products can be efficiently created from the foundation of a common core technology. We call this foundation of core technology the product platform, which is a set of subsystems and interfaces that form a common architecture from which a stream of derivative products can be efficiently developed and produced. A platform approach to product development dramatically reduces manufacturing costs and provides significant economies in the procurement of components and materials because so many of these are shared between individual products.

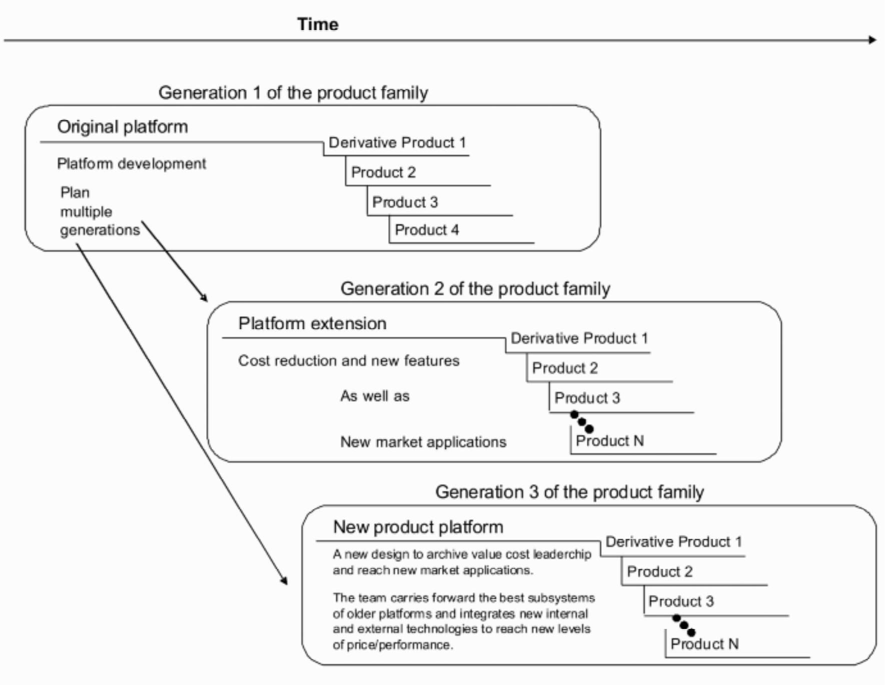

Time
Generation 1 of the product family
Original platform
Platform development
Plan multiple generations
Derivative Product 1
Product 2
Product 3
Product 4
Generation 2 of the product family
Platform extension
Cost reduction and new features
As well as
New market applications
Derivative Product 1
Product 2
Product 3
Product N
Generation 3 of the product family
New product platform
A new design to archive value cost leaderchip and reach new market applications.
The team carries forward the best subsystems of older platforms and integrates new internal and external technologies to reach new levels of price/performance.
Derivative Product 1
Product 2
Product 3
Product N

Fig. 1.2 Product family evolution, platform renewal and new product creation

Figure 1.2 represents a single product family starting with the initial development of a product platform, followed by successive major enhancements to the core product and process technology of that platform, with derivative product developments within each generation. Successful extended enterprises continuously renew their platform architectures and their manufacturing processes by integrating advances in core product and process technologies.

Add to this figure greater depth and you end up with the framework shown in Fig. 1.3. At the top of the figure are the market applications of the extended enterprise's technology. The market for the product family is defined in a traditional way through a matrix of market segments that identifies particular user groups and product price or performance characteristics. The market applications of a product family take the form of derivative products based on product platforms. Most corporations tend to view their market segments in isolation from one another. Simply placing these segments on one page may allow management to then consider how product technology and manufacturing processes can be shared or made common across product lines serving different market segments.

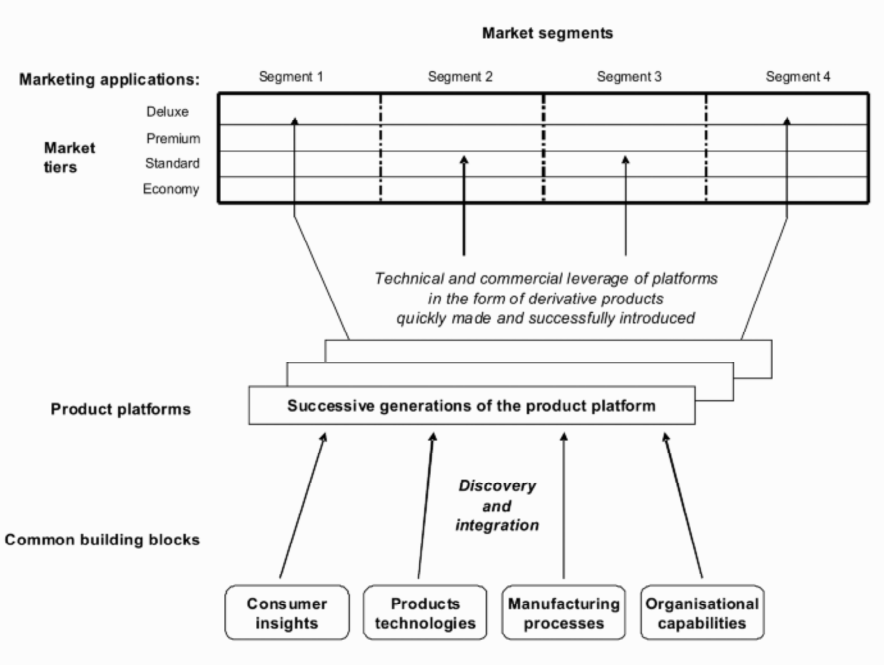

Market segments
Marketing applications:
Segment 1
Segment 2
Segment 3
Segment 4
Market tiers
Deluxe
Premium
Standard
Economy
Technical and commercial leverage of platforms in the form of derivative products quickly made and successfully introduced
Product platforms
Successive generations of the product platform
Discovery and integration
Common building blocks
Consumer insights
Products technologies
Manufacturing processes
Organisational capabilities

Fig. 1.3 An integrative model of product and process innovation

In the middle tier are the extended enterprise's product platforms as defined earlier. Every company must determine precisely the structure of the product platforms suitable for its business, e.g. those subsystems and interfaces that are the essence of the stream of products or services it provides. Product platforms capable of accommodating new component technologies and variations make it possible for firms to create derivative products at incremental cost relative to initial investments in the platform itself. This is possible because the fundamental subsystems and interfaces of each new derivative are carried forward. Since the costs associated with the carried forward elements are essentially sunk costs, only the incremental costs of creating variations to them accrue to the derivatives. Typically, these incremental costs are a small fraction of the cost of developing the original product platform, leading to what may be called platform leverage. Product platforms can also improve development cycle times of derivative products by facilitating a more streamlined development process and more frequent model changes (Clark and Fujimoto 1991; Ulrich and Eppinger 2007).

At the bottom tier of Fig. 1.3 lies the heart of all product development activity: those core technologies and competencies in product and process arenas that are brought together to form a current generation product platform. We think of technology as the implementation of knowledge with the potential to be incorporated into a product. Product technology takes many forms: chemistries, programming languages and algorithms, hardware or logic design, and so forth. The building blocks are the essential components within the subsystems of product platforms. Product technologies also include subsystem interfaces, their proprietary connections or those based on regulatory imposed standards.

1.2.3 Product Cost Management

Product cost management is generally approached from a strictly internal point of view. From this perspective, the management of costs is typically limited in scope to materials, labour, production, research, distribution and other such costs of producing and distributing a product (Shank and Govindarajan 1988; Burgelman et al. 2003; Tucker 2008). While the management of these internal costs is certainly critical to achieving a desired level of profitability, it is also important to consider the total cost of a product to the customer. Customer costs include, in addition to the purchase price, the cost of acquiring and installing a product as well as costs related to using, maintaining and disposing of a product. Consideration of these post-purchase customer costs, along with internal product costs, can provide a more complete approach to product cost management.

The cost management process includes three inputs: technology, production and systems. Each has a meaningful effect on the three major areas of product costs – customer costs, variable costs and fixed costs. Effective management of these inputs and areas of cost has the potential to lower customer costs, which can translate into increased customer demand, and indirectly lower a business's variable and fixed costs, which, in turn, contributes to greater profitability. The first step in the product cost management process is to understand how technology, production and system inputs affect the three major areas of product cost. To achieve a meaningful reduction in any of these areas requires management of one or more of these inputs.

Technology is the first of the three product cost management inputs. Product, process and information technologies each have the potential to lower product costs and customer non-price costs. Gains in any of these broad areas of technology can reduce variable costs with lower unit manufacturing, distribution, and inventory costs. Technology inputs can also be used to lower fixed costs associated with product development time, marketing, and product administration. These technology inputs also have the potential to reduce customer non-price costs through product designs/redesigns that lower the customer's costs of acquiring, installing, using, maintaining or disposing of a product.

There are three production inputs that have the potential to lower product costs (Best 2000). The first of these is economies of scale. A production capacity of 2x will achieve a lower unit cost than a production capacity of 1x. However, to achieve a lower unit cost, a business has to operate near full production capacity. A business with a 2x production capacity that operates appreciably below that capacity could actually have a higher per-unit cost than a competitor with 1x capacity that operates near full capacity. Product line scope is another production input into product cost management. As more products using similar product designs or processes are added to a business's product line, the unit manufacturing cost of each product is likely to be reduced. This is due to the purchase and manufacture of common components or subassemblies in larger quantities as well as providing a higher utilisation of capital equipment. Thus, product line scope has the potential to lower unit costs, which impact prices (a customer cost) and margins (a profitability component).

Learning through production experience also lowers the unit cost of a product. As a business learns more from the experience of producing a product, there is a tendency for the unit cost of a product to decrease exponentially. For example, most microchip production plants are on learning curves near 70%. This means that every time the cumulative production experience for a specific product doubles, the unit cost decreases by 30%. Thus, production experience (learning) also lowers unit cost, which could lower customer costs (through lower prices) or improve profitability (with higher margins).

System inputs can be derived from three major areas – manufacturing, purchasing, and distribution. Systems such as just-in-time inventory management, materials resource planning, CAD/CAM design systems, outsourcing, and shared production each offer product cost management opportunities that could lower customer costs and business costs. For example, a more efficient inventory system could lower customer acquisition costs since customers would be able to reduce inventories of a purchased product. Likewise, a more efficient distribution system could lower transportation costs, which could lower customer acquisition costs as well as a business's variable transaction costs. CAD/CAM design systems and outsourcing have also been shown to shorten product development time and the fixed cost of product development, which could benefit customers with earlier access to better or lower-cost products as well as benefit a business with lower fixed operating expenses.

1.2.4 Market Analysis

The combination of unique product benefits at a specific price creates a particular product position relative to competing products. However, being distinct and different in price and product benefits is not sufficient for success. To be successful, a business must create, communicate, and deliver a product position and a value proposition that is appealing to target customers and differentially superior to competing alternatives. The product positioning process consists of the following steps:

- Inputs: This is where the product positioning process has to start. A technologically superior product that does not meet customer needs will result in limited customer demand and economic failure. To be successful, product positioning must deliver an attractive combination of customer benefits at an acceptable cost or an attractive cost with an acceptable level of customer benefits. Customer benefits can include hard benefits, such as operating performance, reliability, durability and value-added features, and soft benefits, such as local buying and customer service, accompanying software, on-line technical support, product warranty and manufacturer reputation.
- Product positioning: A product position built around superior product benefits can often be attractive to target customers even at a price premium. Conversely, a product position built around a lower price can attract customers when product benefits match those of competing alternatives. A value proposition is a short statement designed to communicate a business's product position to target customers; it highlights key costs, benefits and how the customer should derive increased value from their product (Band 1995).
- Outputs: To attract customers, a product position must deliver a superior value relative to competing alternatives. This means that the overall benefits derived from the product must exceed the total cost of purchase. A potentially attractive product position will fail if target customers are unaware of a product's value proposition or cannot easily and fully utilise the product. Achieving market penetration requires an attractive product position and marketing effort (Best 2000).

1.2.5 Process Coordination

Effective product development requires the integration of specialized capabilities. Integrating is difficult in most circumstances but is particularly challenging in extended enterprises with strong functional groups, extensive specialisation, large numbers of people, and multiple, ongoing operating pressures. Before starting the process of developing a new product, it is essential to establish mechanisms to allow effective coordination between project team members. This is unlikely to happen unless we provide the basis for communication between workers so that they get motivated to work as a team. Contrary to what we expect, if there is a lack of communication, then people will reach their own conclusions and work independently.

One of the most powerful resources for enabling rapid development is the use of cross-functional teams that include representatives of all the disciplines involved in the innovation and that have the necessary autonomy to carry out the project. Teams of this kind are not formed simply by grouping people together; successful practice involves extensive investments in team building, providing them with the necessary training to solve problems, to manage conflict, to interact with other parts of the organisation and with outside stakeholders. Empowering teams and providing them with autonomy and resources will only work if they have a clear sense of direction. One important way of providing this is to involve them in the process of vision-building, evolving the product concept in the context of a clear understanding of the underlying business drivers and competitive realities.

Closely linked with the concept of teamwork is the need to get a good match between the demands of development and the operating structure that enables it. Traditionally the choices were between functional teams, cross-functional project teams or some form of matrix between the two. However, in recent studies two models have emerged that appear correlated with success in extended enterprises:

- Heavyweight product manager structure – essentially a matrix structure led by a product (project) manager with extensive influence over the functional personnel involved but also in strategic directions of the contributing areas critical to the project. By its nature this structure implies considerable organisational authority.
- Project execution teams – a full-time project team where functional staff is seconded from their roles and areas to work fully on the project, under project leader direction.

Associated with these different structures are different roles for team members and particularly for project managers. For example, the heavyweight project manager has to play several different roles, which include extensive interpreting and communication between functions and players. Rather than being either neutral or a facilitator with regard to problem solving and conflict resolution, leaders see themselves as championing the basic concept around which the platform product is being shaped. They make sure that those who work on subtasks of the project understand the concept and they play a central role in ensuring the system integrity of the final product. Some of the ways in which the heavyweight project manager achieves project results are highlighted by the five following roles: direct market interpreter, multilingual translator, direct engineering manager, programme manager in motion, and concept infuser.

1.3 The ICT Tools and New Product Development

Actually, the fact is that in our increasingly saturated markets the list of requirements attached to a successful new product is continuously growing, as is the list of the potential stakeholders. Superior products, lower costs and shorter development time are a cocktail that has prompted a revolution in the organisation and technologies of development.

To cope with this situation new paradigms have appeared and become almost the standard. The new paradigms, while meant to boost the chances of success of a product addressing all problems at the earliest possible stages, have greatly increased the complexity of the process. The drive of the extended enterprise to concentrate on core competencies has added a further complicating dimension. The new paradigms are based on the following methodological and organisational concepts:

- Design by platform, whereby a whole family of products is designed at the same time. This is meant to protect the brand image and to minimise manufacturing costs;
- Modular and concurrent design to minimise the time and cost of product development;
- Reengineering of product design process, where the process is modelled and optimised using modelling tools;
- Knowledge-based design, whereby experience and competencies, beyond data, are made available and leveraged to improve effectiveness and efficiency;
- Simulation applied to all envisaged processes.

For all the above factors, new product development has become extremely complex, requiring easy and seamless access to extensive knowledge, together with high control and coordination capability and superior simulation tools. Information and communication technologies (ICT) have provided a tremendous variety of tools with varying degrees of scope and generality. The purpose of this section is to provide a possible taxonomy, a quick overview of these tools, and to put them in the perspective of an extended enterprise. The proposed taxonomy reflects a user's point of view concerning the available tools and systems. The idea is to group the solutions according to the provided group of functions:

- Tools that support the execution of a new product task;
- Tools that support the planning and control of the process;
- Tools that support cooperation among actors;
- Tools that support the management of information.

The above classification is useful for separating the different groups of functions provided by the ICT tools, but it does not easily map into existing commercial tools. With the exception of the first group of tools, the commercial systems tend to offer combinations of these functions within the same application.

1.3.1 Execution-supporting Tools – Modelling and Simulation Tools

CAD is used as an input for a great number of CAE simulation tools that address a variety of problems in a number of technological domains, measuring the product performance or simulating the manufacturing process. CAE tools are available to evaluate fatigue, vibration, noise, lighting, etc. Well-known legacy programs such as NASTRAN®[1] are used to perform structural analysis at a very detailed level, while simpler analysis can be carried out by Abaqus®[2] or ADAMS.

CAE tools are increasingly used to simulate an increasing number of manufacturing processes, such as metal stamping (OPTRIS, SIMEX®[3], PAMSTAMP), plastic injection (I-IDEAS, C-MOLD, FABEST), casting (MAGMAsoft, ProCast™[4], FLOW-3D®[5]), welding, etc. The use of CAE tools makes the predictive assessment and virtual analysis of the manufacturability of a product well before committing resources and requesting quotes or issuing orders to suppliers. Current product development systems use CAD packages with CAE software and have been doing so for a number of years. Thus there is now a seamless data transfer and acceptability of a range of file formats, mesh types and operating systems.

The above are process simulation tools. Other tools are available on the market to simulate discrete operations performed on an assembly line. Layout and ergonomics can be studied in great detail, as can the control software of a robot (ROBCAD). The performance of a production line can be studied as a function of the components, control algorithms, and failure distribution using simulation software such as ARENA®[6], WITNESS™[7], PROMODEL, etc.

The above sets of tools, which aim at the simulation of individual processes and of the entire factory, are referred to as virtual manufacturing. Digital Mock Up (DMU) plays an increasing role in the design of complex systems requiring a strong interaction with users. DMU is capable of including users in new product development, well beyond the more traditional focus groups, and can provide the

[1] NASTRAN® is a registered trademark of NASA, Washington, DC 20546-0001, http://www.nasa.gov

[2] Abaqus® is a registered trademark of Dassault Systèmes or its subsidiaries, 10, Rue Marcel Dassault, 78140 Vélizy-Villacoublay, FRANCE, http://www.3ds.com

[3] SIMEX® is a registered trademark of SIMEX s.r.1., I – 40017 San Giovanni in Persiceto (BO), Via Newton, 31, http://www.simex.it

[4] ProCast™ is a trademark of Daktronics, 201 Daktronics Drive PO Box 5128 Brookings, S.D. 57006-5128201 Daktronics Drive PO Box 5128 Brookings, S.D. 57006-5128, http://www.daktronics.com

[5] FLOW-3D® is a registered trademark of Flow Science Inc., 683 Harkle Road, Suite A, Santa Fe, NM 87505, http://www.flow3d.com

[6] ARENA® is a registered trademark of Rockwell Automation, Inc., Boulevard du Souverain 36, 1170 Brussels, Belgium, http://www.rockwellautomation.be

[7] WITNESS™ is a trademark of Lanner Group Limited, The Oaks, Clews Road, Redditch, Worcestershire, B98 7ST, UK, http://www.lanner.com

user with a feeling for the product, both from the aesthetic point of view (geometrical DMU) and the dynamical point of view (functional DMU).

Cost modelling tools are meant to support the designer with an estimation of the costs to be incurred in the manufacturing stage. They belong to the simulation tools in the sense that they produce a key performance indicator that is among the most crucial. Different tools apply to the different development stages. Parametric and analogical cost tools apply at the offering stage, while at later stages analytical tools are more relevant, when more details are available for the product and the attached processes.

1.3.2 Process Planning and Control Tools – Business Process Modelling Tools

Modelling a new product development provides a clear picture of the process. Some of the tools support the simulation of the process, which provides the overall achievable performance. But the representation of the process is likely to suggest improvements that may not be obvious at first sight. For all these reasons business process modelling (BPM) tools are most often used for business process reengineering.

But the scope of application of BPM is much wider. In some applications a process model is used to plan a development process before release for actuation. BPM tools are also the vehicle for the integration of enterprises processes – enterprise integration –, as mentioned elsewhere, supplying the necessary orchestration infrastructure. However, such an orchestration assumes stable and well-defined processes, which hardly applies to new product development for a variety of reasons:

- Firstly, for all modelling efforts there are features in new product development that are difficult to capture, such as the continuous exchange of data and feedback typical of concurrent engineering, or the conditions that trigger the start of an activity.
- Secondly, the process is more related to a problem-solving process, where trial and error is the rule rather than the exception. This gives the process a degree of uncertainty that is difficult to represent and predict.

Inside this group of tools it is possible to include project management tools and workflow tools. Project management tools support the actual (re)planning of a process by managing the activity priorities and the resources attached to them. Among the most effective tools are Primavera, Deltek Open Plan®[8], and Artemis.

[8] Deltek Open Plan® is a registered trademark of Deltek, Inc., 13880 Dulles Corner Lane, Herndon, VA 20171, http://www.deltek.com

MS® Project 2000 is high on the list due to its widespread use, second only to MS® Excel®[9].

Workflow tools are used for the actuation of the planned process. A workflow brings information to the people who can act on it and by this means imposes the job and deadlines, priorities and responsibilities more easily and transparently on the actors. Although based on an assumed process, workflow allows for condition-based routings to specific individuals or groups of individuals. Several types of workflow tools exist ranging from cooperative and administrative to ad-hoc and production depending on the frequency and value. Among the more popular commercial tools are MQ Series of IBM, TIBCO InConcert and TIBCO Staffware[10].

1.3.3 Cooperation Tools – Computer-supported Collaborative Work Communications

New product development processes require frequent interaction among actors, both to carry out an activity and to deal with decisions and feedback. Typical systems (phone, fax, e-mail, audio and video conferencing, and chat lines) have greatly contributed to reducing development times and costs. Their general purpose matches quite well with the unstructured character of most new product development activities, and for these reasons computer-supported collaborative work communications (CSCW) are among the best tools for cooperation:

- Synchronous cooperation tools support the work of two or more actors on the same document and work one at a time, releasing control to others. Examples of these tools are Net-Meeting, eVis and Co-Create.
- Asynchronous cooperation tools support cooperation in the sense that data, also centrally managed, are updated by the allowed actors, and the change is communicated to all interested parties.

1.3.4 Management of Information – Product Management Systems

There is much knowledge to be used from the knowledge accumulated over the course of the product life cycle. Modular and platform approaches require access to past work and solutions. To capitalize on all this knowledge, a strategic requirement is to store product data concerning design, production, maintenance, etc. and to retrieve them when needed.

[9] MS® and Excel® are registered trademarks of Microsoft Corporation in the United States and/or other countries.

[10] TIBCO InConcert and TIBCO Staffware are registered trademarks of TIBCO Software Inc. in the United States and/or other countries.

Product data management (PDM) is the solution provided by ICT. Applied to new product development processes, PDM provides a centralised repository where all versions are safely stored and accessed by authorised persons. Documents of all kinds are grouped according to relationships established by meta-data and attached to each component all relevant engineering and manufacturing data, tracking versions, effectiveness and design variations (originally meant for managing engineering documents, PDM has been extended to product data over their life cycle, and in this form it has been rechristened product life cycle management (PLM). Thanks to this feature, PDM supports concurrent engineering and asynchronous work cooperation. Some of the best tools on the market are eMATRIX, Windchill[11], IMAN and ENOVIA®[12].

References

Band, W. 1995. Customer-accelerated change. *Marketing Management,* Winter: 19-33.

Best, R.J. 2000. *Market-Based Management: Strategies for Growing Customer Value and Profitability*. 2nd edn. Englewood Cliffs, NJ: Prentice Hall.

Bonnet, D. 1986. Nature of the R&D/marketing cooperation in the design of technology advanced new industrial products. *R&D Management,*16: 121-132.

Burgelman, R.A., C.M. Christensen and S.C. Wheelwright. 2003. *Strategic Management of Technology and Innovation*. New York: McGraw-Hill.

Calantone, R.J., R. Garcia and C. Dröge. 2003. The Effects of Environmental Turbulence on New Product Development Strategy Planning. *Journal of Product Innovation Management,* 20 (2): 90-103.

Clark ,K. and T. Fujimoto. 1991. *New Product Development Performance.* Boston: Harvard Business School Press.

Cooper, R.G. 1993. *Winning at New Products: Accelerating the Process from Idea to Launch.* Reading: Addison-Wesley.

Cooper, R.G. 1995. Developing new products on time. *Research and Technology Management,* 38 (5): 49-57

Cooper, R.G. 2001. *Winning at New Products: Accelerating the Process from Idea to Launch.* 3rd edn. New York: Perseus Books.

Cooper, R.G. 2005. *Product Leadership. Pathways to Profitable Innovation.* 2nd edn. New York: Basic Books.

Cooper, R.G., S.J. Edgett and E.J. Kleinschmidt. 2002. Optimizing the Stage-Gate Process: What Best Practices Companies Are Doing – Part I. *Research Technology Management,* 45 (5): 21-27.

Crawford, C.M. 1992. The hidden costs of accelerated product development. *Journal of Product Innovation Management,* 9 (3):188-199.

Daneels, E. and E.J. Kleinschmidt. 2001. Product innovativeness from the firm's perspective: Its dimensions and their impact on project selection and performance. *Journal of Product Innovation Management,* 18 (6): 357-373.

[11] PTC and Windchill, are trademarks or registered trademarks of Parametric Technology Corporation or its subsidiaries in the U.S. and in other countries.

[12] ENOVIA® is a registered trademark of Dassault Systèmes or its subsidiaries, 10, Rue Marcel Dassault, 78140 Vélizy-Villacoublay, FRANCE, http://www.3ds.com

Dröge, C., R. Calantone and N. Harmancioglu. 2008. New Product Success: Is It Really Controllable by Managers in Highly Turbulent Environments? *Journal of Product Innovation Management,* 25 (3): 272-286.

Khan, K.B. 2004. *The PDMA Handbook of New Product Development. Product Development & Management Association*. 2nd. edn. New Jersey: John Wiley & Sons.

Lawley, B. 2007. *Expert Product Management: Advanced Techniques, Tips and Strategies for Product Management & Product Marketing*. Silicon Valley: Happy About.

Maidique, M.A. and B.J. Zirger. 1984. A study of success and failure in product innovation: the case of the U.S. electronics Industry. *IEEE Transactions on Engineering Management,* 31: 192-203.

Meyer, M.H. and A.P. Lehnerd. 1997. *The Power of Product Platforms.* New York: Free Press.

Montoya-Weiss, M.M. and T.M. O´Driscoll. 2000. From experience: applying performance support technology in the fuzzy front end. *Journal of Product Innovation Management,* 17: 143-161.

Rothwell, R. and P. Gardiner. 1988. Re-innovation and robust designs: producer and user benefits. *Journal of Marketing Management,* 3 (3): 372-387.

Rothwell, R., C. Freeman and A. Horseley, V. Jervis, A.B. Robertson and J. Townsend. 1974. SAPPHO updated – Project SAPPHO Phase II. *Research Policy,* 3: 258-291.

Shank, J. and V. Govindarajan. 1988. The perils of cost allocation based on production volumes. *Accounting Horizons,* 4: 71-79.

Tucker, R. 2008. *Driving Growth Through Innovation: How Leading Firms Are Transforming Their Futures (Business).* 2nd rev. edn. New York: Berrett-Koehler.

Ulrich, K.T. and S.D. Eppinger. 2007. *Product Design and Development*. 4th rev. edn. Singapore: McGraw-Hill Education.

Wonglimpiyarat, J. 2004. The Use of Strategies in Managing Technological Innovation. *European Journal of Innovation Management,* 7 (3): 229-250.

Zirger, B. and J. Hartley. 1996. The effect of acceleration techniques on product development time. *IEEE Transactions on Engineering Management,* 43 (2): 143-152.

2 A Co-evolutionary Perspective on Distributed Manufacturing

Rob Dekkers

Division Management & Business Economics, UWS Business School, University of the West of Scotland, Paisley Campus, Paisley PA1 2BE (United Kingdom)
e-mail: rob.dekkers@uws.ac.uk

Abstract Research into Distributed Manufacturing has embraced the challenges facing industrial networks. Existing strands of research into networks often explore social-dynamic relationships and contractual aspects, thereby ignoring the underlying dynamics based on the characteristics: collaboration, decentralisation of decision-making and interorganisational integration, all pointing to mutual relationships in which co-evolution has gained a prominent place for modelling. Essential to the modelling of co-evolution is the combined development of agents involved, expressed by the factor for connected traits in the *NK[C]* model. However, in this model co-evolution happens in semi-static landscapes, which hardly exist in the reality of industry. Hence, more advanced game-theoretic applications might serve as a foundation for understanding the development of networks since these describe the interactions between agents. This chapter expands on co-evolutionary models and includes the autonomous development of agents in a network, the connectivity between agents and the dynamic forms of collaboration and communication to advance research in Distributed Manufacturing.

2.1 Introduction

When Hermann Kühnle posed a request for a different perspective on collaboration in networks, the first thing that came to mind was how does the notion of Distributed Manufacturing differ from the concept of industrial networks? For industrial networks, we might assume that collaboration has become an eminent issue which has already caught the attention of academics for a considerable time. But what about Distributed Manufacturing with its origins in information technology? For this reason, this chapter deliberates on collaboration in Distributed Manufacturing.

Most efforts in Distributed Manufacturing have been directed towards applications of information technology from the mid-1990s onwards, like the design of its architecture (e.g. Maturana and Norrie 1996; Ryu and Jung 2003), resource and task allocation (e.g. Maropoulos et al. 2002; Tharumarajah 2001), and scheduling and control (e.g. Azevedo and Sousa 2000; Candadai et al, 1996; Duffie and Prabhu 1996; Fujii et al. 1999; Kingsman 2000; Maturana and Norrie 1995; Shen 2002). All these contributions have in common that they take autonomous agents in a network as their starting point (Sousa et al. 1999). This originated in the 1980s when the introduction of flexible manufacturing systems (FMS) called for a new control paradigm; that meant moving away from the centralised resource allocation embedded in material requirements planning (MRP) applications towards decentralised decision making, and it called on computer applications to supersede the control of independent units. Consequently, the emphasis has been on manufacturing architecture and control within single plants. Later, the term Distributed Manufacturing came to include the virtual manufacturing of products crossing the borders of a monolithic company Holonic Manufacturing Systems (Van Brussel et al. 1998, p. 255), bionic manufacturing systems (BMS), Fractal Factory and multi-agent systems (Leitão and Restivo 2000, pp. 2–4) and started to include the networked organisation (e.g. Tian et al. 2002, pp. 326–327). However, the impact of this expansion has been little discussed because of the traditional focus on information technology.

With its contemporary meaning, the research into Distributed Manufacturing has disconnected from the traditional drive towards developing simulations and software applications to the issues that surround industrial networks (Kühnle et al. 2005). As a result, only a few have written about collaboration in Distributed Manufacturing (e.g. Fagerström and Jackson 2002). Collaboration is also a hot topic in industrial networks and needs expansion beyond the current concepts to arrive at a more grounded theory (Bennett and Dekkers 2005; Dekkers et al. 2004;

Dekkers and van Luttervelt 2006). This call embraces the remark of Nassimbeni (1998, p. 539) that the bulk of available works is devoted to the contractual aspects and the social dynamics of interorganisational relationships in collaboration. Most likely, that attention to contractual and social aspects originates in the direct conversion from concepts for the hierarchical firm, with the direct control of resources and its strategy towards suppliers, to concepts for networks with more loosely connected entities; Camarinha-Matos and Afsarmaneshi (2005, p. 443) provide a similar argument. Research into industrial networks has mostly neglected the dynamic forms of communication and coordination, although networks do not present a new phenomenon. To that purpose, this chapter deliberates on collaboration in Distributed Manufacturing and connects this theme to co-evolutionary models to address dynamic forms of communication and coordination.

2.1.1 Emergence of Industrial Networks

Historically, networks have existed for a long time. It will suffice to point to the Silk Route as an ancient example of the global supply chain or to the existence of trading between Asia and Europe by the Dutch *Vereenigde Oostindische Compagnie* during the Golden Age of the Republic of the Netherlands (16th and 17th centuries). Even then, the contextual environments, i.e. the social environment in which the networks existed, determined to extend the transactional environment of trading relationships. Social-economic historians have investigated this domain to understand the networks that were present during the Commercial Revolution in the Middle Ages, an era seeing the resurgence of Mediterranean and European long-distance trading (e.g. Greif 1996). Later, the global supply chains, focusing on basic needs, agricultural goods and raw materials, were affected by the Industrial Revolution (Brasseul 1998, p. 8). Firstly, growing demand during that period increased the volume of trade. Secondly, the capability of sources (regions and nations) to produce their own intermediaries or products presaged the emergence of industrial networks. For a long time, trade and industry relied on networks they created to sustain competitive advantage.

Henceforth, the academic attention paid to particular characteristics of networked organisations had already developed previously (Wiendahl and Lutz 2002, p. 1). In particular, academic interest has increased during two periods (Bennett and Dekkers 2005). The first of these was during the 1970s and 1980s when attention was focused on Japanese manufacturing concepts and techniques, including just-in-time (JIT), co-makership and keiretsu networks. The second period started during the 1990s because of the drive for even lower cost, greater efficiency and responsiveness to customer demands. This resulted in the networked organisation following the paradigm of core competencies (Prahalad and Hamel 1990), which found its origin in the resource-based view (Hemphill and Vonortas 2003, p. 261), and consequently the move towards outsourcing. The overview by Miles and Snow (1984, p. 19) illustrates the move from the simpler paradigms to the more complicated forms of network-based organisations that we have witnessed in recent decades.

2.1.2 Challenges for Contemporary Industrial Networks

In this respect, the shift from make-or-buy to co-makership and alliances, the search for flexibility in manufacturing, the emergence of concepts for computer integrated manufacturing and the design of production cells all demonstrate a continuous move to more loosely connected industrial entities. The associated flexibility has allowed an increasing degree of customisation and the production of goods on demand (Lee and Lau 1999, p. 83). Contemporary changes point to a further repositioning along the dimension of loosely connected entities with increasing pressure to respond to market opportunities and to create flexibility (Wüthrich and Philipp 1998). Hence, networks are perceived as potential solutions to the increasing demands on performance, especially those of flexibility and customisation (Dekkers and van Luttervelt 2006).

More than ever, the dominance of response time (of both product development and supply chain) and flexibility (product range and response to changes in demand) affects the operations of industrial companies. Goldman and Nagel (1993, p. 19) identified the twin characteristics of flexibility and response time as key contributors to agility. Within industrial networks, response time might be mostly associated with the reduced lead time for product development to capture product-market opportunities (note that Lee and Lau refer to “speed” instead of response time). Seizing those opportunities depends also on the capability to meet customer requirements, e.g. through order entry points a.k.a. order decoupling points (e.g. Dekkers 2006). That capability strongly depends on the competencies in the networks to collaborate and exceeds the potential of individual companies.

2.1.3 Scope of Chapter

Network organisations differ from monolithic companies in the absence of a central decision-making unit, in the lack of a consistent strategy across all the different agents and in the capability for reconfiguration (for example, the elimination of existing agents and the inclusion of new agents). This makes it difficult to deploy the concepts of the monolithic company to the domain of industrial networks (e.g. Dyer and Singh 1998, p. 661, 675; Möller and Halinen 1999, p. 416). Additionally, direct transferences of these approaches for singular entities to the realm of networked enterprises regularly fail as they lack problem-oriented interdisciplinary inferences which should rely on consilience (Wilson 1998, p. 8, 68); this is congruent with the remark of Camarinha-Matos and Afsarmanesh (2005, pp. 443–444) that research into collaborative networks constitutes a new interdisciplinary domain. Since concepts for Distributed Manufacturing applied to networks originate in concepts from manufacturing control in monolithic companies, this chapter will refer to the difference between this strand of research and the research into industrial networks, although this is not the main theme.

The core of this chapter will outline further routes for resolving issues of collaboration by looking at the evolutionary models; additionally, it will offer a synthesis of several studies regarding theories that contributes to understanding co-evolution in this respect. It represents an extension of the evolutionary concepts as introduced in Dekkers et al. (2004, pp. 70–71), and it aligns with the call for theoretical foundations by Camarinha-Matos and Afsarmanesh (2005, p. 444, 449), especially network analysis and game theory. Most of all, game theories have been used by many others (e.g. Larsson et al. 1998) to tackle issues of collaboration; these efforts not yet resulted in an overall approach, unlike the domain of evolutionary biology where these theories have gained a prominent position. This chapter must be viewed as a contribution to the discussion on foundations for a theory on networked organisations by converting models from the domain of natural sciences, with an emphasis on evolutionary biology (particularly co-evolution), to the domain of management science (the application to collaborative networks).

The chapter will start by looking into co-evolutionary models to describe collaboration. Particularly, it researches the *NK[C]* model, already identified as being of paramount importance to understanding organisational development (see McKelvey 1999). This chapter extends that model to collaboration and co-evolutionary approaches and links it to game-theoretical approaches. The next section deals with the link between co-evolution and collaboration in networks as a new rationale for Distributed Manufacturing. A final section concludes by discussing the findings and further avenues for research.

Within the domain of industrial networks, many studies have preceded this one in outlining prospects for research (e.g. Camarinha-Matos and Afsarmanesh 2005; Gulati et al. 2000; Karlsson 2003). In the view of Camarinha-Matos and Afsarmanesh, a discipline of collaborative networks should focus on the structure, behaviour and evolving dynamics of autonomous entities that collaborate to better achieve common or compatible goals. There are many perspectives from which to look at the structure and dynamics of collaborations, like technology transfer and valorisation, knowledge management and contractual relationships. This chapter elaborates on the complexity perspective for collaboration as co-evolution.

2.2 Evolutionary Perspectives

The existing strands of research are rooted in empirical studies, taken as theories drawn from observations. One other route is the formation of tentative theories, like the logic of induction (Popper 1999, p. 14). One origin of tentative theories is the natural sciences. The possible yield of perspectives of the natural sciences for the domain of social sciences, which includes management science, has been elaborated by Wilson (1998, pp. 125–163). Such a quest for consilience requires the evaluation of different perspectives. However, within the context of this chapter, the issue of collaboration has been narrowed down to the formation of tentative theories, mainly based on co-evolution.

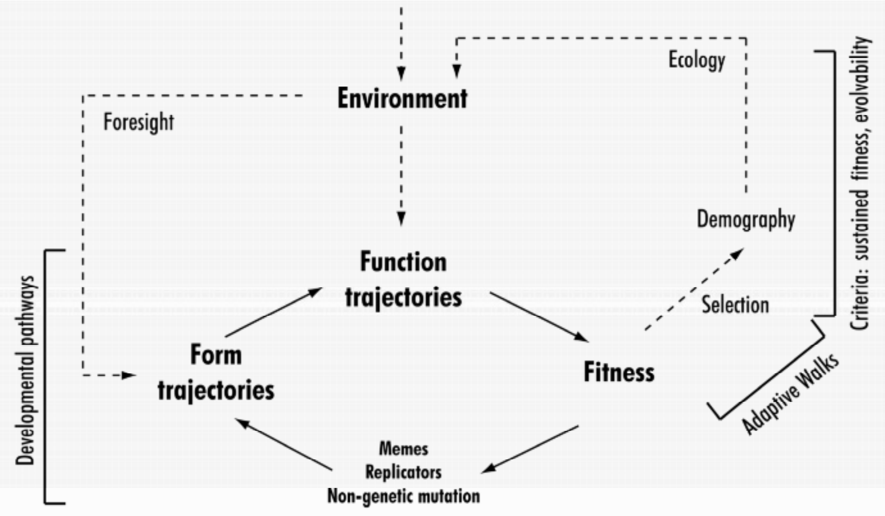


Fig. 2.1 Evolutionary mechanisms for organisations as reference model. Memes and replicators serve as input for genetic formation, which exists beside non-genetic formation. Developmental pathways determine the form and function trajectories. These pathways also relate to organisations being a class of allopoietic systems. The selectional processes select beneficial phenotypes on fitness following adaptive walks. Organisations have the capability of foresight, in contrast to organisms

The development of organisations, and therewith networks, might follow universal laws that arrive from the conversion of models from evolutionary biology. Hence, a reference model was developed to describe the interaction between organisation and environment (Fig. 2.1); it consists of two intertwined cycles: the generation of variation and the selection by the environment (Dekkers 2005, pp. 150–155). Now, one might argue that organisations are not comparable with biological entities. In any case, sufficient similarities exist to allow drawing an analogy (e.g. McCarthy 2005). In this sense, collaboration should be seen as a strategy for the phenotype, which is expressed in the fitness of an entity for selection.

Kauffman (1993) describes these fitness landscapes as mathematical models. A more powerful description is found in the emerging theory of adaptive dynamics (Geritz et al. 1997; Meszéna et al. 2001), which is based on game theory but has not been linked yet to co-evolution. The metaphor of co-evolution, the mutual dependence on each other, explains collaboration, the working together with one or more others; although not exactly identical, it provides an opportunity to explore collaboration with models from evolutionary biology.

2.2.1 Co-evolution and Industrial Networks

Even within the domain of biological (evolutionary) models, a larger number of theories exist that might describe adequately the existence of industrial networks and collaboration. In biology, co-evolution, as an adequate description for collaboration, is the mutual evolutionary influence between two species that become dependent on each other. These concepts from evolutionary biology cover a wide range of interaction between agents, for example reciprocal altruism (Trivers 1971). Within the domain of industrial networks, mutual dependence has been recognised as a potential direction for research into collaboration. Assuming this is true, how might collaboration evolve?

Co-evolution – as a basis for descriptions of dependencies – has been discovered by other management scientists such as Lewin and Volberda (1999). They focus on the emergence of new organisational forms (Lewin et al. 1999), without clearly defining the "organisational form" (McKendrick and Carroll 2001, p. 662). Co-evolution has appeared in writings that build on the work of Nelson and Winter (1982). For the purpose of this chapter, it suffices to remark that these models do not address the intertwined cycles of the reference model in Fig. 2.1. In particular, the concept of fitness landscapes is absent in the writing, which limits the validity of the outcomes. Co-evolution, when used in its sense of the mutual development of organisms, favors selectional forces (i.e. survival in the long run). Thus describing co-evolution starts with fitness landscapes as an expression of the fitness of the associated genotypes.

2.2.2 Fitness Landscapes

Fitness resembles height, a measure for expressing the fitness of a genotype, similar to Wright's adaptive landscape (Wright 1982). Fitter genotypes move at greater heights than less fit genotypes. Consider a genotype with only four genes, each having two alleles: *1* and *0* (i.e. a Boolean representation of the state of each gene), resulting in 16 possible genotypes, each a unique combination of the different states of the four genes (Fig. 2.2). Each vertex differs by only one mutation from the neighbouring vertices, representing the step of a single mutation, thereby showing that each mutation as such is independent of the state of the other genes. An adaptive walk begins at any vertex, moves to vertices that have higher fitness values and ends at a local optimum, not necessarily the highest optimum (a vertex that has a higher fitness value than all its one-mutant neighbours). Figure 2.2 shows that three local optima exist where adaptive walks may end. In random landscapes, looking for the global peak by searching uphill is useless; it is tantamount searching the entire space of possibilities (Kauffman 1995, pp. 166–167). In the *N* model, the traits are not related.

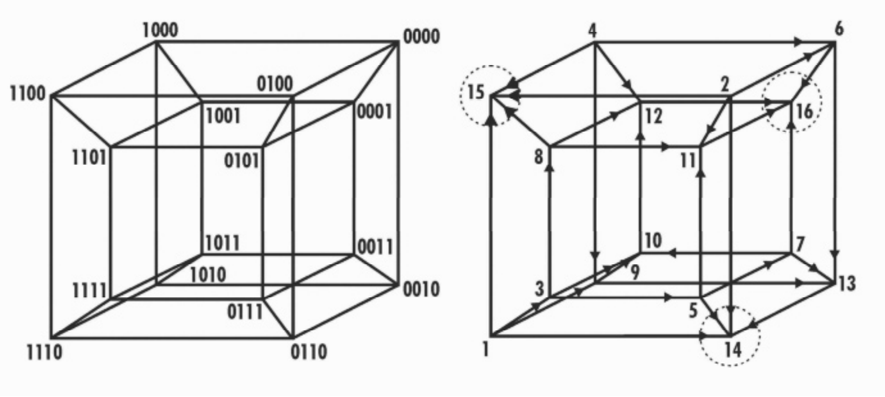
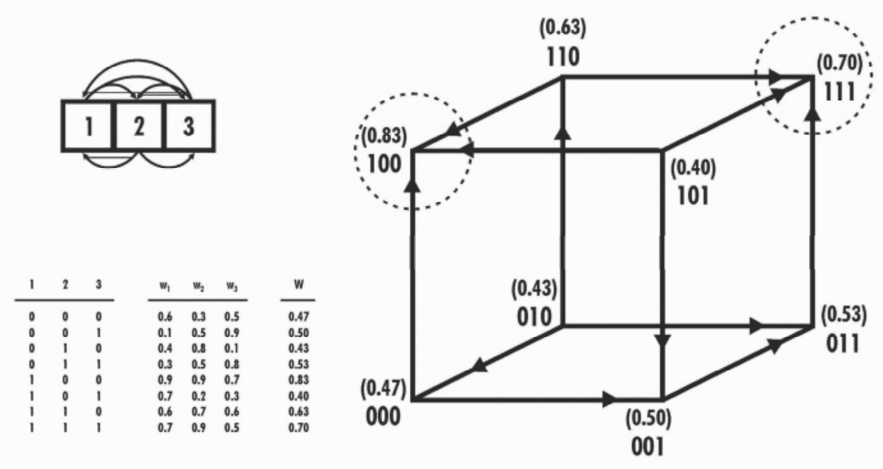


Fig. 2.2 The N-model as proposed by Kauffman (1993, p. 38). Sixteen possible peptides 4 aminos long are arranged as vertices on a four-dimensional Boolean hypercube. Each peptide connects to its four one-mutant neighbours, accessible by changing a single amino acid from 1 to 0 or from 0 to 1. The hypercube on the left represents this four-dimensional peptide space. In the hypercube on the right-hand side, each peptide has been assigned, at random, a rank-order fitness, ranging from the worst, 1, to the best, 16. Directions of such moves between adjacent positions are shown by arrows from the less fit to the more fit. Peptides fitter than all one-mutant neighbours are local optima (three in this case)

However, in reality, the fitness landscapes that underlie the mutation steps of gradualism are correlated, and local peaks do often have similar heights. Through the existence of particular evolutionary phenomena (developmental pathways, regulatory genes and epigenetics), no gene exists on its own; all genes correlate to other genes; this is often referred to as epistatic coupling or epistatic interactions. Rugged landscapes are those landscapes in which the fitness of one gene depends on that one part and upon *K* other parts among the *N* present in the landscape. Building on this, the *NK* model offers further insight into the mechanisms of evolution and selection (Kaufmann 1993, pp. 40–54). Again, consider an organism with *N* gene loci, each with two alleles, *1* and *0*. Let *K* stand for the average number of other loci, which epistatically affect the fitness contribution of each locus. The fitness contribution of the allele at the i locus depends on itself (whether it is 1 or 0) and on the other alleles, *1* or *0*, at *K* other loci, hence upon *K+1* alleles. The number of combinations of these alleles is just 2^{K+1}. Kauffman selects at random from each of the 2^{K+1} combinations a different fitness contribution from a uniform distribution between *0.0* and *1.0* (Fig. 2.3). The fitness of one entire genotype can be expressed as the average of all of the loci. Generally, epistatic interactions create a more deformed landscape.

Despite the importance of fitness landscapes for evolutionary processes, Kauffman (1995, p. 161) states that biologists hardly know what such fitness landscapes look like or how successful a search process is as a function of landscape structure. The landscapes may vary from smooth, single-peaked to rugged, multi-peaked landscapes. During evolution, species search these landscapes using mutation, recombination and selection, a process for which the *NK* model provides insight into particular phenomena accompanying the adaptive walk.



1	2	3	w_1	w_2	w_3	W
0	0	0	0.6	0.3	0.5	0.47
0	0	1	0.1	0.5	0.9	0.50
0	1	0	0.4	0.8	0.1	0.43
0	1	1	0.3	0.5	0.8	0.53
1	0	0	0.9	0.9	0.7	0.83
1	0	1	0.7	0.2	0.3	0.40
1	1	0	0.6	0.7	0.6	0.63
1	1	1	0.7	0.9	0.5	0.70

Fig. 2.3 *NK* model as developed by Kauffman (1993, p. 42). In the upper left corner it shows the assignment of *K*=2 epistatic inputs to each site. These fitness values then assign fitness to each of the 2^3=8 possible genotypes as the mean value of the fitness contributions of the three genes. The figure depicts the fitness landscape on the three-dimensional Boolean cube corresponding to the fitness values of the eight genotypes. More than one local optimum exists

These fitness landscapes have already been used in the context of networks. Worth mentioning is the work of Kaufman et al. (2000), who show that searches are most likely more effective for combining technologies rather than those for new technologies; this finding indicates firms collaborating by combining technologies might have more success than those that search solely for new technologies. Wilkinson et al. (2000) apply the concept of fitness landscapes to the case of automotive distributors and dealers, illustrating their interdependence. They conclude that firms operate in complex adaptive systems in which control is distributed throughout the system, in fact, the realm of Distributed Manufacturing. Nevertheless, the *NK* model needs supplementation because it describes the fitness of one species, i.e. one type of company, and not of more species dependent on each other, the domain of co-evolution.

2.2.3 Co-evolution and the NK model

Kauffman (1993, pp. 243–245) extends the *NK* model to co-evolution by adding the constraint that each trait in species *1* depends epistatically on *K* traits internally and on *C* traits in species *2*, the so-called *NK[C]* model. More generally, in an ecosystem with *S* species, each trait in a species will depend on *K* traits internally and on *C* traits in each of the S_i among the *S* species with which it interacts. Therefore, if one species adapts, it both changes the fitness of other species and deforms their landscapes in the *NK[C]* model.

The coupling of the fitness landscapes will affect the search for increased fitness (Kauffman 1993, pp. 252–253). When a new link is introduced (i.e. increasing *K*), the genetic locus spreads throughout a population in three ways: (a) the new epistatic link, when it forms, causes the genotype to be fitter, (b) the new epistatic link is near neutral and spreads through the population by random drift, and (c) the new link not only has a direct effect on the fitness of the current genotype but also increases the inclusive fitness of the individual and its genetic descendants. This suggests that optimisation in co-evolutionary dynamics becomes possible by optimisation mechanisms that search for optimal traits in relation to the coupled traits (we could view the development of the Pearl River Delta in that respect Noori and Lee (2006); The Economist (2002)). The second option for a network consists of increasing its reach, which is like increasing the number of species *S*. When that happens, the waiting time to encounter a new equilibrium increases, the mean fitness of the co-evolving partners decreases (McKelvey 1999, p. 312), and the fluctuations in fitness of the co-evolving partners increase dramatically. The increase of agents might lead to a new optimisation in traits and coupled traits, but only after going through a period of instability.

2.2.4 Percolation in Networks

These instabilities might come along with phase changes, or percolation, in the Boolean networks captured in the *NK* model (Kauffman 1995, pp. 80–92). Four particular states arise when the *NK* model is analysed for the principles of self-organisation. Firstly, at *K*=1, the orderly regime appears, in which independent subsystems function as largely isolated islands with minimal interaction. At *K*=2, the network is at the edge of chaos, the ordered regime rules at maximum capacity but chaos is around the corner. At values ranging from *K*=2 to *K*=5 the transition to chaos appears although indications are that this transition happens already before $K=3$. From $K>5$, the network displays chaotic behaviour. All four of these possibilities of K indicate that the behaviour of networks strongly varies according to the connectivity.

In addition, human-influenced complex networks, e.g. World Wide Web, human acquaintance networks, have common properties for connectivity, which are hardly compatible with existing cybernetic approaches (as mostly present in software applications). The so-called *small-world property*, the best known of these specific properties, states that the average path length in the network is small relative to the system size (Milgram 1967). This phenomenon was already scientifically studied more than three decades ago, long before becoming notorious. In fact, the phrase *six degrees of separation* (Guare 1990), another popular slogan depicting the small-world phenomenon, is due to Milgram's 1967 experiment. Another property of complex networks is clustering, i.e. the increased probability that pairs of nodes with a common neighbour are also connected. Since 1967, increased efforts have been dedicated to identifying other measures of complex (enterprise) networks (Fricker 1996). Perhaps the most important is the *distribution of degrees*, i.e. the distribution of the number of links the nodes have. It has been shown that several real-world networks have scale-free distributions, often in the form of a power law. In these networks, a huge number of nodes have only one or two neighbours, while a couple of them are massively connected (similar to order and chaos in the *NK* model). These three specific properties of human-influenced networks strongly influence the behaviour of the constituent agents and the development of these networks.

The properties have been translated into mathematical models and applications focusing on large networks and connectivity (e.g. Klemm et al. 2003; Krapivsky and Redner 2001; Newman 2003; Watts and Strogatz 1998); most of these applications show that these properties make networks behave more dynamically. Industrial networks consist of a limited number of agents – consider the industry sector for flow-wrapping packaging equipment that consists of only 300 to 350 companies worldwide – and therefore, might display behaviour other than that of large networks. The expansion to industrial networks should include the behaviour of agents (not just agents as nodes) and the development of traits for selection for smaller networks.

2.2.5 Symbiosis

The concept of symbiosis deserves some more attention as a form of co-evolution in networks. Symbiosis is an interaction between two organisms living together in more or less intimate association or even the merging of two dissimilar organisms. The various forms of symbiosis include:

- Parasitism, in which the association is disadvantageous or destructive to one of the organisms and beneficial to the other;

- Mutualism, in which the association is advantageous to both;
- Commensalism, in which one member of the association benefits while the other is not affected;
- Amensalism, in which the association is disadvantageous to one member while the other is not affected.

In some cases, the term symbiosis is used only if the association is obligatory and benefits both organisms. Sometimes, altruistic behaviour benefits another organism not necessarily closely related. While being apparently detrimental to the organism the behaviour (Trivers 1971, p. 35; Aldrich 1999, p. 301) differentiates between commensalism referring to competition and cooperation between units and symbiosis taken as mutual interdependence between dissimilar units. Symbiosis as defined in this chapter does not restrict the term only to mutually beneficial interactions. It has strong similarities to the coupling of the traits in the *NK* model to describe co-evolution; these traits might lead to cooperative species as Potter and de Jong (2000, p. 26) demonstrate, albeit based on generic algorithms that can hardly account for the dynamics of the organisations' environment. It indicates that mutual relationships have at least two dimensions: the fitness of each of the two agents involved.

2.3 Distributed Manufacturing and Co-evolution

Taking Distributed Manufacturing as a concept for autonomous agents that are mutually dependent on each other, equivalent to complex adaptive systems, what does the perspective of co-evolution hold? This question goes beyond issues like network architecture, resource allocation and scheduling, the traditional domain of software applications. Rather it focuses on the specific characteristics of (international) networks of companies: collaboration, decentralisation of decision-making and interorganisational integration (O'Neill and Sackett 1994, p. 42). The traditional themes of research into Distributed Manufacturing support the decentralisation of decision-making and the interorganisational integration; the move towards industrial networks implies that collaboration should be covered, too.

2.3.1 New Rationales for Distributed Manufacturing

This calls for new rationales for the contemporary meaning of Distributed Manufacturing which view the networks as a co-evolutionary system, i.e. agents dependent on each other. The similarity in the new and old approaches, the autonomous agents, serve as a basis for looking for models and tools that adequately address the challenges of networks. The move towards more loosely connected entities calls for models of collaboration that stretch beyond the emphasis on contractual and social dynamics of interorganisational relationships, which represents the main stream of research into networks. In that respect tools like matchmaking and brokerage through Web services (Field and Hoffner 2003; Molina et al. 2003), and electronic contracts (Angelov and Grefen 2003; Barata and Camarinha-Matos 2003) will insufficiently counter the challenges of industrial networks. Concepts like factory-on-demand (Lee and Lau 1999) and the research into industrial districts (e.g. Biggiero 1999) align more with the principles of complex adaptive systems as systems of human interaction, driven by the search for governing laws of collaboration. Hence, research into Distributed Manufacturing should include concepts of agents dependent on each other to account for the human factor.

Even though some of the concepts in Distributed Manufacturing account for the human factor, like the concept of holonic systems, or take the biological perspective, like the concept of bionic systems (Leitão and Restivo 2000, p. 3), they do so by looking at the collaboration from an information technology perspective. The conversion of truly biological concepts to the domain of networked organisations will yield additional insight, especially into the interaction between humans (actors) as agents. The mutual relationships point to connectivity and coupling where traits become interrelated; companies engage in new relationships and industrial networks evolve. The dynamics of these networks represent the search for increased fitness by the constituent agents; henceforth, research into Distributed Manufacturing should embrace connectivity and coupling of traits to describe the mutual relationships of agents.

Similar to the mutual relationships of symbiosis, this implies that both the fitness of individual agents and mutual fitness should be accounted for. In that perspective, Khanna et al. (1998) have used the terms private and common benefits. They state that in a partnership, each enterprise has cooperative as well as competitive motives. The cooperative aspect arises from the fact that firms can collectively use their knowledge to produce something that is beneficial to them all (common benefits). The competitive aspect is a consequence of each firm's attempt to use the knowledge of its partners for private gains, the motive for setting up strategic networks (Hemphill and Vonortas 2003, pp. 260–261). For a sustainable partnership, a combination of private and common benefits is needed, its ratio described by relative scope (Khanna et al. 1998, p. 195). When private benefits are the only motive of a company, racing behaviour will arise and the alliance will be cancelled after a while. Kale et al. (2000) demonstrate the same idea based on a contingency model for interorganisational learning and opportunistic behaviour. Henceforth, the perception of agents in networks about relative scope will drive their behaviour and ultimately the development of the network; this requires that research into Distributed Manufacturing should incorporate both private and common benefits.

2.3.2 Models for Co-evolution in Collaborative Networks

Already, several approaches exist in the literature to describe the evolution of cooperation and collaboration as mutual behaviour. Dierkes et al. (2001, p. 665) state that the evolution of coorporations can be seen as the development of a cooperative alliance over time. Doz (1996, p. 55) stresses that the evolution of cooperation might be constrained by the conditions of the inception of the alliance and influenced by the consequent collaboration process. Larsson et al. (1998, pp. 291–295) propose two different interorganisational learning dynamics using game theories. Both describe the dynamics of the transparency and receptivity as a result of (initial) conditions. The first kind of interorganisational learning dynamics deals with possible barriers, while the second one concentrates on empowerment. Understanding the evolution of alliances can provide critical insight into how such ties can be managed (Gulati 1998, pp. 305–306). This underlines that collaboration in concepts for Distributed Manufacturing should account for learning behaviour.

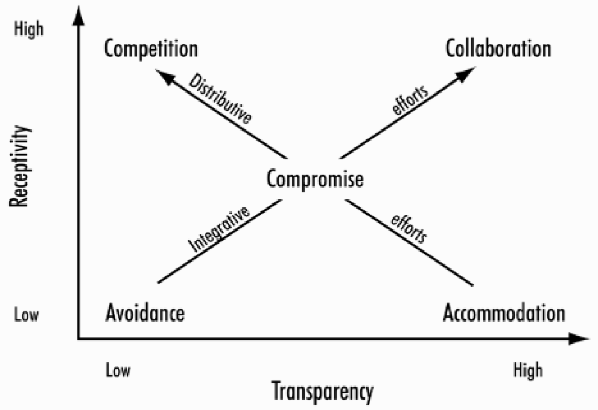


Fig. 2.4 Individual strategies for interorganisational learning (Larsson et al. 1998, p. 289). The integrative dimension concerns the total joint outcome, from avoidance to collaboration, and the distributive dimension indicates one party's share of the joint outcome, ranging from accommodation to competition

According to Larsson et al. (1998, p. 289), interorganisational learning is a joint venture of interacting organisations' choices to be more or less transparent or receptive. Within this setting, each organisation has five different strategies at its disposal: collaborate, compete, compromise, accommodate and avoide (Fig. 2.4). Collaboration represents the ultimate strategy for both agents to create benefits, but because of the high score on transparency, might easily lead to exploitation by other firms. The framework is expanded with the initial research of Parkhe (1993), who proposed a game-theoretic view to understand and describe the mixed-motive (cooperative vs. collaborative) nature of interfirm relationships. The resulting dynamic barriers to interorganisational learning (Larsson et al. 1998, p. 292) are presented in Fig. 2.5. These interorganisational learning strategies show different outcomes depending on the initial strategies of each agent in the network. To that purpose, the effect of initial conditions on learning behaviour potentially influences the effectiveness of concepts for Distributed Manufacturing.

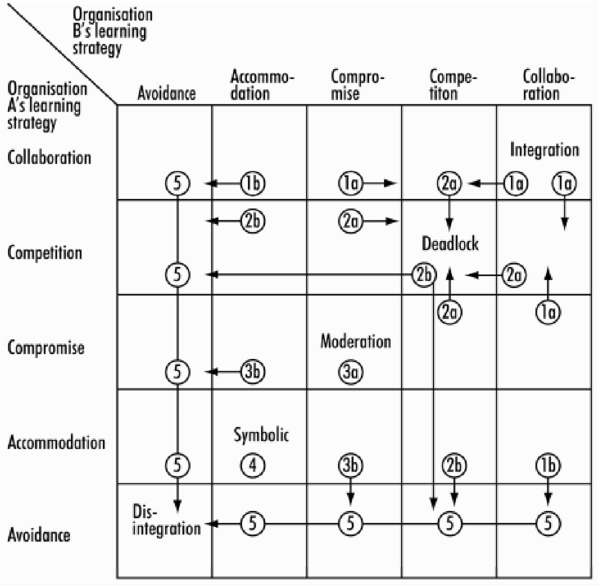


Fig. 2.5 Dynamic barriers to organisational learning (according to Larrson et al. 1998, p. 292). The figure indicates the pathways of interaction depending on the individual organisation's actions. Arrows show which new combination is likely to develop from original starting positions determined by the actions from Fig. 2.4. Most likely, the dyadic relationships will end in disintegration (resulting in arms-length contracts) or collaboration

2.3.3 Game Theories and Collaborative Networks

In comparison to the *NK[C]* model, the development of interorganisational learning might have a limited number of outcomes. Clearly, in both models, the individual organisations undertake adaptive walks to increase fitness, and these fitnesses mutually depend on each other. But according to the *NK[C]* model, more local optima will exist, which aligns with the more advanced modelling by adaptive dynamics; this strand of research has the strength that it recognises different criteria for (in)stability that will affect the evolutionary outcomes. All three streams exploit the game-theoretic applications in different fashions and all three might lead to different underpinnings of Distributed Manufacturing models.

It is too early to conclude which models or which combinations best explain the phase transitions in collaborative networks, like those in Distributed Manufacturing. This becomes more complicated when considering the outcomes of social-economic research into networks. Using game-theoretic considerations Greif (1993) examined the social-economic relationships with respect to the Jewish Maghribi traders who operated during the 11th century in the Muslim Mediterranean. This investigation reflects a reciprocity based on a social and commercial information network with very flexible, but not bilateral, agency relations (even when imposing rules on the distribution of common and private benefits); Uzzi (1997, p. 38) points out also that these types of regularities fit with the behaviour observed in networks. The Maghribis' network expanded from within rather than relying on outsiders. Hence, collective punishment prevailed in contrast to Italian traders who operated (particularly from the 12th century on) in the same area as the Maghribis, trading in the same goods and utilising comparable naval technology. Among the Italian traders bilateral rather than collective punishment existed (Greif 1994; Uzzi 1997, p. 38). Within a game-theoretic view, networks might operate in different modes with quite different rules, guidelines and interactions (Gulati et al. 2000, pp. 209–210) mentions similar findings; this perspective might lead to a better understanding of dynamic forms of communication that should be added to theories and concepts for Distributed Manufacturing.

2.3.4 Avenues for Research

If evolutionary models based on game theories address issues of collaboration in industrial networks, they should incorporate fitness landscapes and at least two dimensions of fitness (i.e. the fitnesses of mutually dependent agents). The current model of Larsson et al. (1998) and the semi-static *NK* model insufficiently incorporate these features and do not address the evolution of the network itself; the *NK[C]* model offers an explanation by addressing the coupled landscapes but still offers a semi-static view. Therefore, these models might be expanded with the dynamics of the environment captured by adaptive dynamics. According to Lawless (2002), the more advanced quantum game theory also accounts for these dynamics (e.g. Eisert et al. 1999) and avoids the traditional pitfall of game theory, which overstates cooperation (e.g. van Enk and Pike 2002); Colman (2003) points to the weakness of the orthodox game theories. Pietiranen (2004, pp. 403–407) states that game theories adequately connect to multi-agent systems (which closely relate to general systems theories). The research presented in Dekkers et al. (2004) captures these findings as the starting point for new avenues that could also include research into Distributed Manufacturing.

Further, through consilience by synthesis (Wilson 1998, p. 68) such research would be able to relate these models and findings through simulations to the contemporary challenges of industrial networks. Loosely connected entities experience greater instability than the fixed forms of initial networks like alliances and partnerships. Even then, other research has indicated the instability of these arrangements, as a natural mechanism for dissolving (Kogut 1989) or as a power and trust perspective (Gulati et al. 2000, p. 209). This will emphasise the search for chaos and order in the networked regime that applies to both industrial networks and Distributed Manufacturing.

Therefore, the application of the evolutionary models of fitness landscapes and game theories might underpin new and more effective models for comprehending the dynamics of collaborative relationships. In addition, the different modes of these theories, arriving originally from evolutionary biology, call for synthesis to fully understand the interrelationships between agents and their actions. The research domain of collaborative networks will profit from these new, more effective models and in that way will become a true discipline in its own right. Even archival research might be used to compare findings related to these more dynamic approaches to enhance our understanding of their development; the literature used in this chapter represents only a fraction of the available works on the matter and can only be considered as indicative of the advancements made by research into collaborative networks. Although similar conclusions have been reached by others (e.g. Gulati 1998, pp. 304–306), the underlying theories have not been expanded as in this chapter. We have not yet reached the stage where the formation of tentative theories and their evaluation have resulted in grounded theory that underpins the behaviour of autonomous agents in networks and that allows the design of sustainable industrial networks.

2.4 Conclusion

The foregoing discussion implies that the research into Distributed Manufacturing, characterised by control of autonomous agents, has gone beyond the reach of information technology itself; hence it has become necessary to include collaboration. This inclusion drives the research in the direction of that into industrial networks where collaboration (emerging in different forms) is common ground. Many research efforts into industrial networks focus on the identification of contractual aspects and the social dynamic of interorganisational relationships. They have proven insufficient to address the characteristics of networks: collaboration, decentralisation of decision-making and interorganisational integration, which calls for approaches that are more dynamic. But Distributed Manufacturing has always taken autonomous agents as a starting point for developing software applications for control; the loosely connected entities in contemporary networks follow their own autonomous strategies, and henceforth the base of Distributed Manufacturing might address the issues surrounding the dynamics of networks if it includes concepts for collaboration.

Models for co-evolution, originating in evolutionary biology and especially those based on game theories, might prove fertile ground for developing more adequate collaboration models for industrial networks. Part of the literature views co-evolution from the perspective of the monolithic company and arrives at conclusions that fit circumstances that are more static. The decentralisation of decision-making entails that partners in industrial networks behave like autonomous agents that mutually interact and requires dynamic descriptions. The interaction in networks will benefit from insight into game-theoretic applications to understand the underlying patterns, such as the investigations of ancient trading networks. Even that research shows that industrial networks display dynamic behaviour that evolves over time and that bilateral relationships or collective networks shape the interactions.

Game-theoretic models that incorporate private and common benefits and that make it possible to analyse the instability of networks should lead to new, grounded theory. Those models cover the internal development of traits by agents, their associated strategy, the connectivity (including the interorganisational integration) and the dynamics of the environment. So far, these models are found in separate strands of research; they need to be expanded and further synthesised to produce new insights that will advance our understanding of how industrial networks operate and how Distributed Manufacturing will contribute to addressing the collaborative challenges of these networks.

References

Aldrich, H. 1999. *Organizations Evolving*. London: Sage

Angelov, S. and P. Grefen. 2003. The 4w framework for b2b e-contracting. *International Journal of Networking and Virtual Organizations,* 2: 78-97.

Azevedo, A.L. and J.P. Sousa. 2000. A component-based approach to support order planning in a distributed manufacturing enterprise. *Journal of Materials Processing Technology,* 107: 431-438.

Barata, J. and L.M. Camarinha-Matos. 2003. Coalitions of manufacturing components for shop floor agility - the cobasa architecture. *International Journal of Networking and Virtual Organizations,* 2: 50-77.

Bennett, D. and R. Dekkers. 2005. Industrial networks of the future-a critical commentary on research and practice. In: Proceedings of the 12th International EurOMA Conference. Budapest.

Biggiero, L. 1999. Market, hierarchies, networks, districts: a cybernetic approach. *Human Systems Management,* 18: 71-86.

Brasseul, J. 1998. Une revue des interprétations de la révolution industrielle. *Revue Région et Développement*: 1-74.

Brussel, H.V., J. Wyns, P. Valckenaers, L. Bongaerts and P. Peeters. 1998. Reference architecture for holonic manufacturing systems: Prosa. *Computers in Industry,* 37: 255-274.

Camarinha-Matos, L.M. and H. Afsarmanesh. 2005. Collaborative networks: a new scientific discipline. *Journal of Intelligent Manufacturing,* 16: 439-252.

Candadai, A., J.W. Herrmann and I. Minis. 1996. Applications of group technology in distributed manufacturing. *Journal of Intelligent Manufacturing,* 7: 271-291.

Colman, A.M. 2003. Beyond rationality: rigor without mortis in game theory. *Behavorial and Brain Sciences,* 26: 180-198.

Dekkers, R. 2005. *(R)evolution, Organizations and the Dynamics of the Environment*. New York: Springer.

Dekkers, R. 2006. Engineering management and the order entry point. International *Journal of Production Research,* 44: 4011-4025.

Dekkers, R. and C.A.V. Luttervelt. 2006. Industrial networks: Capturing changeability? *International Journal of Networking and Virtual Organisations,* 3: 1-24.

Dekkers, R., A. Sauer, M. Schönung and G. Schuh. 2004. Collaborations as complex systems. In: Designing and Operating Global Manufacturing & Supply Networks, 9th Annual Cambridge International Manufacturing Symposium, 60-77. Cambridge: IMNet/CIM.

Dierkes, M., A.B. Antal, J. Child and I. Nonaka. 2001. *Handbook of Organizational Learning and Knowledge*. Oxford: Oxford University Press.

Doz, Y.L. 1996. The evolution of cooperation in strategic alliances: initial conditions or learning processes. *Strategic Management Journal,* 17: 55-83.

Duffie, N.A. and V.V. Prabhu. 1996. Heterarchical control of highly distributed manufacturing systems. *International Journal of Computer Integrated Manufacturing,* 9: 270-281.

Dyer, J.H. and H. Singh. 1998. The relational view: cooperative strategy and sources of interorganizational competitive advantage. *Academy of Management Review,* 23: 660-679.

Economist. 2002. A new workshop of the world. Issues 59-60.

Eisert, J., M. Wilkens and M. Lewenstein. 1999. Quantum games and quantum strategies. *Physical Review Letters,* 83: 3077-3080.

Enk, S.J.V. and R. Pike. 2002. Classical rules in quantum games. *Physical Review A,* 66: 024306/1-2.

Fagerström, B. and M. Jackson. 2002. Efficient collaboration between main and subsuppliers. *Computers in Industry,* 49: 25-35.

Field, S. and Y. Hoffner. 2003. Web services and matchmaking. *International Journal of Networking and Virtual Organizations,* 2: 16-32.

Fricker, A.R. 1996. *Eine Methodik zur Modellierung, Analyse und Gestaltung komplexer Produktionsstrukturen*. Aachen: RWTH Aachen.

Fujii, S., A. Ogita, Y. Kidani and T. Kaihara. 1999. Synchronization mechanisms for integration of distributed manufacturing simulation systems. *Simulation,* 72: 187-197.

Geritz, S.A.H., J.A.J. Metz, E. Kisdi and G. Meszéna. 1997. Dynamics of adaptation and evolutionary branching. *Physical Review Letters,* 78: 2024-2027.

Goldman, S.L. and R.N. Nagel. 1993. Management, technology and agility: the emergence of a new era in manufacturing. *International Journal of Technology Management,* 8: 18-38.

Greif, A. 1993. Contract enforceability and economic institutions in early trade: the maghribi traders' coalition. *American Economic Review,* 83: 525-548.

Greif, A. 1994. Cultural beliefs and the organization of society: a historical and theoretical reflection on collectivist and individualist societies. *Journal of Political Economy,* 102: 912-950.

Greif, A. 1996. Economic history and game theory: a survey. In: *Handbook of Game Theory*, 1989-2024. Amsterdam: North Holland.

Guare, J. 1990. *Six Degrees of Separation: A Play*. New York: Vintage Books.

Gulati, R. 1998. Alliances and networks. *Strategic Management Journal,* 19: 293-317.

Gulati, R., N. Nohria and A. Zaheer. 2000. Strategic networks. *Strategic Management Journal,* 21: 203-215.

Hemphill, T.A. and N. Vonortas, S. 2003. Strategic research partnerships: a managerial perspective. *Technology Analysis & Strategic Management,* 15: 255-271.

Kale, P., H. Singh and H. Perlmutter. 2000. Learning and protection of proprietary assets in strategic alliances: building relational capital. *Strategic Management Journal,* 21: 217-237.

Karlsson, C. 2003. The development of industrial networks: challenges to operations management in an extraprise. *International Journal of Operations & Production Management,* 23: 44-61.

Kauffman, S. 1995. *At home in the universe: The search for laws of self-organization and complexity*. New York: Oxford University Press.

Kauffman, S.A. 1993. *The origins of order: self-organization and selection in evolution*. New York: Oxford University Press.

Kaufman, A., C.H. Wood and G. Theyel. 2000. Collaboration and technology linkages: a strategic supplier typology. *Strategic Management Journal,* 21: 649-663.

Khanna, T., R. Gulati and N. Nohria. 1998. The dynamics of learning alliances: competition, cooperation and relative scope. *Strategic Management Journal,* 19: 193-210.

Kingsman, B.G. 2000. Modelling input-output workload control for dynamic capacity planning in production planning systems. *International Journal of Production Economics,* 68, 73-93.

Klemm, K., V.M. Eguíluz, R. Toral and M. San Miguel. 2003. Nonequilibrium transitions in complex networks: a model of social interaction. *Physical Review E,* 67: 026120/1-6.

Kogut, B. 1989. The stability of joint ventures: reciprocity and competitive rivalry. *The Journal of Industrial Economics,* 38: 183-198.

Krapivsky, P.L. and S. Redner. 2001. Organization of growing random networks. *Physical Review E,* 63: 066123/1-14.

Kühnle, H., U. Hess and D. Scheffter. 2005. Distributed planning and design of production systems. In: Integrated engineering of products, services and organisations (11th International Conference on Concurrent Enterprising), pp 3-10, Munich.

Larsson, R., L. Bengtsson, K. Henriksson and J. Sparks. 1998. The inter-organizational dilemma: collective knowledge development in strategic alliances. *Organization Science,* 9: 285-305.

Lawless, W.F. 2002. Adversarial collaboration decision-mking: an overview of social quantum information processing. In: Collaboration Learning Agents AAAI - Spring Symposium, 122-123. Stanford, CA.

Lee, W.B. and H.C.W. Lau. 1999. Factory on demand: the shaping of an agile production network. *International Journal of Agile Management Systems,* 1: pp 83-87.

Leitão, P. and F. Restivo. 2000. A framework for distributed manufacturing applications. In: *Advanced Summer Institute,* pp 1-7. Bordeaux.

Lewin, A.Y., C.P. Long and T.N. Carroll. 1999. The coevolution of new organizational forms. *Organization Science,* 10: pp 535-550.

Lewin, A.Y. and H.W. Volberda. 1999. Prolegomena on coevolution: a framework for research on strategy and new organizational forms. *Organization Science,* 10: pp 519-534.

Maropoulos, P.G., K.R. Mckay and D.G. Bramall. 2002. Resource-aware aggregate planning for the distributed manufacturing enterprise. *Annals of the CIRP,* 51: pp 363-366.

Maturana, F.P. and D.H. Norrie. 1995. A generic mediator for multi-agent coordination in a distributed manufacturing system. In: IEEE International Conference on Systems, Man and Cybernetics, pp 952-957. New York: IEEE.

Maturana, F.P. 1996. Multi-agent mediator architecture for distributed manufacturing. *Journal of Intelligent Manufacturing,* 7: 257-270.

Mccarthy, I.P. 2005. Toward a phylogenetic reconstruction of organizational life. *Journal of Bioeconomics,* 7: 271-307.

Mckelvey, B. 1999. Avoiding complexity catastrophe in coevolutionary pockets: strategies for rugged landscapes. *Organization Science,* 10: 294-321.

Mckendrick, D.G. and G.R. Carroll. 2001. On the genesis of organizational forms: evidence from the market for disk arrays. *Organization Science,* 12: 661-682.

Meszéna, G., É. Kisdi, U. Dieckmann, S.A.H. Geritz and J.A.J. Metz. 2001. Evolutionary optimisation models and matrix games in the unified perspective of adaptive dynamics. *Selection,* 2: 193-210.

Miles, R.E. and C.C. Snow. 1984. Fit, failure and the hall of fame. *California Management Review,* 26: 11-28.

Milgram, S. 1967. The small world problem. *Psychology Today,* 2: 60-67.

Molina, A., R. Mejía and M. Velandia. 2003. Core processes, methods and e-services to support virtual enterprise brokerage. *International Journal of Networking and Virtual Organizations,* 2: 33-49.

Möller, K.K. and A. Halinen. 1999. Business relationships and networks: managerial challenge of network era. *Industrial Marketing Management,* 28: 413-427.

Nassimbeni, G. 1998. Network structures and co-ordination mechanisms: a taxonomy. *International Journal of Operations & Production Management,* 18: 538-554.

Nelson, R.R. and S.G. Winter. 1982. *An Evolutionary Theory of Change*. Cambridge, MA: Belknap Press.

Newman, M.E.J. 2003. The structure and function of complex networks. *SIAM Review,* 45: 167-256.

Noori, H. and W.B. Lee. 2006. Dispersed network manufacturing: adapting SMEs to compete on the global scale. *Journal of Manufacturing Technology Management,* 17: 1022-1041.

O'neill, H. and P. Sackett. 1994. The extended manufacturing enterprise paradigm. *Management Decision,* 32: 42-49.

Parkhe, A. 1993. Strategic alliance structuring: a game theoretic and transaction cost examination of interfirm cooperation. *Academy of Management Journal,* 36: 794-829.

Pietarinen, A.-V. 2004. Multi-agent systems and game theory – a peircean manifesto. *International Journal of General Systems,* 33: 395-414.

Popper, K. 1999. *All Life Is Problem Solving*. London: Routledge.

Potter, M.A. and K.A.D. Jong. 2000. Cooperative coevolution: an architecture for evolving coapdated subcomponents. *Evolutionary Computation,* 8: 1-29.

Prahalad, C.K. and G. Hamel. 1990. The core competence of the corporation. *Harvard Business Review,* 168: 79-91.

Ryu, K. and M. Jung. 2003. Agent-based fractal architecture and modelling for developing distributed manufacturing systems. *International Journal of Production Research,* 41: 4233-4255.

Shen, W. 2002. Distributed manufacturing scheduling using intelligent agents. *IEEE Intelligent Systems,* 17: 88-94.

Sousa, P., N. Silva, T. Heikkila, M. Kollingbaum and P. Valckenaers. 1999. Aspects of cooperation in distributed manufacturing systems. In: 2nd Workshop on Intelligent Manufacturing, 695-717. Leuven.

Tharumarajah, A. 2001. Survey of resource allocation methods for distributed manufacturing systems. *Production Planning & Control,* 12: 58-68.

Tian, G.Y., G. Yin and D. Taylor. 2002. Internet-based manufacturing: a review and a new infrastructure for distributed intelligent manufacturing. *Journal of Intelligent Manufacturing,* 13: 323-338.

Trivers, R.L. 1971. The evolution of reciprocal altruism. *Quarterly Review of Biology,* 46: 35-57.

Uzzi, B. 1997. Social structure and competition in interfirm networks: the paradox of embeddedness. *Administrative Science Quarterly,* 42: 35-67.

Watts, D.J. and S.H. Strogatz. 1998. Collective dynamics of 'small-world' networks. *Nature,* 393: 440-442.

Wiendahl, H.-P. and S. Lutz. 2002. Production in networks. *Annals of the CIRP,* 51: 1-14.

Wilkinson, I.F., J.B. Wiley and A. Lin. 2000. Modeling the structural dynamics of industrial networks. In: International Conference on Complex Systems. Nashua, NH.

Wilson, E.O. 1998. *Consilience: The Unity of Knowledge*. New York: Alfred A. Knopf.

Wright, S. 1982. The shifting balance theory and macroevolution. *Annual Review of Genetics,* 16: 1-19.

Wüthrich, H.A. and A. Philipp. 1998. Virtuelle Unternehmensnetzwerke. *IO Management,* 67: 38-42.

3 Flexibility and Re-configurability in Manufacturing by Means of Distributed Automation Systems – an Overview

Daniela Wünsch[1], Arndt Lüder[2] and Michael Heinze[3]

[1] SAP Research CEC Dresden, SAP AG, Chemnitzer Straße 48, 01187 Dresden (Germany)
e-mail: daniela.wuensch@sap.com

[2] Otto-von-Guericke University of Magdeburg, Faculty for Mechanical Engineering, Institute for Ergonomics, Manufacturing Systems, and Automation, Universitätsplatz 2, 39106 Magdeburg (Germany), e-mail: arndt.lueder@ovgu.de

[3] Otto-von-Guericke University of Magdeburg, Faculty for Mechanical Engineering, Institute for Ergonomics, Manufacturing Systems, and Automation, Universitätsplatz 2, 39106 Magdeburg (Germany), e-mail: michael.heinze@ovgu.de

Abstract The fast changing bordering conditions for industrial manufacturing systems have raised the need to increase manufacturing system flexibility regarding different types of flexibility. To enable this enhanced flexibility, manufacturing control systems must be changed resulting in new challenges which have to be tackled by management and engineers of the affected companies. In parallel, within information sciences new paradigms for structuring and implementing software systems must be developed which are also applicable to design and implementation of control architectures. This paper deals with the applicability of these new paradigms for structuring and implementing software systems to address recent challenges within the manufacturing industry. Therefore, the paradigms and challenges are described and mapped to each other.

3.1 Introduction

Fast changing economical conditions force companies based on production systems to reconsider their business models and reengineer production systems. Their basic drivers of change are

- The changing market conditions on nearly all sides including increasing customer power with increasing varieties of customer requirements regarding product quality and delivery,
- Increasing technological and technical production possibilities and useable production processes for comparable products with different economical bordering conditions,
- Fast changing raw material market conditions with sometimes highly volatile material costs, and
- Changing legal requirements such as environmental protection laws and labour laws.

All these needs drive companies to rethink their competitive advantages [1]. The main result of this rethinking process is the interest of companies in increasing their competitiveness by increasing their flexibility regarding attainable product features, useable technologies, and exploitable production resources (Fig. 3.1, [2]), and their adaptability to changing expectations regarding company embedding conditions.

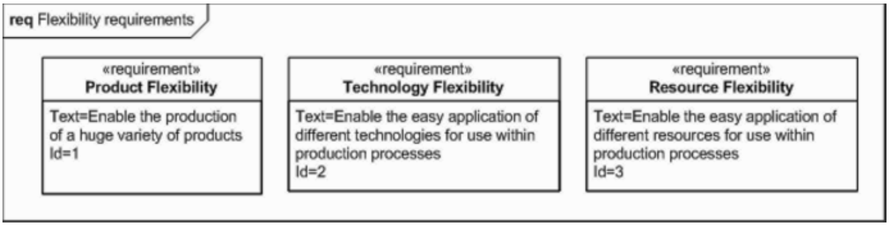

req Flexibility requirements
«requirement»
Product Flexibility
Text=Enable the production of a huge variety of products
Id=1
«requirement»
Technology Flexibility
Text=Enable the easy application of different technologies for use within production processes
Id=2
«requirement»
Resource Flexibility
Text=Enable the easy application of different resources for use within production processes
Id=3

Fig. 3.1 Flexibility requirements

Beneath the flexibility and adaptability companies are being forced to adapt cost cutting measures within all its activities. This affects the consumption of consumables and materials in the same way as the engineering process and the consumption of noneconomic objects like nature and air.

But to reach this flexibility, adaptability, and economic resource consumption totally new technologies and architectures on all levels of control are required. Several technologies and architectures have been considered in recent years all of which aim to increase special flexibility characteristics ranging from distributed control architectures, plug-and-participate technologies, Web services, virtualization, and much more. The application of most of these technologies signifies an important step forward towards flexibility, adaptability, and economic resource consumption. But they are not necessarily compatible with each other (sometimes the application of one technology contradicts the application of another). This results in inefficient structures of the complete control pyramid (each part has its own optimal solution but the combination results in huge problems), improper system behaviour (violation of temporal conditions, wrong interpretation of data, etc.), and, in the best case, suboptimal usability (doubling of data integration, bad business processes, etc.). Nevertheless, the issue of basic architectural structures and basic technological conditions enforcing a wide-ranging step forward in the direction of flexibility, adaptability, and economic resource consumption has still not yet been considered.

In this chapter at first we first intend to briefly introduce the three basic technology paradigms of object orientation, service orientation, and agent orientation that are the current candidates for improving manufacturing system control structures and architectures. Then, we highlight a set of recent production challenges occurring in line with the intended increase of production system flexibility, adaptability, and economic resource consumption. On this basis we want to evaluate the applicability of the paradigms as a response to the production challenges.

3.2 Current Technology Paradigms

Among the most recent technology paradigms applicable to addressing the challenges to be described are object orientation, service-oriented architectures, and agent-oriented architectures.

Object orientation (OO) was developed in the area of software engineering in the late 1980s and early 1990s [3]. It is a structuring and behaviour paradigm for systems underlying characteristics, knowledge, and rights of entities within the system. It was developed to provide means for modeling, analyzing, and implementing software systems resulting in the model sets of UML [4] and SysML [5], which can be used as a description basis for OO.

Since OO is a very powerful paradigm that is not focused on software design, it has been applied very quickly in other domains as well including control system design and implementation.

The main characteristics of OO is the definition of types of objects with a specification of possible object data, admissible object behavior, and usable object interfaces (visibility of data and behavior) and the definition of different types of dependencies among object types. Based on these types objects can be instantiated.

Thus, OO provides capabilities to inherit structures and behavior between different types of objects, to encapsulate structures and behavior, and to apply different objects without direct knowledge of their internal structure and behavior in a similar way. The basic concepts are also shown in Fig. 3.2. Detailed information can be found in Booch [6].

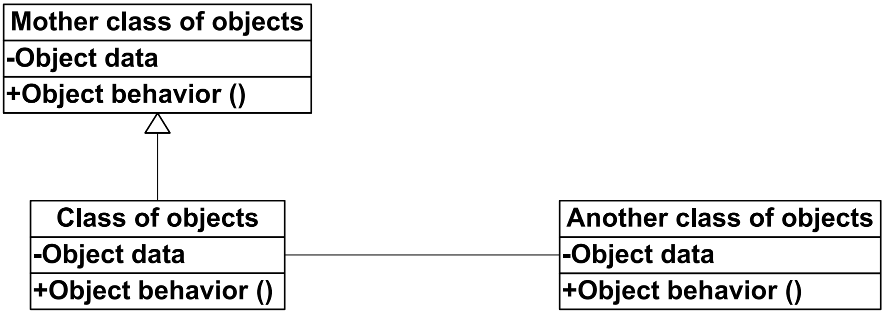


Fig. 3.2 Basic object orientation concepts

Within manufacturing system control OO can be easily exploited within the design and implementation of control applications by identifying relevant control entities and their control-relevant behavior [7]. Examples are the modeling of mechatronical units exploiting the hierarchy of objects with inheritance relations to design more detailed units or the analysis of dependencies within distributed control systems exploiting object dependencies [8, 9].

Based on the OO paradigm and riding on the wave of powerful IT devices, the Service-Oriented Architecture (SOA) paradigm has been developed. It is based on the ideas of Sculte and Natis from 1996 [10]. Nevertheless, usually SOA is defined as a structuring and behavior paradigm providing overall system functionality by exploiting the local functionalities of distributed entities in a coordinated way [11].

The main characteristics of SOA are the definitions of a service provided by a system entity and the rules by which this service can be exploited by other system entities (Fig. 3.3).

The provided service has to be accessible within a network of entities using standardized service interfaces. Thus, detailed knowledge about the internal behavior of the service is not required for the service user. In addition, the service implementation has to be independent from the service application. Usually, services are registered with some sort of yellow page service. Thus, services can be found and accessed at runtime of the system.

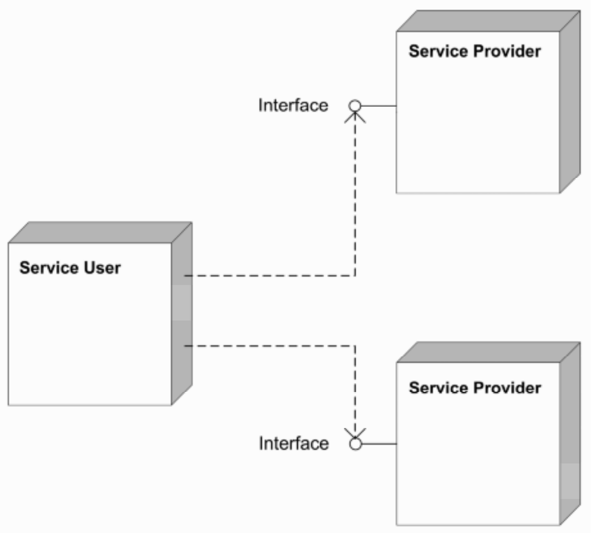


Fig. 3.3 Basic structure of SOA

The implementation of an SOA can be based on OO. Each service provider and each service user can be an object and the implementation and access to services can be based on OO mechanisms.

The use of SOA for controlling a system became relevant in recent years. Examples for its application are Web-service-based systems for device configuration [12] or Web-service-based interaction among companies [13].

Similar to SOA, Agent-Oriented Architectures (AOA) are based on OO paradigms. The main foundation of an AOA is the term of an agent. Despite various definitions of the term agent, agents are considered as independently acting entities with a dedicated environment model, agent internal aims, and the ability to act purposefully in order to reach the aim for the given behavior of the environment.

Within manufacturing system control, usually, multi agent systems (MAS) are used to enable the distribution of the control decision process among autonomous but cooperatively acting entities. There are several examples of AOA-based systems for control as given in [14, 15].

The main benefit of the application of AOA within control is the possibility to define appropriate encapsulations of control decision process parts and the explicit modeling and implementation of its interactions, as represented later within this book.

3.2.1 General Technology Application Ideas

The general idea of the application of OO, SOA, and AOA concepts in control is based on the application of mechatronical units within the design, implementation, and use of production systems. The main idea here is to compose production systems by mechatronical units in a hierarchical way [16, 17].

A mechatronical unit itself is a functionality oriented combination of mechanical, electrical, and control-related components providing functionalities to an overall system. It can be divided into a physical layer and a logical layer, as shown in Fig. 3.4, where the physical layer is responsible for the physical execution of activities necessary for the production process and the logical layer is responsible for the control of these activities.

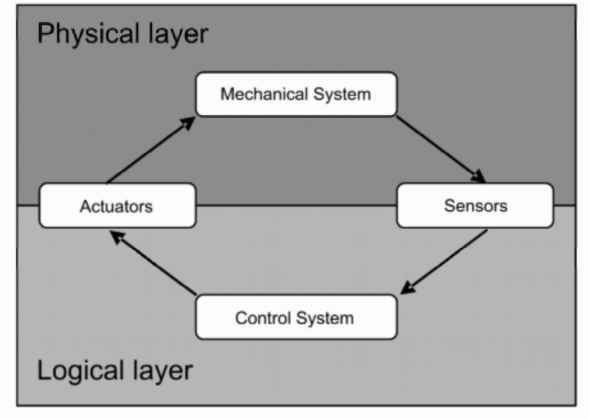


Fig. 3.4 *General structure of mechatronic systems*

The modeling of the behavior of a mechatronical unit within the design process as well as the implementation of its control part can be based on OO mechanisms. Here the object describing a mechatronical unit will provide production services or production support services to its environment. Thus, the use of OO encapsulation mechanisms enables the hiding of internal unit behavior resulting in a kind of white box behavior representation, the use of OO inheritance mechanisms enables incremental behavior enrichment of mechatronical units, and the use of OO polymorphism mechanisms enables similar usage of different objects within the control system.

In addition, SOA and AOA mechanisms can be used for the implementation of agile production systems consisting of mechatronical units. Therefore, SOA mechanisms can be used to enable plug-and-participate behavior of mechatronical units and dynamic binding of them within production, configuration, maintenance, and other processes. AOA can be used to model and implement self-aware and proactive mechatronical units acting independently but cooperatively within a system to fulfill its part of a common control system.

To enable the use of the mechatronical units within the production system additionally the order control needs to be modeled in an OO-oriented way by objects able to use the services and interfaces provided by the OO objects implementing/modeling the mechatronical units. These order objects will encapsulate order data and order execution control behavior. They will ensure application of SOA services or agent functionalities provided by mechatronical units (Fig. 3.5).

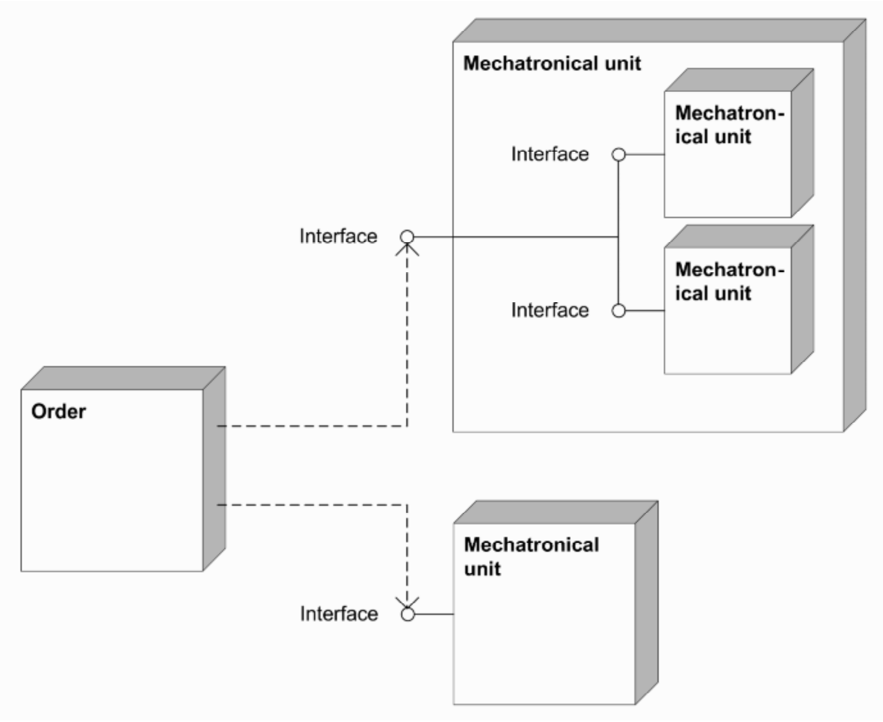


Fig. 3.5 SOA-based implementation structure of mechatronical units

3.3 Challenges in Production Control

As initially mentioned, production system control currently not provide efficient structures, proper behavior, optimal usability, and optimal integration in the overall environment. This results from different contradicting procedures. On the one hand most recent technologies are intended to be used in control systems without proper consideration of their applicability. On the other hand, the real needs of production system control are not sufficiently investigated. Finally, the technological, architectural, economic, and customer-oriented conditions of production systems, along with the conditions for production control systems, change very fast.

In the following discussion we will describe some of the recent challenges identified in Fig. 3.6 emerging from the latest developments and sketch how they can be addressed by the aforementioned basic technology paradigms. For a more detailed investigation of these challenges we refer the reader to [18].

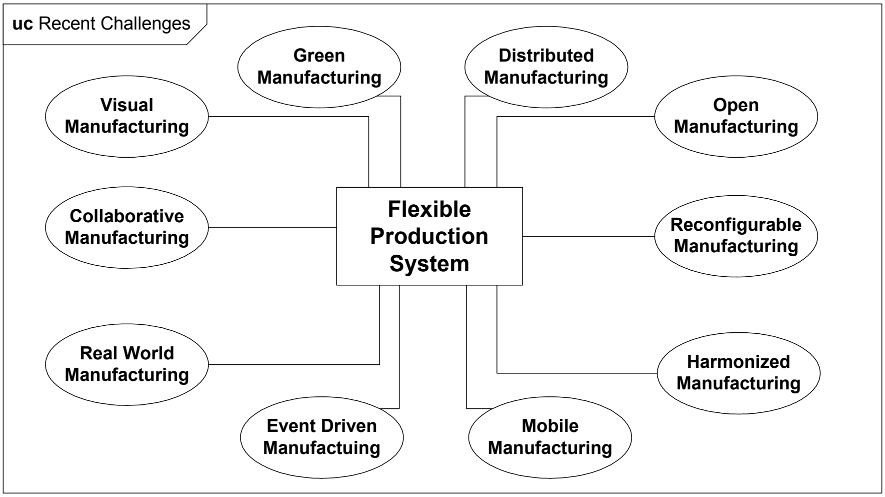


Fig. 3.6 Recent challenges

3.3.1 Visual Manufacturing

Many currently used systems within production system control (production confirmation applications, report generators, etc.) still do not consider the actual needs of the user. Often the problem is one of information overload or time-consuming manual processes to handle simple tasks.

The needs of users (in this case control engineers) requiring a proper system control interface must be met. Bad usability of the interface reflects poorly on the entire IT system. While usability problems will be overcome with time and require more iterative development work rather than groundbreaking research, other aspects of the user interface present more interesting challenges. Advanced visualization of the logistics processes within the plant is one opportunity for innovation. With today’s IT support it is often difficult to get an integrated and appropriately filtered view on the current status of the shop floor as needed in recent complex and agile production systems. Low-level control systems like Human-Machine Interface (HMI), SCADA visualize technical parameters and "business" data can be retrieved from Enterprise Resource Planning (ERP) systems, but the "big picture" must be tediously constructed by looking at different sources of data. This big picture may show the plant layout and highlight the status of resources (e.g. broken machine, in repair) as well as logistical data (e.g., Work in Progress (WIP) location, missing material/personnel warnings) (Fig. 3.7). The integrated view for resource data and logistical data must have strong filtering and abstraction mechanisms so as to not overwhelm the user accompanied by a mechanism to highlight interdependencies with suppliers and subcontractors, to navigate from the past (analyze past production) through the present (monitor current operations) to the future (simulate and visualize likely scenarios), and to represent different levels of detail.

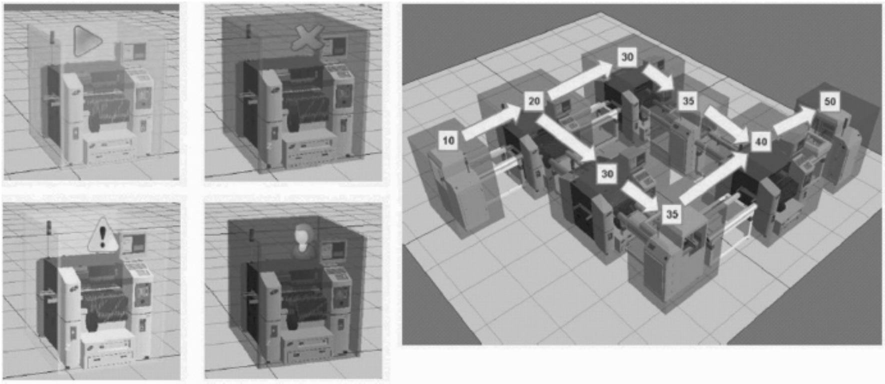


Fig. 3.7 Shop floor visualizations of resource status (left) and routing (right), realized with Visual Components 3DCreate [19]

To enable Visual Manufacturing, complete production resources will be considered as mechatronical units. Hence, production resource objects will be modeled and implemented with different object properties and behaviors encapsulated. Here, the necessary visualization properties and behavior can be integrated in the object behavior usable in larger applications. In addition, necessary data aggregation methods and behavior simulation means usable within these applications in objects can be integrated here.

3.3.2 Collaborative Manufacturing

Almost any production system consists of technical equipment and human personnel. A plant worker does not operate in isolation but is rather in a constant dialog with the machinery and coworkers [20]. He plans and executes actions, responds to exceptional events, and collaborates with coworkers to solve operational problems. Currently, only a few IT systems provide the structure and support to help plant personnel to efficiently communicate and collaborate.

To support the efficient work of plant personnel, systems are required that provide work support that is strongly knowledge based and production system and production system component functionality oriented like equipment failure reporting, diagnostic support, and maintenance assistance as well as problem solving strategy evaluation support.

In a highly networked world, collaboration is naturally not limited to the boundaries of one's own enterprise. Deeper collaboration with suppliers and subcontractors leads to better overall performance of the supply chain but requires data exchange models and security and privacy structures.

For Collaborative Manufacturing diagnosis functionalities can be integrated in the objects' modeling/implementing mechatronical units. These units can then provide diagnostic services to the overall system. In addition, object internal (and thereby mechatronical unit internal) data protection and access rules within the SOA implementation based on OO encapsulation mechanisms can enable company border crossing service applications.

Additionally, MASs can be exploited to implement collaborative manufacturing within the technical system by mapping mechatronical units to agents and implementing the control of a mechatronical unit within the agent and the collaboration of mechatronical units by agent interaction.

3.3.3 Real-World Manufacturing

State-of-the-art production control systems or MES (Manufacturing Execution System) / MOM (Manufacturing Operations Management) systems often do not support the manufacturing process according to real needs but frequently "over-engineer" solutions or, worse, address the wrong problem. Here the scheduling functionality of current MES or ERP systems are among the recent problems. As Pellerin et al. [21] have stated, most scheduling research and products are focused on highly sophisticated algorithms for building optimal production schedules but neglect the agility and required flexibility of real-world schedules. Hence, human planners or schedulers mostly do not use automated tools for schedule creation but rather need support for their highly manual and collaborative work of building and updating the schedules (usually more than one) by hand, evaluating what-if scenarios, and performing impact analyses. Planners often spend a large portion of their time on the shop floor ensuring the synchronicity of their schedules with the real process. Usually, they integrate into their schedule-generation processes engineering knowledge usually not available or not applicable in automatic scheduling procedures.

Scheduling systems should be redesigned to acknowledge these facts, become more flexible, and return the control of the scheduling process to the planner. The underlying problem of (unknowingly) ignoring the actual processes and only supporting the ideal ones needs to be addressed.

Generally, Real-World Manufacturing is an organizational and not a technical challenge. Hence, its elucidation requires organizational and management-based activities. Nevertheless, the aforementioned technologies may support this process. For example, in the case of scheduling aspects of the Real-World Manufacturing challenge, distributed scheduling mechanisms can be designed and implemented. These mechanisms may include the tracing of orders. They can be implemented by services provided by order objects of order agents and can be accessible by human operators. It gets possible to integrate resource-related scheduling functions customized to resource characteristics within the resource modeling/implementation mechatronical unit objects. Here the OO capabilities of polymorphism can be exploited. Additionally, expert systems can become integrated in these objects.

3.3.4 Open Manufacturing

Any standard software will fail to "win the hearts of the manufacturing staff" if it cannot be adapted to the particular needs of the company using it. Every shop floor is different if not unique. While standard processes and approaches do exist (like production order execution, confirmation, tracking & tracing, Kanban, etc.), usually they have very specific features that depend on the industry, the manufacturer's size, the level of automation, and many more individual characteristics forcing the user to adapt them to its special needs, making them unique. In order to address this heterogeneity, ideally every manufacturer should be able to design tailor-made applications.

Of course, growing costs prevent the implementation of all functionalities from scratch. Only a high level of reuse can guarantee a profitable solution. Therefore, reusable structures and functionalities have to exist and have to be implemented in a modular and block-oriented way. Ideally, these blocks would be based on open standards that would further facilitate ease of integration of best-of-breed solutions. The necessary reuse can be addressed by the hierarchy of mechatronical objects and its modeling/implementation by OO objects. Here each mechatronical unit in the hierarchy can be inherited from generalized mechatronical units as, for example, in [22]. Here, each mechatronical unit should provide customizable services accessible via standardized interfaces.

3.3.5 Reconfigurable Manufacturing

Flexibility of production systems is a necessary requirement to enable adaptability of the system to changing product, customer, or technology requirements. However, only reconfigurability ensures that production systems can be adapted after their initial design (see also [23]). This is what distinguishes Reconfigurable Manufacturing Systems (RMSs) from Flexible Manufacturing Systems (FMSs). FMSs provide different forms of flexibility for anticipated variations by their a-priori design. In contrast to this, RMSs provide adaptation capabilities when and where needed, i.e., "customized flexibility" delivered in a short time. They offer reconfiguration options at the hardware and software levels enabling the update of the system to face changing requirements until the complete change of the initial system.

Reconfigurability on a strategic level refers to a redesign of the overall manufacturing processes and system landscape. The Japanese "Kaizen" approach to continuous process improvement long ago revolutionized the way manufacturers develop their processes by benefiting from the creativity of their own personnel. Today, a number of tools exist to support the process of collecting and evaluating ideas. However, putting the ideas into practice is largely unsupported as current systems are not designed for change.

Hence, as above, structure and architectures are required that enable the easy integration of system changes in the running production system and, therefore, in its control system.

To attain Reconfigurable Manufacturing the classical modular architectures need to be improved. They only guarantee limited flexibility due to the complexity of single-application modules. In contrast to this, SOA facilitates the composition of customer-specific applications because the blocks of functionality are typically more fine-grained, clearly separated, and accessible as services. Ideally, the composition environment is targeted at domain experts such as production engineers and should not require much training. Reconfigurable systems based on services as described above can be adapted faster as the services are only loosely coupled. Similarly, AOA enables the composition of applications following the distribution of the control decision process resulting in structures similar to those of SOA with the main difference of a stronger focus on proactivity of control entities in contrast to the server behavior of services.

3.3.6 Harmonized Manufacturing

In order to enable smooth manufacturing operations, IT systems at the different layers of the automation pyramid (management layer, process control layer, field control layer) need to be synchronized. A series of standards exist to address this problem (e.g., ISA-95 [24] or OAGIS [25]) but these typically fall short of providing a comprehensive solution. Two major shortcomings are the synchronization of master data and of processes at the different layers. Both, the ERP system and plant-local MES/MOM system, need production master data such as resource information, routings, bills of material, and bills of operation to execute their tasks. Although these systems often need the same or similar data, they typically maintain their own master data, which tend to be difficult to consolidate and keep in sync. On the other hand, some of the data used by these systems are quite different. Therefore, simple data replication from one system to the other does not solve the problem. Rather, the different data models maintained by the ERP and MES/MOM need to be harmonized. Furthermore, the processes that operate on the data also need to be synchronized – a nontrivial problem in a distributed environment.

OO-based technologies may assist as a sound solution to this problem. Based on appropriate ontologies exploiting, for example, ISA 95 object structures can be developed handling the necessary data of the different layers. They can be adapted to the necessary data exchange among layers in an easy way reflecting the different data needs of the different layers.

The integration of these data objects within SOA-based communication architectures will enable the development of a sound and stable data exchange architecture for all levels.

3.3.7 "Green" Manufacturing

Processes and technologies should assist manufacturing enterprises to better monitor, manage, and optimize their material usage and energy consumption. For example, increased transparency and integration would enable major power consumers (e.g. steel, or paper industry) to adjust their production schedule to the changing price of the power and produce when energy capacities are in low demand (e.g., at night). Better quality control would reduce scrap or the need for rework. Optimized production schedules would reduce carbon dioxide emissions as resources that are important for environmental protections (e.g., ovens) are better utilized. The care for the environment is and is going to be one of the major challenges for the near future.

Like real Real-World Manufacturing Green Manufacturing requires management-based organizational intervention. Nevertheless, advanced technologies may support these interventions and provide a base for its technical implementation. For example, with the integration of monitoring services within mechatronical unit modeling/implementing objects, the requirements to support Green Manufacturing can be created. Here, the monitoring service search and application structures can enable the collection of data required for optimization. These services may be implemented in line with diagnostic services. The resource-related adaptation of the services and of the relevant optimization strategies can be reached by application of the OO concepts of inheritance and polymorphism.

3.3.8 Distributed Manufacturing

Virtually all current production control systems are centrally organized and tightly connected to the current state of the factory layout. Any change to the plant layout (e.g., adding new machines, changing the setup) typically requires considerable changes to the IT systems.

A decentralized approach could make production systems more flexible and adaptable. In a decentralized system, individual functionalities may be distributed to several entities that in turn may be logically or even physically distributed. Thus, each entity is able to act independently from the others and offer its services via standardized interfaces.

The implementation of Distributed Manufacturing systems can be based on the above mentioned distribution and encapsulation of control decisions within OO objects responsible for the control of mechatronical units. The control-decision-relevant data can be exchanged via interfaces using either SOA concepts or agent concepts.

3.3.9 Event-driven Manufacturing

A machine breakdown is perhaps the most prominent example of a significant event that can occur in a production environment, but it is by no means the only one. The completion of a production order, the occurrence of scrap, missing material, the rescheduling of a production plan, the press of a button on an HMI – these are more examples representing the multitude of events occurring on the shop floor.

For each event there may be none, one, or more interested parties, be it a machine, a production control system, or plant personnel. In many cases there are multiple interested parties per event. Most current production control systems do not have a clean and scalable event management architecture. Instead, a prominent approach is the data pull paradigm in which important data points are pulled in regular time intervals. This mechanism can cause considerable system load and is therefore in many cases a rather inefficient solution. Other “event mechanisms” like OPC DA subscription or OPC A&E, both currently being unified into OPC UA [26], offer event registration at the device level but do not provide any support for complex event processing. An example of a slightly more complex event that includes a temporal dimension is when one of three bottleneck resources becomes unavailable for more than 5 min. Even this basic event cannot be described by most current MES/MOM systems.

The development of event brokering services enabling event registration by event producers and event notification by event consumers as displayed in Fig. 3.8 can be used to solve the problem of Event-Driven Manufacturing. Here, the SOA can be exploited on the level of mechatronical units to design event provider services where event consumers register for events that will be transmitted only if they occur. The event provider services can be integrated with event monitoring engines within OO objects describing mechatronical units.

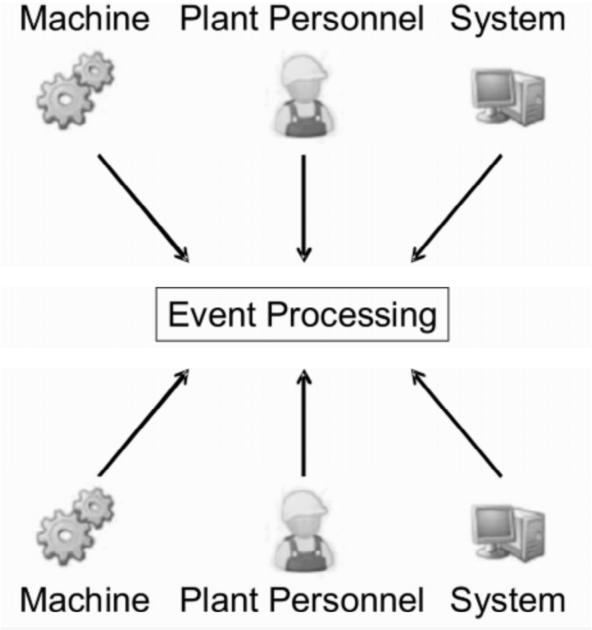


Fig. 3.8 Event processing engine for shop floor events

3.3.10 *Mobile Manufacturing*

Finding material or tools in a warehouse, performing maintenance and problem analysis on machines, configuring small devices without one's own user interface, and monitoring critical events are primary use cases for mobile and wireless technologies like PDA, wireless laptops, or cell phones.

While mobile applications have long found their place in manufacturing, the design of mobile and/or occasionally disconnected applications is still challenging.

Apart from hardware-related problems like battery life and the reliability of wireless connections, issues like the platform-independent design of a usable application or problems with data synchronization hinder further adoption. Multimodal applications (using voice, image, or even gesture recognition and other input/output modalities) increase the opportunities (e.g., enabling online support during machine maintenance) but also the costs.

Finally, the application of OO concepts enables the development of platform-independent programming and embedded networking capabilities as, for example, given in the programming technology Java[1] [27]. These platform-independent programming technologies and the agent technologies based on them like Jade [28] can be exploited to enable Mobile Manufacturing. Mobile devices can be implemented using a layered architecture including hardware, runtime environment for control applications, and hardware-independent control applications. In addition, OO concepts as developed for Distributed Manufacturing can be adapted to mobile devices enabling their on-demand integration in control applications.

3.4 Application Example

One application example of the above-described technologies for the solution of the mentioned problems is the PABADIS’PROMISE (PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises) approach developed in the EC project PABADIS’PROMISE [29]. One outcome of the PABADIS’PROMISE project is a highly generic control system architecture and control system design methodology enabling a stringent decision responsibility distribution among acting control entities modeled by OO objects. It distinguishes especially between a description of plant resources providing services and products requiring services and a generic unique executor able to read these data, to find the match between supply and demand, and then to control the production that is based on SOA mechanisms and OO mechanisms (depending on the layer of control). The proposed control architecture is thus characterized by a complete separation between system data and execution environment.

[1] Sun, Sun Microsystems, the Sun Logo and Java are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

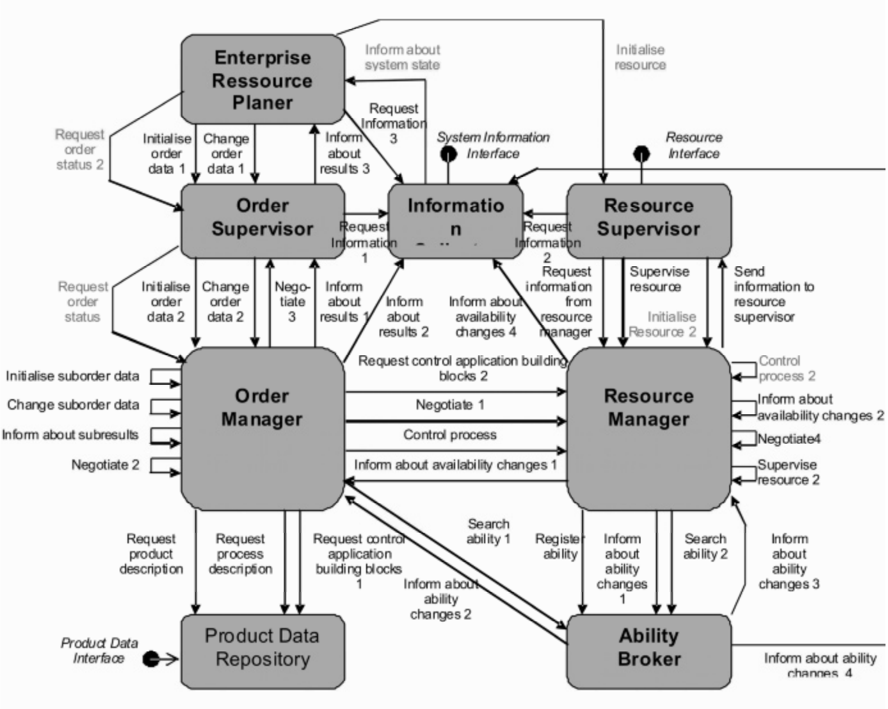


Fig. 3.9 PABADIS'PROMISE basic architecture

The PABADIS’PROMISE architecture itself (Fig. 3.9) is modeled by a set of high-level OO object entities containing the enterprise resource planner for the overall order, resource, and data management at the ERP level, the order supervisor for order execution initialization and supervision at the MES level, the order manager for order execution control at the MES level, the product data repository for product-related data provision at the MES and field control layer, the resource manager for resource control at the MES and the field control layer, the resource supervisor for resource supervision at the MES and the field control layer, the information collector for data collection at the MES and the field control layer, and the ability broker for manufacturing process execution ability announcement and search.

For the implementation of the different entities and their interaction mechanisms different OO-based technologies have been exploited. Thereby, a consistent structure has been defined applying the SOA approach at the ERP layer implementing the Enterprise Resource Planner and parts of the Resource Supervisor, Order Supervisor, Information Collector, and Product Data Repository, agent-oriented structures at the MES layer implementing Order Supervisor, Order Manager, Product Data Repository, Resource Manager, Resource Supervisor, Information Collector, and Ability Broker, and function-block-oriented structures at the field control layer implementing parts of the Resource Manager (Fig. 3.10).

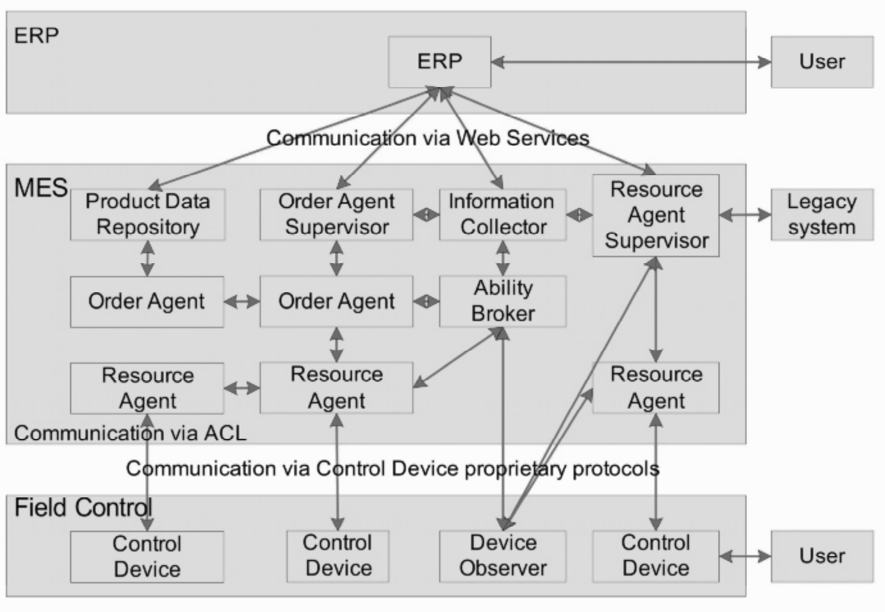


Fig. 3.10 PABADIS'PROMISE agent system architecture

The resulting structure of the PABADIS'PROMISE control system implementation integrates several of the aforementioned problem solutions successfully in the following way.

The concept of an information collector entity enables the implementation of *Visual Manufacturing, Green Manufacturing,* and *Collaborative Manufacturing.* The interentity communication between Resource Managers and Information Collectors and Resource Manager internal data collection and preprocessing enables an easy adaptable data provision to both systems while the information collector entities provide the necessary HMI for users.

The distributed scheduling mechanisms implemented in the PABADIS'PROMISE system result in *Real-World Manufacturing* structures.

The implementation of the MES layer based on the open source agent system Jade [28] has been published by the project on its Web page for further evaluation and extension. Hence, it follows the ideas of *Open Manufacturing.*

The plug-and-participate architecture for manufacturing resources controlled by Resource Managers as one example enables *Reconfigurable Manufacturing* since the Resource Mangers implement a special kind of SOA using agent interaction mechanisms to provide manufacturing services used by Order Managers.

In general the PABADIS'PROMISE architecture follows the ideas and characteristics of *Distributed Manufacturing.*

Since the plug-and-participate structure at the MES layer also involved event-based notification mechanisms for resource availability changes, *Event-Driven Manufacturing* is also covered by this architecture.

For more information on the PABADIS'PROMISE project results see [30].

3.5 Conclusions

In this chapter, three recently emerged software engineering paradigms have been presented. It has been shown how they can be applied as part of a strategy to tackle recent challenges related to factory automation.

Therefore, this chapter started with a description of the paradigms of object orientation, agent orientation, and service orientation and a consideration of their general application within industrial control exploiting the control of mechatronical units.

Then, ten major challenges were presented necessary to be considered to make industrial systems sustainable. They concern the impact of market, engineering, government, environmental protection, etc. combining technical and organizational challenges.

For each challenge it has been shown how the three paradigms can be exploited to successfully deal with the challenges.

In light of the results of this chapter it must be stated that structuring paradigms originating from information sciences and their application to mechatronical units will be an increasing trend within industrial automation and control.

References

[1] Kühnle, H. 2007. Post mass production paradigm (PMPP) trajectories. *Journal of Manufacturing Technology Management*, 18 (8): 1022-1037.

[2] ManuFuture Platform. 2005. *STRATEGIC RESEARCH AGENDA assuring the future of manufacturing in Europe*. Available from: http://www.manufuture.org/SRA_form.html [Accessed December 2005].

[3] Brügge, B. and A. Dutoit. 2004. *Objektorientierte Softwaretechnik.*, München: Pearson Studium, Prentice Hall.

[4] Booch, G., J. Rumbaugh, and I. Jacobson. 2005. *The Unified Modeling Language User Guide, The Addison-Wesley Object Technology Series*. Amsterdam: Addison-Wesley Longman.

[5] Weilkiens, T. 2008. *Systems Engineering with SysML/UML - Modeling, Analysis, Design*. OMG Press.

[6] Booch, G. 1997. *Object-Oriented Analysis and Design with Applications*. Addison-Wesley.

[7] Lüder, A., J. Peschke, and R. Sanz. 2006. Design Patterns for Distributed Control Applications, *ATP International,* 3: 32-40.

[8] Vyatkin, V., Z. Salcic, P. Roop, and J. Fitzgerald. 2007. Information Infrastructure of Intelligent Machines based on IEC61499 Architecture. *IEEE Industrial Electronics Magazine,* 1 (4).

[9] Thramboulidis, K. 2008. Challenges in the Development of Mechatronic Systems: The Mechatronic Component *(13th IEEE International Conference on Emerging Technologies and Factory Automation),* September 2008, Hamburg. Proceedings [online] Available from: http://seg.ee.upatras.gr/thrambo/dev/Papers/ETFA08paper.pdf

[10] Sculte, R. and Y. Natis. *SSA Research Note SPA-401-068: Service Oriented Architectures, Part 1and Part 2*. Gartner.

[11] Bieberstein, N., R.G. Laird, D. K. Jones, and T. Mitra. 2008. *Executing SOA – a Practical Guide for the Service-oriented Architect.* Upper Saddle River: Pearson.

[12] Neumann, P., A. Poeschmann, and R. Messerschmidt. 2008. Architectural Concept of Virtual Automation Networks *(17th IFAC World Congress Proceedings),* 6-11 July, Seoul.

[13] Papazoglou, M.P. 2007. *Web Services: Principles and Technology.* Essex: Prentice Hall.

[14] Shen, W., Q. Hao, H.J. Yoon, and D.H. Norrie. 2006. Applications of agent-based systems in intelligent manufacturing: An updated review. *Advanced Engineering Informatics,* 20 (4): 415-431.

[15] Wannagat, A., B. Vogel-Heuser, H. Mubarak, and P. Göhner. 2007. Evaluation of Agent Oriented Methodologies for the Development of Flexible Embedded Real-Time Systems in Automation. *ATP-International,* 1.

[16] Czichos, H. 2006. *Mechatronik – Grundlagen und Anwendungen technischer Systeme*. Wiesbaden: Vieweg Verlag.

[17] Habib, M.K. 2007. Mechatronics – a unifying interdisciplinary and intelligent engineering science paradigm. *IEEE Industrial Electronics Magazine,* Summer: 12-24.

[18] Rode, J. and D. Wünsch. 2007. A Research Agenda for Adaptive Manufacturing. *(12th IEEE Int. Conf. on Emerging Technologies and Factory Automation Proceedings),* September, Patras.

[19] Visual Components Oy, 3DCreate. 2008. *Bring 3D CAD Data to life*. Available from: http://www.visualcomponents.com/index.php?id=56.

[20] Bergmann, U. 2008. *Kommunikation als Optimierungskriterium: ein Beitrag zur Systematisierung der Layoutplanung von Produktionssystemen*. Thesis (PhD). Otto-v.Guericke University Magdeburg.

[21] Pellerin, R., P. Batiste, J. Robert, S. Guessous, M. Guevremont, J. Proulx, and B. Saenz De Ugarte. 2007. Lean scheduling – case studies of 10 manufacturing plants, Ecole Polytechnique Montreal. *Internal report for SAP Research.*

[22] Drath, R., A. Lüder, J. Peschke, and L. Hundt. 2008. An evolutionary approach for the industrial introduction of virtual commissioning. *(13th IEEE International Conference on Emerging Technologies and Factory Automation Proceeding-CD).* September, Hamburg.

[23] ElMaraghy, H. 2005. Flexible and reconfigurable manufacturing systems paradigms. *International Journal of Flexible Manufacturing,* 17: 261-276.

[24] The Instrumentation, Systems and Automation Society. 2000. *ANSI/ISA-95.00.01-2000: Enterprise Control System Integration.*

[25] Open Applications Group. 2007. *Open Applications Group Integration Specification (OAGIS).* Release 9.1 Available from: http://openapplications.org/oagis/9.1/index.html.

[26] OPC Foundation. 2007. OPC Unified Architecture. Available from: http://www.opcfoundation.org/Default.aspx/01_about/UA.asp?MID=AboutOPC.

[27] Savitch, W. and F.M. Carrano. 2008. *Java – An Introduction to Problem Solving & Programming.* Pearson International Edition 5th ed. Prentice Hall.

[28] Bellifemine, F., G. Caire, and D. Greenwood. 2007. *Developing multi-agent systems with JADE*, Wiley series in agent technology, Wiley publisher.

[29] Lüder, A. and J. Peschke. 2007. Order oriented manufacturing control: the Pabadis'Promise approach. *Automazione e Strumentazione,* November: 84-93.

[30] PABADIS'PROMISE consortium. 2008. PABADIS'PROMISE project homepage. Available from: www.pabadis-promise.org.

4 Collaborative Virtual Environments and Immersion in Distributed Engineering Contexts

Marc Pallot[1] **and Ulf Bergmann**[2]

[1] Centre for Concurrent Enterprise, Nottingham University Business School, Jubilee Campus, Nottingham, NG8 1BB (United Kingdom), e-mail: Marc.Pallot@ESoCE.net

[2] Otto-von-Guericke University of Magdeburg, Faculty for Mechanical Engineering, Institute for Ergonomics, Manufacturing systems, and Automation, Universitätsplatz 2, 39106 Magdeburg (Germany), e-mail: ulf.bergmann@ovgu.de

Abstract Distributed Manufacturing is mostly associated with computing features and sophisticated software logics rather than with working environments concurrently used in distant locations. Nowadays, the global success of Web2.0 and, more specifically, social networking web applications are quite obvious. In fact, information and communication technology (ICT) users are creating web content, applications and online role-playing games with their own activities and related data in using web applications such as eBay®[1], Facebook[2], Second Life[3] virtual environment, and World of Warcraft[4]. In manufacturing, interests are quite different and therefore other solutions should contribute to overcome collaborative distance factors that are impeding an effective and efficient distributed collaboration. Using important existing options in this field, this chapter presents an overview on key developments and web applications in the area of distributed computing systems to support effective and efficient collaborative work carried out by different groups of people in different organisational units or companies with a strong focus on engineering communication.

[1] eBay® is a registered trademark of eBay Inc., 2145 Hamilton Avenue, San Jose, California 95125, http://www.ebay.com

[2] Facebook is a registered trademark or trademark of Facebook, Inc. and its corporate affiliates in the U.S. and/or other countries, http://www.facebook.com

[3] Second Life is a trademark of Linden Research, Inc.

[4] World of Warcraft is a trademark and Blizzard Entertainment is a trademark or registered trademark of Blizzard Entertainment in the U.S. and/or other countries.

4.1 Introduction

Any kind of collaboration in manufacturing becomes more effective as soon as information and communication technology (ICT) support is deployed. At the beginning, the emphasis was on data storage and data processing; later computer networks and finally the Internet gradually added more and more communication capabilities. Meanwhile, broadband communication networks have allowed doing all kinds of communications both synchronous and asynchronous. With this premise, individuals involved in an organisation can intensively communicate in all dimensions without necessarily meeting physically or being located close to each other (collocated at the same office, floor, building or geographical site). On the other hand, people from different places, with various backgrounds, capabilities, skills and knowledge may emerge as somewhat ad-hoc organisations (communities) after they have activated essential motivational quantities and agreed upon powerful communication technologies as well as (network) rules and standards. However, most of the established methods and procedures to communicate and to build up knowledge have been developed without taking into account these technological options.

The technologies in question are also summarised as distributed information systems. Of course, much inspiration for Distributed Manufacturing originates from distributed computing or distributed information systems, which are usually abbreviated as distributed systems. Purely technically speaking, distributed systems are networks of independent communication systems (computers or human actors) that enable electronic data exchange at a range of locations. Thus, information and communication processes will be maintained by database administrators under consideration of specified standard arrangements.

Two characteristics of communication processes in means of collaboration between different actors of different companies by using distributed systems may be differentiated:

- Distributed control[5] in virtual automation networks–mostly based on industrial communication[6] (machine-to-machine interaction)
- Distributed engineering[7] in collaborative virtual environments–mostly based on business communication; especially engineering communication[8] (human-to-human interaction)

It is currently anticipated that people will no longer work mainly individually but rather more as dynamically assembled groups of diverse and complementary skilled professionals working together within an enhanced collaboration environment. Therefore, the combination of social, intellectual and emotional capital will become the leading ingredient. As already demonstrated by the massive usage of social web applications, engineers will also spend more time in people-networking-like activities than ever. [5]

One approach is the gradual implementation of ICT in distributed structures, where research attempts to provide people with better and better communication methods, eventually arriving at a stage where people want to rely on these communication tools rather than on conventional ways. This would remove most of the communication distance factors and make all communication and knowledge building independent from geographical locations.

Thus, the focus of this contribution is on engineering communication processes, their main characteristics and the different aspects in their technical support. Classifications and capability profiles are given to be used as starting points for company- or network specific- set-ups of Distributed Manufacturing.

4.2 Related Theories – Terms and Definitions

4.2.1 Collaborative Distance

Wilson et al. propose two categories of collaborative distance, namely objective distance and subjective distance [6]. They argue that past research has mainly focused on objective distance (e.g. geographic distance), which produced conflicting

[5] Distributed control (DC): A system whereby control processing is decentralised and independent of a central computer [1]

[6] Industrial communication (IC): IC or Industrial Ethernet, is designed to maintain control of a production process while monitoring many production items traditionally relegated to the analog world [1]

[7] Distributed engineering (DE): Collaboration between different human individuals (i.e. engineers) and/or data processing devices (any technical communication systems) by means of engineering processes such a plant design process [2]

[8] Engineering communication (EC): A simple definition of engineering communication can be described as follows:

- EC is the flow of intelligence from one mind to another [3] … provided …
- the successful transmission and exchange of information between sender and receiver [4]

and inconsistent findings regarding the relationship between distance and a variety of important processes and outcomes (e.g. team communication and performance). They define subjective distance as individuals' cognitive and affective representation of the distance between them and their team members and argue that it is likely to predict important team outcomes better than objective distance.

O'Leary and Cummings argue that objective (geographical) distance has spatial, temporal as well as configurational elements [7]. These three elements represent respectively physical distance, time zones or working hours and team members' patterns or arrangements (e.g. roles, resources, power and status). Hence, communication and identification are key mediating processes by which subjective distance emerges. These processes are affected in important ways by several types of factors, including individual (openness to experience, need for affiliation, experience with dispersed work, technology and travel), social (perceived similarity, status differences, role centrality), task (interdependence), and organisational (structural assurance and culture) factors [7].

Pallot proposes to adopt "standardised" dimensions of collaborative distance without respect to the observed area: structural, social, technical and legal [8]. This exploration field is tentatively named "collaborative distance", inducing a balanced observation of any collaboration case along these four dimensions as a kind of reference framework including a holistic view of factors affecting collaboration. Categorised types of distance or proximity allow one to make various measurements that could then be combined into a single overall indicator of collaborative distance (Fig. 4.1).

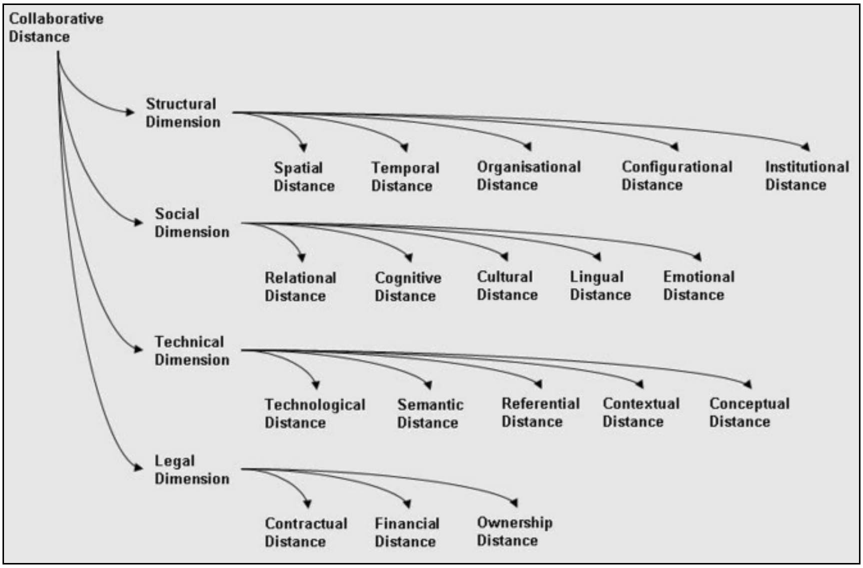


Fig. 4.1 Collaborative distance dimensions and respective distance types [8]

Various authors may come up with different types of proximity depending on whether they have considered collaboration among individuals or teamwork as distant collaboration or nearness collaboration. Presently, the phrase “collaborative distance” is used to reflect the impact of distant collaboration among individuals or organisations, though organisations do not collaborate as such, only people have the capability to collaborate [5].

4.2.2 Information and Communication

Every kind of successful development of a company is based on the intra- and interorganisational acquisition of knowledge, the development of knowledge, and its economic realisation. Basically, knowledge can be characterised by two features. On the one hand, knowledge represents information provided directly by the brain or information that can be retrieved quickly from memory available to human beings. On the other hand, knowledge is related to the context of the acquirer. People must be able to display their information directly by means of a model-like illustration of realistic relations, conditions and procedures – referring to means of intermediation (i. e. language) – in such a way that the information is purposeful and can be applied in a practice-oriented context [9].

People obtain knowledge by means of purpose-related handling and application in two fundamental acquisition modes:

- Original acquisition through one’s own experience
- Derivative acquisition of knowledge by communication

In this respect, communication comprises all relations and tools of communication both within and outside a company, serving as the interface with human individuals or data processing devices. Communication is characterised by influences that organisms exert on each other with the intention of purposeful and targeted exchange of information referring to a common interaction space (i.e. the operational task). Various kinds of communication processes may be differentiated according to key criteria [10] and their respective technical support:

- Direction
  unilateral vs. bilateral (multilateral)
- Reliability
  high vs. low
- Synchrony (feedback)
  synchronous/asynchronous (directly/indirectly)
- Complexity (kind and quality)
  complex vs. simple (rich vs. poor)

The very base of communication is information, which is defined as a purpose-related extract for application issues, aiming at the solution of a problem. Any amount of information may become knowledge if it is

1. Clustered and purposeful,
2. Specifically directed towards an issue,
3. Processed in such a way that it will be generally accepted, checked and transferred.

Furthermore, non-purpose-related, latent issues are also labelled as information since this kind of information could, at any time, be combined or condensed into knowledge by means of purpose-related references. Information therefore constitutes projections of the environment surrounding us and the various components and procedures used by humans for self-orientation within the world in order to recognise it and gain influence [11].

Information can be differentiated according to the following key criteria [10]:

- Derivation
  (internal vs. external)
- Formalisation
  (formal vs. informal)
- Usability
  (special vs. general)
- Quantifiability
  (quantifiable vs. non-quantifiable)

Distributed production emphasises human-influenced complex networks as well as planning and execution of processes in networks.

While productivity of individual work has increased considerably for years by ICT, very few efforts have been done in terms of interpersonal productivity.

Individual productivity is still considered as the Holy Grail by most industrial companies. Such companies do not consider social interaction as a vital activity for a business organisation even if social interaction has been widely demonstrated as the main source of knowledge creation.

Especially by using interpersonal interactions, groups of people create new knowledge in this way. Such knowledge will lead most probably to new emerging concepts if they are able to reach the proper level of consciousness [5].

4.3 Collaborative Virtual Environment (CVE) – Technologies

In general, in a collaborative virtual environment (CVE), engineers using the Internet extensively (so-called eProfessionals) can work together regardless of geographical location by sharing information and exchanging views in order to reach a common, shared or mutual understanding. Therefore, different CVE applications have been developed, and a selection will be sketched that seems to be relevant for Distributed Manufacturing.

The most important devices are described and different CVE technologies characterised in Fig. 4.2.

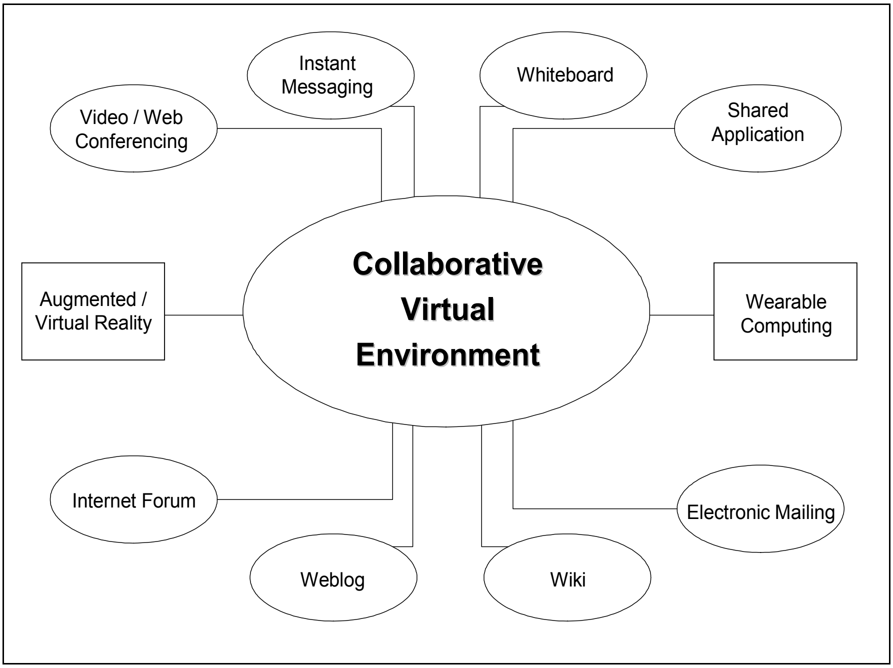


Fig. 4.2 Recent CVE technologies

According to the above-mentioned criteria for describing of communication processes, radar figures illustrate the suitability of all mentioned CVE technologies (cf. Figs. 4.3–4.12). Note that the choice of a specific CVE technology directly depends on the preferences and skills of the collaborating persons.

4.3.1 Video Conferencing and Web Conferencing

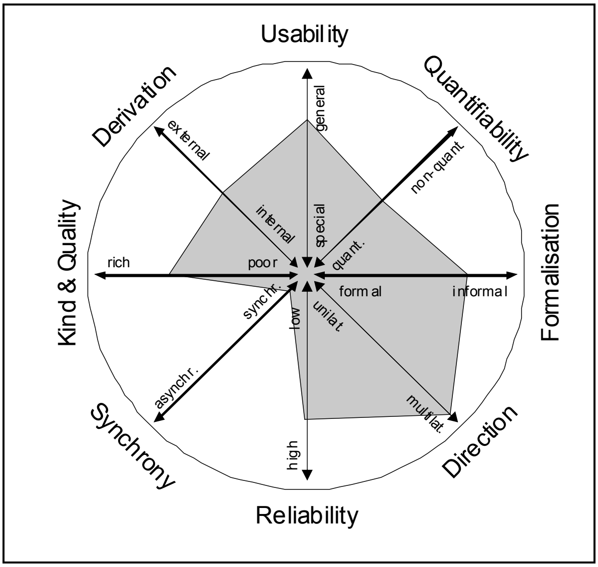


Fig. 4.3 Radar chart of video conferencing systems

Video conference systems were brought out by telephone providers as soon as the available bandwidth was broad enough to transfer image sequences with a reasonable spatial and temporal resolution. Due to the necessity of special communication hardware, the success of commercial video conference systems has been rather limited. However, approaches have been developed in various research projects to overcome the spatial distance of participants. Thus, visual and acoustic effects were often used to provide users with the impression of local presence of the other participants.

With broader bandwidths, video conferencing has become available on PCs. This is even possible using affordable hardware (e.g. webcams), on widespread software standards (e.g. MPEG), and Internet protocols which simplify the development of applications which eventually allow the easy integration of other communication channels [12].

4.3.2 Instant Messaging and Chat

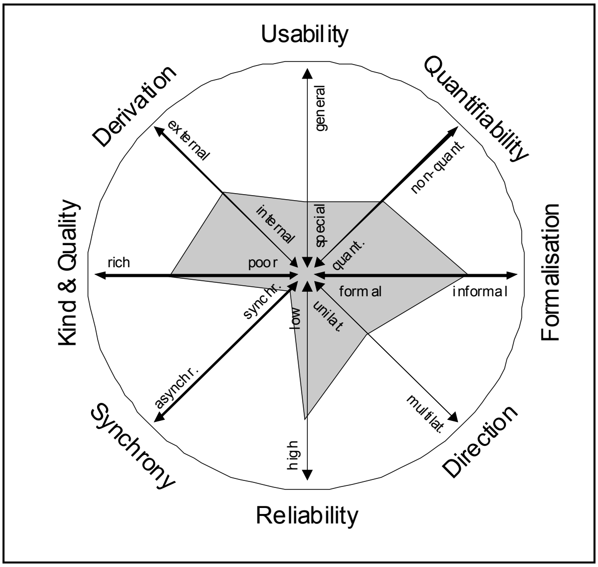


Fig. 4.4 Radar chart of instant messaging systems

One of the first known instant messaging systems was Unix Talk, which provides textual communication between two clients. The talk tool splits the screen in half, with one half used to write messages, the other to read the messages of the other client. Typed messages are immediately transferred to the remote client (unlike e-mail). This concept may be enhanced to obtain so-called chats where more than two users may be involved. Moreover, chats are characterised by textual communication with rather small time lags where attachments of other media are possible. Users are represented by nick names (not necessarily real names) often combined with an individual icon. Messages of individual users are added to a public list and may be read by all users. Additionally, most chats provide the possibility to manage private channels where messages are only transferred to select recipients. Well-known systems are e.g. Internet Relay Chat (IRC) and ICQ [13].

4.3.3 Whiteboard

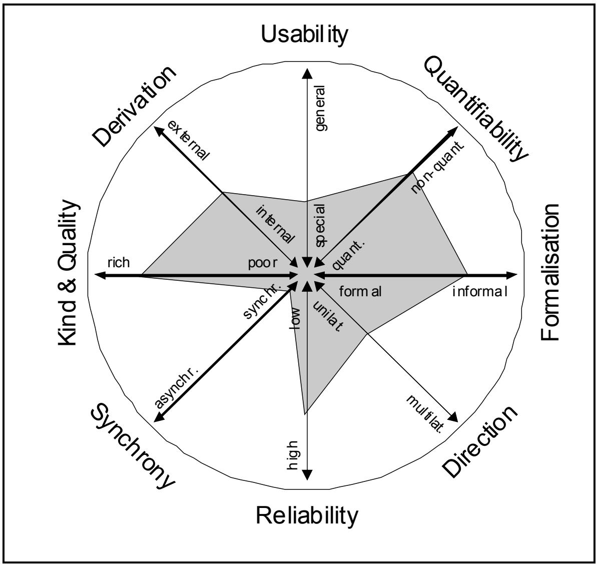


Fig. 4.5 Radar chart of whiteboard systems

Whiteboard systems employed the metaphor of an actual physical whiteboard. A large touch screen is used to display images and recognise inputs by a pen or finger. Combining these techniques with a network transfer facility allows remote users to simultaneously view and edit sketches. Smart boards are another further development of whiteboards. A smart board processes user input strokes as gestures to create data items with added context (sophisticated CAD software programs use similar techniques in their sketch input tools).

Whiteboard systems are generally integrated in video conferencing software, instant messaging software, or web portals rather than offered as standalone applications [14].

4.3.4 Shared Workspace and Shared Application

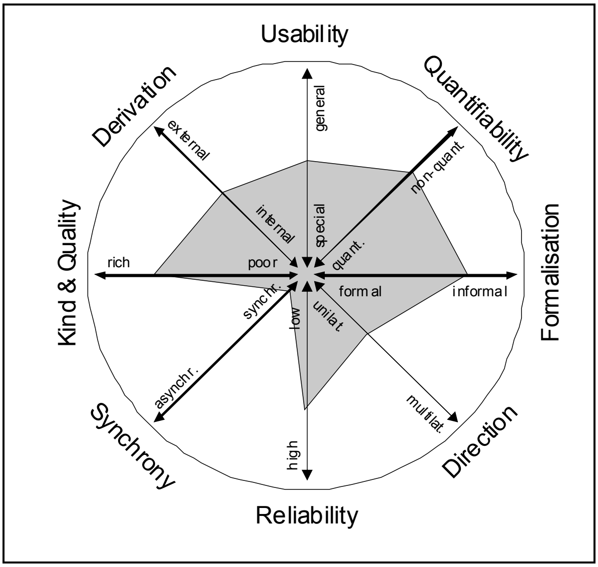


Fig. 4.6 Radar chart of shared workspaces

Database management systems (DBMS) are often used to manage a large amount of data on a central server. The data can be accessed by multiple clients (PCs or terminals with client software) concurrently and quasi-simultaneously (the DBMS server ensures the consistency of the database e.g. by preventing write access collisions). In a shared workspace, the managed data consist of content object files (i.e. documents, images, blog entries, links). The management of groups and user authorisations is part of the system. The clients are often web-based so as to be system independent and use the web infrastructure. Thus, work groups can share project content objects independent of spatial distances between members. A well-known example is the basic support for cooperative work (BSCW) shared workspace system. Shared workspaces are also often used for e-teaching. Shared workspace systems also allow one to dramatically reduce the flooding of e-mail inboxes in terms of both messages and attachments.

A shared application is a piece of software that enhances collaboration, where two or more people are allowed to view, modify and add information simultaneously. Shared applications provide a central management of data processing facilities as well as of data [15].

4.3.5 Internet Forum

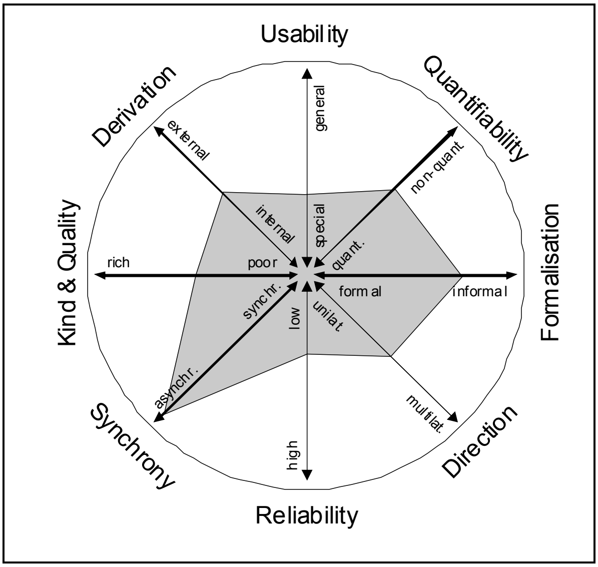


Fig. 4.7 Radar chart of internet forum system

An Internet forum is an online discussion platform managing user-generated content (e.g. feelings, arguments, experiences) with more than one content level. There are two major types depending on the structure of these levels.

The terms forum and board may refer to the entire community or to a specific subforum dealing with a distinct topic. Messages or postings within these subforums are then displayed either in chronological order or as threaded discussions (lean structure of threads). Postings within a classical Internet forum are subject to a strictly hierarchical topic structure (complex structure of topics and threads).

The main advantage of Internet forums lies in their comprehensive data-management concept. In principle, every posted message will be archived. Along with the pull triggered data exchange, an Internet forum represents an excellent data mining platform (cf. the e-mail communication: waiting for response). Depending on the level of reliability, potential users must be reckoned with limited access to Internet forums.

One of the first worldwide distributed web-based discussion systems was Usenet. Users can read and post public messages (postings) to one or more categories, known as newsgroups. A so-called newsfeed informs the user about the latest news or new postings. For this reason users need a newsreader, a small software application that has become a standard add-on in the latest e-mail-software solutions.

Internet forums highlight a central issue of web-based communication. Text-based publications take place in that undefined space between speech and writing. Hence, differences in the communication situation will lead to special linguistic usages that only the forum's community will understand.

Similar to the characteristics of electronic mailing two main disadvantages must be considered: the problem of authenticity and data manipulation attacks, especially in open forums with open threads [16].

4.3.6 Weblog

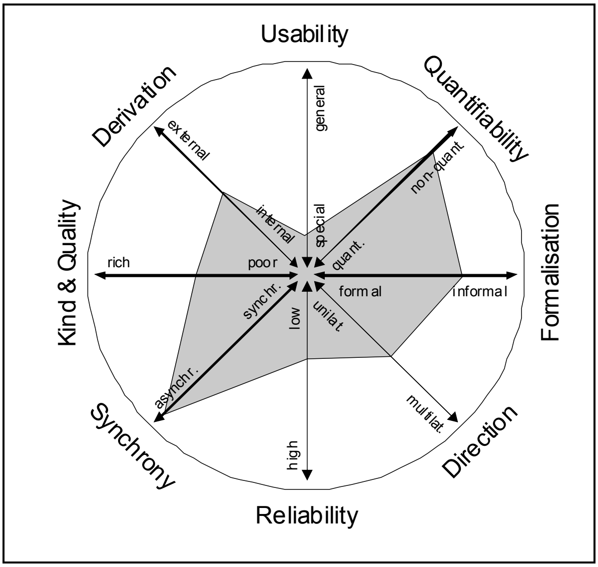


Fig. 4.8 Radar chart of weblog systems

A weblog is a hierarchy of text, images, media objects and data, arranged chronologically, that can be viewed in an HTML browser. Weblogs are unique in that they are the venue in which people can present their ideas without interference from others. They offer the public writer a kind of relaxed form that is not found in other forums. In general, in a weblog no one can change what the weblog owner wrote [17].

Unlike Internet forums, a weblog is a website including only one content level. Thus, a weblog has a simple structure. In conjunction with an easy-to-use content-management system, publishing is quite simple.

Many people use this sort of no-standard publishing to talk about new ideas and new approaches. Two other aspects are responsible for the rapidly rising number of weblog users (so-called bloggers): a low-cost communication system where users can publish “any time any place” (even via mobile phone: micro-blogging services like twitter). Along with the risk of false reports due to missing post-editing functions, infringements of copyright are rife. Furthermore, there is a high potential to become a victim of profiling (in terms of being spied on) if the weblog is fully public [18].

In contrast with public weblogs, a new weblog application, known as group blogging, is emerging where a group of people can edit blog entries that can be read by either the public (anonymous users) or a restricted group of people. For example, BSCW allows users to create a blog content object restricted to the shared workspace members that could be used as a private “project blog”.

4.3.7 Wiki

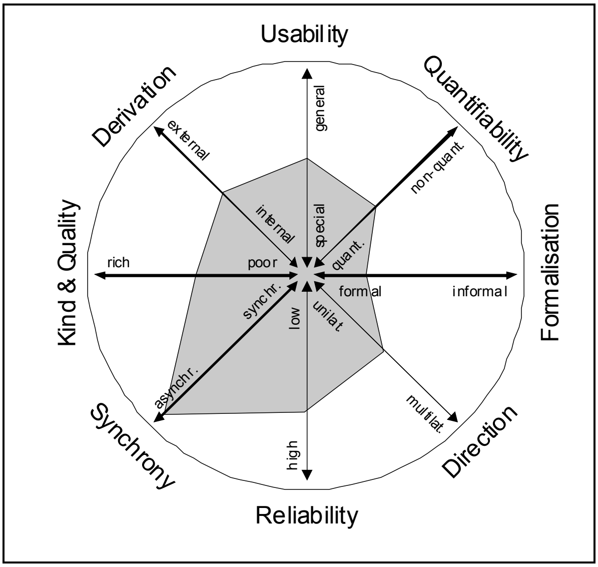


Fig. 4.9 Radar chart of wiki systems

A wiki is a collection of collaboratively edited web pages designed to enable anyone to accesses and to contribute or modify content using a simplified markup language. Wikis are often used to create collaborative websites and to power community websites [19].

A typical feature of wikis is the systematic linkage they create between individual publications without any wilful multiplication of information. These distinctive features are provided by an easy-to-use CMS. Thus, a wiki is a weblog-like system that allows anyone to edit anything and therefore represents an interesting mix of many views, not only one view of a single person [17]. The problem: How many views are mixed together and who wrote the text – a professional expert or a non-professional?

However, looking at the current success of Wikipedia®[9], one may think about the power of "collective intelligence". Furthermore, Wikipedia® was recently rated higher than traditional encyclopaedias.

4.3.8 Electronic Mailing

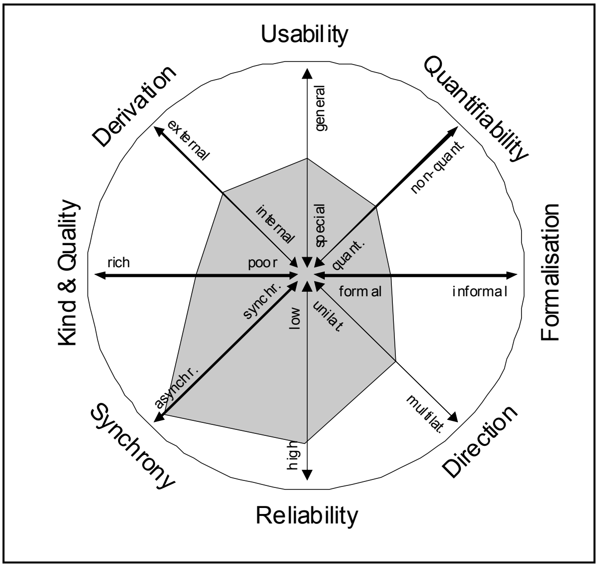


Fig. 4.10 Radar chart of e-mail systems

Electronic mail, or e-mail, is any method of creating, transmitting or storing primarily text-based human communications by digital communication systems. Initially, a variety of electronic mail system designs evolved that were often incompatible or not interoperable.

The latest e-mail systems are based on a store-and-forward model in which e-mail computer server systems accept, forward or store messages on behalf of users, who only connect the e-mail infrastructure with their personal computer or other network-enabled device for the duration of message transmission or retrieval to or from their designated server. Rarely are e-mails transmitted directly from one user's device to another's. Thanks to digitally based information such applications fully guarantee fast data transmission and the easy handling of these data [20].

[9] Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., P.O. Box 78350, San Francisco, CA 94107-8350, USA, http://wikimediafoundation.org

The main problem with using e-mail is typical in an asynchronous communication modus operandi: no direct response. Furthermore, disadvantages include the problem of authenticity and of possible phishing attacks. However, e-mail is the only standardised collaborative tool so far. Users do not need to know in advance whether the receivers will be using the same e-mail tool. This is the main reason why e-mail is still so successful no matter what new exciting features other collaborative tools try to offer. It also explains why e-mail has become like a garbage collector, where every user tries to send out everything he can grab. In the end, it has led to the flooding of everyone's e-mail inbox, which makes it hard to identify what is urgent and what can be deleted.

4.3.9 Virtual Reality and Augmented Reality

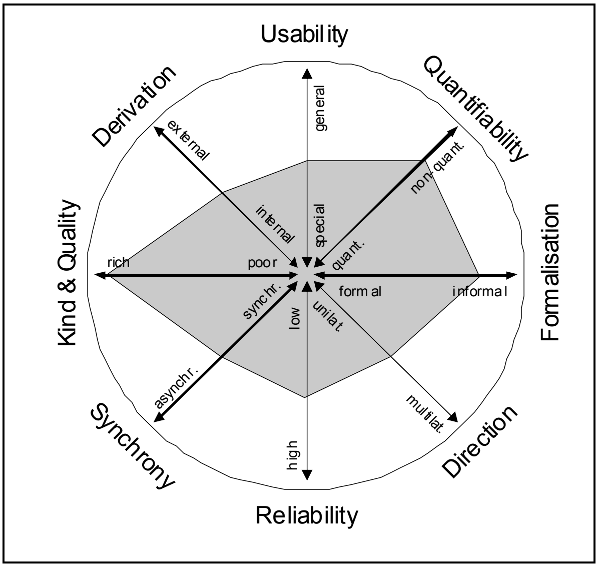


Fig. 4.11 Radar chart of VR/AR systems

Virtual reality (VR) describes a computer-generated appearance which captures selected senses of the user replacing real perception by artificial impressions. The user has the impression of residing in the VR. Three features are essential for VR: imagination, interaction and immersion. Imagination is necessary to close the gap due to the quality differences in real-world perception and the perception of the virtual environment, which is a simulation model and must disregard multiple properties of reality. Interaction makes the user an active part of the VR instead of an uninvolved bystander. Immersion describes the degree to which the user's senses are captured by the VR. Current VR systems are focused on visual and acoustic perception for two reasons. They cover ca. 80 to 90% of human perception, and hard- and software for 3D image plus sound generation are widespread even in the field of home computing [21, 22]. Where VR systems originally were used for flight simulators and military battlefield simulation today it is also used for scientific visualisation and home entertainment (computer games).

Augmented reality (AR) is a further development of VR. In AR the user is not captured against real perception. Instead the (perception of) real world is enhanced by the VR. Current AR systems enhance vision with textual and graphical data using semitransparent displays. The first applications were the so-called head up displays for military pilots. Other potential applications for AR are the subject of research, e.g. in architecture (appearance of future buildings in a real landscape), in engineering (assistance for workers in assembly) [23] and in maintenance.

Whereas human-machine interaction is an essential feature of VR, modifications can also be made for human-human interaction. In the scientific world this is called a collaborative virtual environment (CVE); in the field of home entertainment, it is better known as multi user dungeon. The synchronisation of the local VR models depends on the availability of a powerful network infrastructure [24].

The first applications were in the military field, whereby the distributed interaction simulation (DIS) standard was established. Another science-related development is the high level architecture (HLA).

4.3.10 Mobile and Wearable Computing

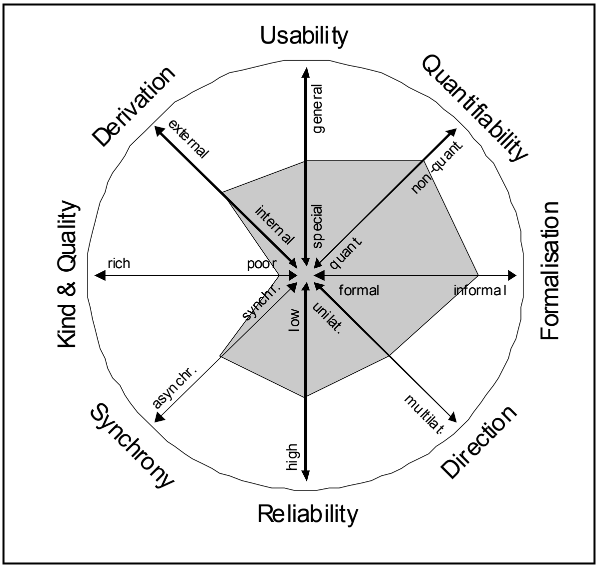


Fig. 4.12 Radar chart of wearable computing

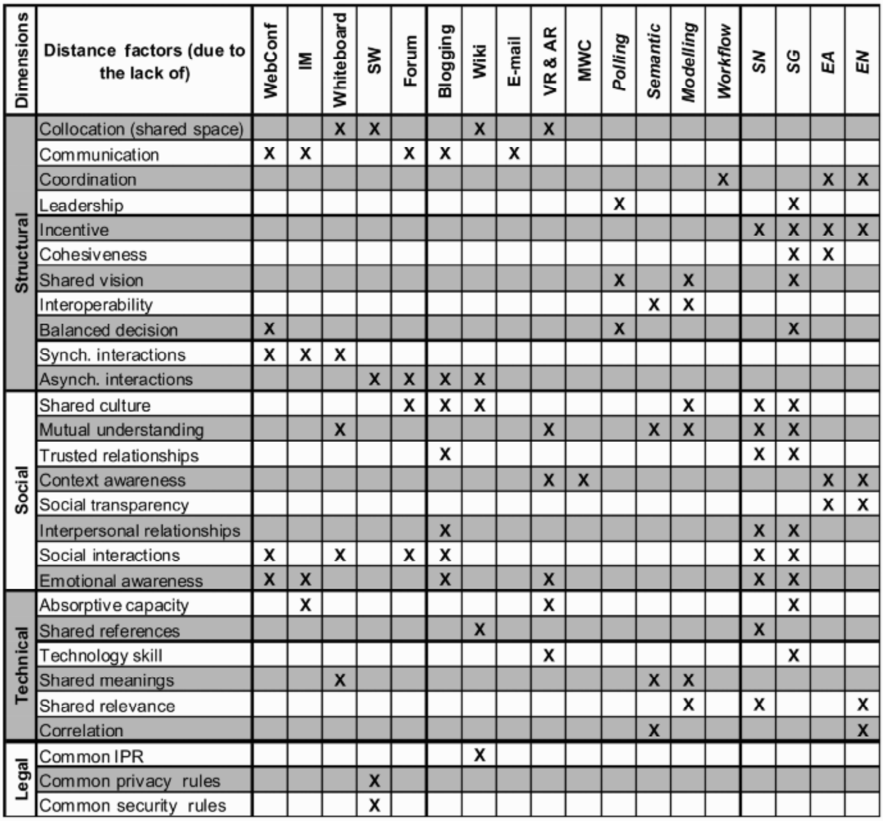
Mobile computing is based on computers which can be used independently of an office desk. The improvements in miniaturisation and in decreased power consumption madeconsumption appear in minicomputers with a performance comparable to that of PCs. On the other hand, Internet connectivity is provided on smart mobile phones, which has led to the development of handhelds, PDAs, and even mobile phones, offering standard office software, e.g. word processing, spreadsheet, calendar, e-mail and Internet browser.

Due to its nature, interaction with mobile computers may distract users' attention from their actual intent (e.g. contemplating the environment, operating a machine or device, driving a vehicle, etc.). Therefore, the (context-sensitive) acquisition of input data beyond classical user interfaces (display, keyboard, pointing device) is a hot research topic. One example of such approaches is e.g. spatial localisation using a global positioning system (GPS).

This leads to the concept of wearable computers. Wearable computers might be integrated with clothing to provide added functions to the user, bypassing explicit interaction. Thus, context cognition is the main difficulty to be solved using artificial intelligence (AI) software processing data of newly dedicated sensor interfaces [25].

4.4 Experiences and Outlook

Motivated by a novel classification of collaborative distances, an overview of the main characteristics and web-based applications within the context of distributed engineering communication using CVE has been given. Moreover, the advantages as well as disadvantages of such solutions were discussed. Last but not least, a table of tools and technologies contributing to overcome distance factors are presented (Fig. 4.13) with 11 factors identified in the structural dimension, 8 factors in the social dimension, 6 factors in the technical dimension and, finally, 3 factors in the legal and ethical dimension.



Dimensions	Distance factors (due to the lack of)	WebConf	IM	Whiteboard	SW	Forum	Blogging	Wiki	E-mail	VR & AR	MWC	*Polling*	*Semantic*	*Modelling*	*Workflow*	*SN*	*SG*	*EA*	*EN*
Structural	Collocation (shared space)			X	X			X		X									
	Communication	X	X			X	X		X										
	Coordination														X			X	X
	Leadership											X					X		
	Incentive															X	X	X	X
	Cohesiveness																X	X	
	Shared vision											X		X			X		
	Interoperability												X	X					
	Balanced decision	X										X					X		
	Synch. interactions	X	X	X															
	Asynch. interactions				X	X	X	X											
Social	Shared culture					X	X	X						X		X	X		
	Mutual understanding			X						X			X	X		X	X		
	Trusted relationships						X									X	X		
	Context awareness									X	X							X	X
	Social transparency																	X	X
	Interpersonal relationships						X									X	X		
	Social interactions	X		X		X	X									X	X		
	Emotional awareness	X	X				X			X						X	X		
Technical	Absorptive capacity		X							X							X		
	Shared references							X								X			
	Technology skill									X							X		
	Shared meanings			X									X	X					
	Shared relevance													X		X			X
	Correlation												X						X
Legal	Common IPR							X											
	Common privacy rules				X														
	Common security rules				X														

Fig. 4.13 Tools and technologies contributing to overcome distance factors (abbreviations used: IM: instant messaging; SW: shared workspace; MWC: mobile wearable computing; SN: social networking; SG: serious gaming; EA: expectation awareness; EN: events notification)

Experiences from field studies confirm that intuitive user interfaces (i.e. drag and drop) with social as well as personalisation features are crucial ingredients for successful user adoption of collaborative platforms. However, social transparency, social awareness and social intelligence are also related to specific working contexts. They all belongs to users' expected features for a more effective and efficient collaborative platform.

The topic of social awareness, if this wording truly corresponds to something other than presence awareness, is an interesting phenomenon to observe in terms of impact on the performance of the collaborative work performed where cooperative effort is a key element.

It appears quite often in the literature that different types of context such as social settings, geographical location, time zone, tools and technologies in use and activity types play a crucial role in people's social cognition, behaviour and task coordination. This means that everyone involved in a collaboration context tries to adapt his behaviour according to the current situation. It looks, more or less, like the stigmergy approach [28] where ants activities are driven by pheromones. This is also known as swarm intelligence.

There is no definition of social awareness corresponding to behavioural adaptation to various situations as social awareness is not only about what people know or obtain of awareness but also about what people consequently deduce and do. This fits perfectly with social intelligence, which is combining social awareness (what we sense) and social facility (what we do) as defined by Goleman [27].

Social awareness is about sensing the inner state of another person in order to understand related feelings and thoughts as well as complicated social situations. It includes various elements such as primary empathy (feeling with others; sensing non-verbal emotional signals); attunement (listening with full receptivity; attuning to a person); empathetic accuracy (understating another person's thoughts feelings and intentions); and social cognition (knowing how the social world works) [27].

Social facility builds on social awareness to allow smooth, effective interactions rather than simply sensing how one feels, or think or what one intends to do someone else. It includes elements such as synchrony (interacting smoothly at the non-verbal level), self-presentation (presenting ourselves effectively), influence (shaping the outcome of social interactions) and concern (caring about others needs and acting accordingly) [27].

During a recent case study, most members of the analysed teams recognised the great benefit of having a central storage (part of the shared space) where documents were available anytime from anywhere as long as an Internet connection was available. It seems that some of the study participants enjoyed working remotely from home and started to update documents outside normal operating hours [5].

Furthermore, most of the participants preferred to use instant messaging tools as synchronous communication rather than asynchronous communication. However, when necessary they preferred e-mails as an asynchronous communication tool rather than group blogging, especially because it was considered easier to get access to e-mail than to blogging. Many people claimed that uploading a document to the shared workspace by “drag and drop” was faster and much more reliable than sending an attached document by e-mail. Few participants expressed an interest in group blogging for the purpose of obtaining a project history and chronology of events but again re-emphasised that external participants would likely have encouraged more teams to take part in a project blog. Finally, a large majority recognised the complementarity of shared workspace and group blogging technologies [5].

In terms of collaborative distances, almost all participants said that the analysed technologies were useful to overcome distance factors, especially spatial distance, but also cognitive distance, emotional distance and, partly, cultural distance. It might seem strange, but emotional and social distances were also mentioned ways to remotely start a relationship with someone who is too shy or too emotional for live interaction. One participant said that it helped him or her to resolve a conflict, without indicating clearly what kind of conflict it was as if one member of the group was trying to impose his own views rather than showing more team spirit of consensus sensitivity. Concerning the distance factors created by those two technologies, the social distance became crucial as well as the technology distance [5].

Another interesting aspect is that most of the participants thought that collaborative platforms were useful if and only if project team members were in a distributed situation. However, the analysed users were progressively becoming more aware of other types of collaborative distance besides the well-known geographical one and that collaborative platforms could also be useful in a collocated situation. Thus, the ICT support of communication processes should be able to extend the aura of a place as well as that of an object for a person.[10]

Summarising, it must be stated that all technical solutions and possible communication modes are already possible; however, a number of problems remain unsolved. The main concerns revolve around the current lack of interoperable collaboration services, security issues, trusted relationships, social intelligence, more widespread skills and qualifications.

References

[1] Lüder, A. and K. Lorentz. 2005. IAONA Handbook Industrial Ethernet. Magdeburg: Industrial Automation Open Networking Alliance e.V. ISBN 3-00-016934-2.

[2] Zaremba, A.J. 2003. Organizational Communication: Foundations for Business & Management. 1st edn. Mason: Thomson/South-Western.

[3] Eisenberg, A. 1982. Effective Technical Communications. New York: McGraw-Hill.

[4] Borowick, J.N. 1996. Technical Communication and Its Application. Englewood Cliffs: Prentice Hall.

[5] Pallot, M., S. Richir and H. Samier. 2008. Shared Workspace and Group Blogging Experimentation through a Living Lab approach. In: Proceedings of the 14th International Conference on Concurrent Enterprising, ICE'2008 "A new wave of innovation in Collaborative Networks", 23-25 June, Lisbon.

[6] Wilson, J., M. Boyer O'Leary, A. Metiu and Q. Jett. 2005. Subjective distance in Teams. Working papers series, INSEAD.

[7] O'Leary, M.B. and J.N. Cummings. 2002. The spatial, temporal and configurational characteristics of geographic dispersion in work teams. Working paper, 148, MIT E-Business Center, Cambridge, MA.

[8] Pallot, M. 2007. Collaborative distance, factors that are affecting collaboration effectiveness and efficiency. ECOSPACE project deliverable. A literature review and a reference framework for collaborative research about distance and proximity factors affecting collaboration.

[9] Bergmann, U. 2008. Kommunikation als Optimierungskriterium: ein Beitrag zur Systematisierung der Layoutplanung von Produktionssystemen. Thesis (PhD). Otto-von-Guericke-University Magdeburg.

[10] Antonsson, E.K. and K.H. Grote. 2007. Springer Handbook of Mechanical Engineering. Berlin: Springer.

[10] Concerning VR and AR the aura of an object is the combination of its cultural and personal significance for a user or group of users. Cultural significance refers to shared meaning in a community. Personal significance refers to the individual associations that the place or object may have for a particular user. [26]

[11] Wigand, R., A. Picot and R. Reichwald. 2004. Information, Organization and Management: Expanding Markets and Corporate Boundaries. Chichester: Wiley (reprint).

[12] Ferran, C. and S. Watts. 2008. Videoconferencing in the field: A heuristic processing model. Management Science, 54 (9): 1565-1578.

[13] Garrett, R.K. and J.N. Danziger. 2007. IM=Interruption management? Instant messaging and disruption in the workplace. Journal of Computer-Mediated Communication, 13 (1), Available from: http://jcmc.indiana.edu/vol13/issue1/garrett.html

[14] Glover, D., D. Miller, D. Averis and V. Door. 2005. The interactive whiteboard: a literature survey. Technology, Pedagogy and Education, 14 (2): 155-170.

[15] Pallot, M., W. Prinz and H. Schaffers. 2005. Future Workplaces, towards the "Collaborative Web". In: Pallot, M. and K.S. Pawar, eds., Proceedings of the AMI@Work Forum 2005, Munic, pp 3-16.

[16] http://en.wikipedia.org/wiki/Internet_forum

[17] Pallot, M., R. Ruland, S. Traykov and K. Kristensen. 2006. Integrating shared workspace, wiki and blog technologies to support interpersonal knowledge connection. In: Proceedings of the 12th International Conference on Concurrent Enterprising, ICE'2006 “Innovative Products and Services through Collaborative Networks”, Milan.

[18] http://www.blogger.com/features

[19] http://en.wikipedia.org/wiki/Wiki

[20] http://en.wikipedia.org/wiki/E-mail

[21] Kühnle, H., U. Hess, D. Scheffter and T. Harada. 2005. Collaborative virtual planning and design of production systems. In: Ma, Q., R.J. Jiao, M.M. Tseng, Mitchell M., M.J. Zuo, eds., Industrial Engineering and Engineering Management in the Global Economy (11th International Conference Northeastern University) 23-25 April, Shenyang. Proceedings: Beijing: China Machine Press: 691-695.

[22] Scheffter, D., U. Bergmann and G. Wagenhaus. 2006. Virtual Commissioning. In: 6th International Conference on Production Engineering "Knowledge-Vision-Framework, 7-8 December, Wroclaw.

[23] Coelho, E.M. and B. MacIntyre. 2005. Augmenting real environments. In: The Third Young Inverstigator's Forum in Virtual Reality (YVR 2005), 24-25 February, Pohang.

[24] Antoniac, P., M. Pallot and P. Pulli. 2006. Virtual and augmented reality supporting group consciousness within collaborative working environments. In: Proceedings of the 12th International Conference on Concurrent Enterprising, ICE'2006 Innovative Products and Services through Collaborative Networks, Milan.

[25] Boronowsky, M., O. Herzog, M. Lawo and P. Knackfuss. 2006. Wearable computing – a new approach in concurrent enterprising. In: Proceedings of the 12th International Conference on Concurrent Enterprising, ICE'2006 Innovative Products and Services through Collaborative Networks, Milan.

[26] MacIntyre, B., J.D. Bolter and M. Gandy. 2004. Presence and the aura of meaningful places. In: 7th Annual International Workshop on Presence, Polytechnic University of Valencia, 13-15 October, Valencia.

[27]Goleman, D. 2006. Social Intelligence: The New Science of Human Relationships. Bantam Books : Random House Inc.

[28] Elliott M. 2006. Stigmergic collaboration: the evolution of group work. M/C Journal 9.2. 21 Mar. 2009 <http://journal.media-culture.org.au/0605/03-elliott.php>.

5 Communication Systems as an Integral Part of Distributed Automation Systems

Thilo Sauter[1] and Albert Treytl[2]

[1] Austrian Academy of Sciences, Research Unit for Integrated Sensor Systems, Viktor Kaplan Straße 2, 2700 Wiener Neustadt, Austria, e-mail: thilo.sauter@oeaw.at

[2] Austrian Academy of Sciences, Research Unit for Integrated Sensor Systems, Viktor Kaplan Straße 2, 2700 Wiener Neustadt, Austria), e-mail: treytl@ict.tuwien.ac.at

Abstract Distributed automation is unthinkable without adequate communication. This fact was discovered quite early, and from the first ideas of computer integrated manufacturing onwards, the need for networks in automation has been addressed. This chapter reviews the history of fieldbus systems and the problems of standardization and sheds light on the complex variety of existing approaches. Furthermore, it will discuss the recent adoption of Internet technologies and Ethernet as automation network and the ongoing work to overcome its real-time limitations. Finally, we address evolution prospects and current research issues ranging from the apparent security problem and synchronization mechanisms in distributed systems to entirely new approaches like bionics-inspired concepts.

5.1 Introduction

Distributed systems are currently a fashionable term, and yet there is no clear definition of what constitutes a distributed system. What can be found, though, are characteristic properties. A distributed system comprises several components that are more or less autonomous in executing their tasks. It does not matter if on a higher level these individual operations are coordinated by a central entity or if the coordination is more decentralized. The essence, however, is the fact that there is some coordination and synchronization, and this necessitates the exchange of information between the distributed system components.

The means for the information flow need not necessarily be communication networks as we know them today. If we look at automation systems of a hundred years ago, when electronic components were not an issue, the different involved devices communicated indirectly with each other – via the workpieces. Assembly lines are still a good example of such early distributed systems. The various machines along the line were activated as soon as the workpiece arrived.

State of the art today is a global communication network inside the automation system, such that all components of the system share the same status of information. In the example given before, all machines could know immediately when a workpiece arrived at one particular station – be it relevant or not. The networks devised for the special problems in automation are the fieldbus systems (derived from the process field they are located in). Not only do they connect machines in production lines (which is already a rather high-level communication), they are also used inside the machines, where the requirements are again different. According to the various application areas with their peculiarities, a large number of fieldbus systems exists today [1]; many people and international organizations spent lots of effort to look for the best technical solutions.

It is interesting to note that on an application level, the current concept of global synchronization inside an automation system is again being questioned. Advanced ideas like holonic or agent-based automation systems [2] return somehow to the original way of synchronizing activities: indirectly via the workpiece, albeit this time with heavy use of information and computer technology. This is seen as a possible way to cope with the growing complexity of automation systems [3].

On the communication level, fieldbus systems and, with increasing importance, also Ethernet- and IP-based communication systems are still using centralized concepts. It is therefore the aim of this article to review the basic ideas and properties of dedicated automation networks. Sections 5.2 and 5.3 sketch the historical evolution of fieldbus systems and provide an overview of the large variety of different solutions. Section 5.4 discusses the new approaches derived from Internet technologies. Section 5.5 is devoted to the emerging topic of security, which becomes an essential issue in networked automation systems. Finally, Sec. 5.6 tries to give an outlook on the future of automation systems. It can be anticipated that with the sharply increasing number of nodes in a network (if we think of building automation or sensor networks), new concepts of dealing with the information flood have to be sought.

5.2 History

From a high-level perspective, communication in automation is just a means to an end that should simply work reliably and offer services supporting the needs of automation applications. This is what the user is interested in; all other details are of no importance. In particular, the peculiarities of the communication system are uninteresting for everyone but the developer and the system integrator concerned with the commissioning of a plant. Understanding this rather plain fact is the key to understanding how the situation in the automation network domain could become as confusing as it is today.

Automation networks, namely fieldbus systems, were often advertised as unique revolutionary achievements in the past. Actually, they were the almost logical product of a long evolution that incorporated largely three independent roots [4].

The first and oldest influence factor, though only concerned with point-to-point connections and therefore not too relevant for distributed systems, were telecommunication systems. The Telex system of the 1930s can be seen as the first system to transmit standardized control characters over wire worldwide in an automated way. In the 1960s it was already common for companies to rent lines from telephone companies to transmit data, and the Comité Consultatif International Télégraphique et Téléphonique (CCITT) defined the first protocols for using "normal" telephone lines. These protocols and definitions regarding signal behavior had some influence on today's automation protocols.

The second predecessor of automation networks were bus systems for instrumentation. They already comprised all the features that are necessary for distributed systems: multipoint connectivity structures, synchronization possibilities, and broadcast and unicast communication. Even though these systems (like the IEEE 488 or "Computer Automated Measurement And Control" (CAMAC)) were parallel bus systems with mostly dedicated control lines, only a small step remained to the development of modern fieldbus systems: the use of one single bus wire for the entire data and control transfer.

In a way, related to the instrumentation busses are board-level busses like the still widely used I^{2}C. These busses appeared later than the instrumentation busses and went hand in hand with the progress in integrated circuits. As their objective was to interconnect Integrated Circuits (ICs), where pin count on the packages was a major issue, they had to have a very limited number of lines, where no dedicated control signals could be transmitted (maybe just a system clock as in the case of I^{2}C). So these busses, although restricted to very short distances and with simple data transmission functionality (it is not really justified to speak of a protocol), anticipated the serial structure of fieldbus systems.

A decisive influence was the development of computer networks. The definition of the International Organization for Standardization/Open Systems Interconnection (ISO/OSI) model paved the way for the realization of complex systems. Powerful protocols, especially in the telecommunications area, came up and still exist, for example, X.25 or SS7. On the other hand, the layered approach to protocol design stimulated the Manufacturing Application Protocol (MAP) and later Manufacturing Message Specification (MMS) initiative [5, 6]. The comprehensive MAP project tried to define and implement an overall communication model for industrial applications following and extending the CIM idea. The still very common "automation pyramid" concept [7] dates back to these days. During the work on MAP it was discovered that for the lowest level in the hierarchy, the field level, no adequate networks existed. This recognition was at least one starting point for the development of fieldbus systems.

Against this backdrop of available and emerging technologies, it is not astonishing that the mid 1980s saw numerous activities in field-level networking. One enabling technology supporting this boom was microelectronics. Only with the drawing era of highly integrated circuits could microcontrollers be realized thatprovided sufficient computing power for handling the increasingly complex communication protocols. Another advantage of circuit integration was a significantly improved electromagnetic compatibility. The last piece in the puzzle was the development of electrical interfaces like the RS 485, which uses fully differential data transmission and thus allows high noise immunity even in harsh industrial environments.

At any rate, the time for fieldbus systems had come, and many companies and consortia developed solutions. The starting point was the same for all these activities: automation systems had grown complex, but the field level was still dominated by conventional, mostly analog, point-to-point cabling structures that were not only costly in installation and maintenance, but also limited in functionality. Digital bus systems promised cheaper and more flexible wiring, as well as a boost in overall system functionality because the digital connection could transfer more information than just the values of sensors and actuators. Improved configuration, commissioning, and operation were among the obvious benefits.

While the starting points for the development were clear and basically identical for all developments, the application areas and their specific problems were diverse. Consequently, the solutions tailored to the particular fields were diverse as well. In addition, the level of innovation attempted by the developers varied. Some fieldbus systems were originally intended mainly as a replacement for the parallel field-level cabling (like Interbus), while others were – even less ambitious from the networking point of view – only conceived as improved interfaces (like CAN). Some approaches followed the top-down strategy proposed by MAP/MMS (such as PROFIBUS), whereas others had a rather bottom-up development of an entirely new, real-time capable distributed operating system in mind (like FIP [8]).

It must be noted that the early days of fieldbus systems were vendor-driven. As the end users' intention was to obtain turnkey solutions for their automation systems, they did not actually care about low-level details. For the system integrators, on the other hand, the overwhelming plethora of fieldbus systems emerging all around was discouraging. The plain old interconnection standards, above all the current loop interface, were vendor-independent and universal. The new fieldbus systems were mostly vendor-specific and closed solutions, which did not improve their acceptance on the market. Noticing this, many vendors made the specifications public and worked together with others in user groups to foster so-called open systems.

The next important step to strengthen the position of fieldbus systems (as well as users' confidence) was standardization. Actually, an international project to define fieldbus standards had been launched by the International Electrotechnical Commission (IEC) in 1984 for industrial and process automation soon after the need for field-level networking had become obvious [1, 9]. The work started with a thorough investigation of possible solutions. As a result of this evaluation, two European approaches (FIP and PROFIBUS) were retained for further consideration. During the following years, several attempts to select either of the two or to finally combine the benefits of both failed. The reason for the increasing obstruction of the standardization work was that by the time fieldbus systems started to become market relevant, the vendors tried to achieve strategic advantages through standardization. This was mainly due to the fact that standards are strictly enforced in some countries (especially in Europe). So, having one's own (and the competitor's not) system standardized would improve market position. Consequently, the actions of companies present in the standardization committees were marketing- and less technology-oriented.

The situation of the standardization process became tangled [10]. Parallel activities were started in Europe by the Comité Européen de Normalisation Electrotechnique (CENELEC) to overcome the IEC's inability to reach a conclusion. After US companies attained more influence on the IEC committee SC65, European pressure groups tried to inhibit an international fieldbus standard, contradicting the European solution that had been found in the meantime. Finally, after endless debates and intrigues, the IEC compelled a standard consisting of the solutions that had supporters inside the standardization committee. So it came about that the standards IEC 61158 and IEC 61784 define the fieldbus systems Foundation Fieldbus, ControlNet™[1], EtherNet/IP™[2], PROFIBUS DP and PA, PROFINET, P-NET, WorldFIP, Interbus, and SwiftNet. The one and only field-level communication system – the original intention of the standardization – remained wishful thinking [11].

[1] ControlNet™ is a registered trademark of ControlNet International Ltd., PMB 315, 20423 State Road 7 #F6, Boca Raton, Florida 33498-6797, USA

[2] EtherNet/IP™ is a trademark of ControlNet International under license by ODVA, Inc., Ann Arbor, Michigan, USA, http://www.odva.com

5.3 Varieties of Bus Systems

The fieldbus solutions derived from the different application areas are as different as the application areas themselves. Nevertheless, and despite their differences, they all share a common foundation in that they are communication networks for distributed systems. This has had a significant influence on the design of the communication concepts and protocols. Another typical property of fieldbus protocols is the desire for efficiency. Historically, developers had to cope with limited resources on the fieldbus nodes, as well as with transmission capacity limitations on the bus itself. Therefore, the protocols were usually designed for utmost efficiency, with respect to both complexity and frame size. Only recently – with the growing use of Ethernet and Internet technologies for automation purposes – the efficiency goal was questioned with respect to the large amount of resources high-speed Ethernet and modern processors offered. There is no point in discussing the peculiarities of individual fieldbus systems. Therefore, what are described in the sequel are the general concepts and differences with computer networks that can be found in most fieldbusses.

5.3.1 Communication Concepts

One difference with LANs concerns the protocol stack. Like all modern communication systems, fieldbus protocols are modelled according to the ISO/OSI model. However, in industrial and process automation normally only layers 1, 2, and 7 are actually used [1]. This is in fact a tribute to the lessons learned from the problems with MAP, where it was found that a full seven-layer stack required far too many resources and does not permit an efficient implementation. For this reason, the MiniMAP approach, and based on it the IEC fieldbus standard, explicitly prescribes a three-layer structure consisting of physical, data link, and application layers.

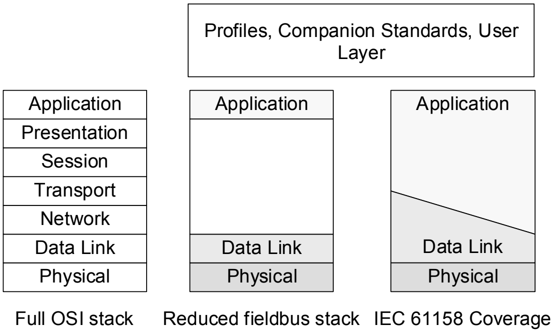


Fig. 5.1 Communication protocol stacks according to the ISO/OSI model and typical stack structure in fieldbus systems

In most cases, this reduced protocol stack reflects the actual situation found in many automation applications anyway. Fieldbusses typically are single-segment networks, and extensions are realized via repeaters or, at most, bridges. Therefore, network and transport layers – which contain routing functionality and end-to-end control – are simply not necessary if systems in areas like home and building automation are not taken into consideration. The functions of layers 3 to 6, if needed, are often included in layer 2 or 7. This was the explicit design guideline for the IEC 61158 fieldbus standard (Fig. 5.1) [12]. Fieldbus systems in the home and building automation domains (LonWorks, EIB/KNX, and BacNet) have other constraints and should not be an issue here. Owing to the possibly high number of nodes, these fieldbus systems must offer the capability of hierarchically structured network topologies, and a reduction to three layers is not possible.

For typical process control applications, determinism of data transfer is a key issue, and cycle time is a critical parameter. This fact has been the optimization criterion for many different fieldbus protocols and the reason that they are different from conventional LANs. Particularly the physical layer has to meet substantially more demanding requirements like robustness, immunity to electromagnetic disturbances, intrinsic safety for hazardous areas, or costs. The significance of the physical layer is underpinned by the fact that this area was the first to reach (notably undisputed) consensus in standardization [10].

On the data link layer, all medium-access strategies also known from LANs are used, plus many different subtypes and refinements [13]. Simple master-slave polling (ASi, PROFIBUS-DP) is used as well as token-based mechanisms in either explicit (PROFIBUS, WorldFIP) or implicit (P-NET) form. Carrier sense multiple access is mostly used in a variant that tries to avoid collisions either by dynamic adaptation of retry waiting times (LonWorks) or the use of asymmetric signaling strategies (CAN, EIB). Especially for real-time applications, TDMA-based strategies are employed (TTP, essentially also Interbus). In many cases, the lower two layers are implemented with ASICs for performance and cost reasons. As a side benefit, the preference of dedicated controllers over software implementations also improves the interoperability of devices from different manufacturers, since only a few hardware-based solutions exist.

An essential part of fieldbus protocol stacks are comprehensive application layers. They are indispensable for open systems and form the basis for interoperability. Powerful application layers offering abstract functionalities to the actual applications, however, require a substantial software implementation effort, which can negatively impact the protocol processing time and also the costs for a fieldbus interface. This is why in many cases (like Interbus or CAN) an application layer was originally omitted. While the application areas were often regarded as limited in the beginning, market pressure and the desire for flexibility finally forced the addition of higher-layer protocols, and the growing performance of controller hardware facilitated their implementation.

5.3.2 Communication Paradigms

The characteristic properties of the various data types inside a fieldbus system differ strongly according to the processes that must be automated. Application areas like manufacturing, process, and building automation pose different timing and consistency requirements that are not even consistent within the individual application areas [14]. Typical examples for different timing parameters are continuous measurement data, which are sampled and transmitted in discrete-time fashion and form the basis for continuous process control and monitoring (like temperature, pressure, etc.). Other data are typically event-based, i.e., they need transmission only in case of status changes (like switches, limit violations, etc.). As far as consistency is concerned, there are on the one hand process data that are continuously updated and on the other hand parameterization data that are transferred only upon demand. In case of error, the former can easily be reconstructed from historical data via interpolation or simply be updated by new measurements. The system-wide consistency of configuration data, on the other hand, is an important requirement that cannot be met by mechanisms suitable for process data.

These fundamental differences led to the evolution of several communication paradigms that are used either individually or in combination. The applicability in different fieldbus systems is quite different because they require various communication services and media access strategies. The three basic paradigms are the client-server, the producer-consumer, and the publisher-subscriber model [13]. The first employs a point-to-point communication strategy, whereas the latter two follow a point-to-multipoint approach.

Processes with mostly event-based communication can get along very well with producer-consumer-type communication systems, especially if the requirements concerning dynamics are not too stringent. The obvious advantage is that all connected devices have direct access to the entire set of information since the broadcasting is based on identification of messages rather than nodes. Reaction times on events can be very short due to the absence of slow polling or token cycles. Generally, producer-consumer-type systems (or subsystems) are necessarily multimaster systems because every information source (producer) must be able to access the bus. The selection of relevant communication relationships is based solely on message filtering at the consumer's side. Such filter tables are typically defined during the planning phase of an installation.

The publisher-subscriber paradigm is often used synonymously for the producer-consumer model. The only subtle difference mentioned sometimes is that multicast communication services are being employed. The subscribers are typically groups of nodes that listen to information sources (publishers). Relating publishers and subscribers can be done online. As both paradigms are message-based, and therefore connectionless on the application layer, they are not suited for the transmission of sensitive, nonrepetitive data such as parameter and configuration values or commands. Connectionless mechanisms can inform the respective nodes about communication errors on layer 2, but not about errors on the application layer.

The client-server paradigm avoids this problem by using connection-oriented information transfer between two nodes with all necessary control and recovery mechanisms. The communication transfer itself is based on confirmed services with appropriate service primitives (request, indication, response, confirm) as defined in the OSI model. Basically, a client-server-type communication can be implemented in both mono- and multimaster systems. In the latter cases (Carrier Sense Multiple Access (CSMA) and token-based systems) every master can take on the role of a client, whereas in monomaster systems (polling-based) this position is reserved for the bus master. Consequently, the client-server paradigm is used mainly for monomaster systems as well as generally for discrete-time (cyclic) information transfer and for reliable data transfer on the application level (e.g., for parameterization data).

It is a characteristic feature of fieldbus systems that they do not adhere to single communication paradigms but support a mix of strategies on different levels of sophistication. Examples of typical client-server systems are Interbus, PROFIBUS DP, P-NET, or ASI. Broadcast services are here only used for special cases like synchronization purposes. Likewise, there are special ways of receiving messages (e.g., direct slave-to-slave communication) that require temporary delegation of certain bus master aspects. The other two paradigms are widely used in systems like CAN, CANopen, DeviceNet, ControlNet, EIB, or LonWorks. Yet these systems also employ the client-server paradigm for special functions such as node configuration, file transfer, or the like.

5.4 The Internet Revolution

The evolution of the Internet has had a tremendous influence on automation systems in general and on their underlying communication systems in particular. Specifically, there are two major aspects of "the Internet" that gained importance during the last decade: Web technologies in a broad sense on a higher level and Ethernet as communication system on a lower level.

5.4.1 The Internet in Automation

Although the Internet consists of much more than the World-Wide Web, the sheer ubiquity of the WWW and the availability of software tools on any arbitrary platform has enforced public perception. Hence it is no wonder that these quasi-standards became attractive for the automation area. First and foremost, Web-based approaches (as well as other Internet-specific protocols) are being used as a means to remotely access automation systems, mostly for monitoring and configuration purposes. Obviously, the biggest advantage is the simple integration into the higher levels of a company hierarchy dominated by office applications. This has finally revived the old CIM idea of vertical integration [3]. Generally speaking, Internet technology seems to make possible these days what was bound to fail even 20 years ago. Examples of this trend are the adoption of Web standards for business and enterprise integration [15], but also the use of distributed software paradigms – which originated in the Internet – for resource and production planning [2]. This last example is particularly appealing because it bridges the traditional gap between the high-level, information-technology-oriented enterprise environment and the low-level, communication-technology-oriented factory floor.

More related to actual communication systems, Internet technology is today the basis for device description languages like Electronic Device Description Language (EDDL) that simplify and unify device and system configuration [16]. At the same time, more and more field devices are being equipped with embedded Web servers to permit easy remote access via http [17]. As a supporting measure, many fieldbus systems have already included in most recent versions the ability to tunnel IP traffic over the fieldbus protocol – or are gradually being replaced by new solutions based on modern communication technologies.

5.4.2 Industrial Ethernet

It is only a small step from the relatively cumbersome incorporation of IP traffic in fieldbus protocols to the use of fully IP-based communication systems – and Ethernet as an underlying transport medium. Actually, much of the Industrial Ethernet hype has to be seen under the aspect of using Internet technologies. Traditional fieldbus systems had already been developed in a variety before the WWW (the Internet itself was already ten years old by that time) was even invented. The tremendous impact of Ethernet and the IP suite as defacto standards in the office world came too late for the first evolutionary wave of automation systems. Nowadays, the market seems ready for a second round. The significant price drop in office Ethernet equipment supports marketing campaigns advertising Industrial Ethernet as an affordable solution (even though the comparison with standard office equipment is unjustified both in a technical and a financial sense).

The impression of Industrial Ethernet as a homogeneous standard (as in the office world) is deceptive and does not reflect reality in automation. There are in fact many approaches to Ethernet and Internet technologies, and more than 15 different systems are available.

One particular problem concerns the native lack of real-time capabilities in Ethernet [18]. For automation purposes, especially for distributed systems, some sort of coordination between nodes and processes is essential, which requires real-time capabilities. Many traditional fieldbus system have dedicated and highly sophisticated mechanisms for this purpose. Plain Ethernet does not. This creates the need for special measures and is one of the reasons for the variety of Industrial Ethernet and "Industrial Internet" approaches. Roughly there are four main categories [19]:

1. Tunneling of TCP/IP over an existing fieldbus protocol, e.g., to access embedded Web servers in field devices;
2. Tunneling of a fieldbus protocol over UDP/TCP/IP;
3. Definition of new (possibly real-time enabled) protocol outside the classical IP suite;
4. Real-time Ethernet by changing the medium access, including low-level hardware modifications.

Category 1 is a rather simple one. Solutions were already contained in the original IEC Standard 61158. Interbus defines the tunneling of TCP/IP over its acyclic communication channel. Also for WorldFIP, an extension for transmitting IP frames inside the fieldbus packets is available.

Category 2 is more involved. Here, a conventional fieldbus protocol is "tunneled" over TCP/IP. To be specific, it is not real tunneling, where data packets of a lower fieldbus OSI layer are wrapped in a higher-layer protocol of the transport medium. Instead, the same application-layer protocol that was already defined for the fieldbus is also used over the TCP/IP or UDP/IP protocol stack, depending on the type of data. Typically, process-related data are transmitted via UDP because it is faster. Configuration-related data, on the other hand, are transferred via TCP, because it is more reliable. Foundation Fieldbus High-speed Ethernet and Ethernet/IP (note that here IP intelligently stands for "Industrial Protocol" to make confusion perfect) are two such solutions from the IEC 61158. Modbus/TCP is a third example now under consideration for the new IEC 61784-2. In some sense, also the first version of PROFINET (CBA, component-based automation) can be seen in this class, although the high-level protocol is not a well-known fieldbus application protocol but a newly defined one originally based on DCOM as a distributed middleware platform.

Category 3 is the first step toward actual real-time solutions for Ethernet-based automation networks. The typical approach is to entirely bypass the TCP/UDP/IP stack (which can nevertheless be used for non-real-time purposes) or to use modified implementations of the IP suite. The second version of PROFINET (RT, real time) is a typical example of bypassing IP to achieve better performance. VNET/IP as a second approach of this kind uses an optimized IP stack.

Category 4 comprises all systems developed for special real-time requirements up to particularly demanding motion control applications. All these approaches modify Ethernet itself in one form or another. Ethernet Powerlink places a TDMA scheme on top of conventional Ethernet hardware, thus eliminating the inherently unpredictable CSMA/CD access mechanism. Furthermore, it uses a shared (not switched) medium to avoid stochastic delays in network switches. TCnet also uses a shared medium and a modified Media-Access-Control (MAC) layer, like EPA, which employs a time-slicing mechanism. PROFINET version 3 (IRT, isochronous real time) uses switched Ethernet but requires a dedicated switch ASIC with short and reproducible cut-through times together with hardware time stamping. EtherCAT [27] uses Ethernet frames and a special ring topology and data exchange mechanism resembling Interbus. In addition, it needs a nonstandard controller. This is also true for SERCOS III, where Ethernet frames are used in a ring network based on special hardware.

5.4.3 Synchronization in Distributed Systems

Essential components of distributed systems are synchronization mechanisms. Applications spread over the network need to be coordinated, with varying requirements concerning the temporal tightness. At one extreme are *tightly coupled* systems, where actions on different network nodes must be precisely timed relative to one another. Most low-level control activities in larger plants fall into this category. Tightly coupled control systems need to rely on synchronization mechanisms that are part of the communication network, mostly connected to its lower protocol layers. In traditional fieldbus systems, such mechanisms have been included more or less explicitly depending on the primary application field. For safety-relevant applications (e.g., automotive), time-triggered systems have been developed with very strict TDMA schemes ensuring hard real-time behavior. In factory and process automation, the very common master-slave approaches (such as the sync/freeze method in PROFIBUS) in principle also allow for a synchronization of the nodes.

Things are different with Industrial Ethernet and IP-based networks. Here, the communication system by itself does not provide adequate mechanisms (unless TDMA schemes are used instead of the conventional access mechanism). Even in switched Ethernet, where collisions cannot occur, the stochastic queuing in the switches imposes bounds on the achievable synchronization accuracy. Therefore, other approaches have been devised that aim at the synchronization of local clocks present in every node in the system. The underlying idea is that if all nodes share a common notion of time, the application processes can be coordinated by some overall scheduling, and imperfections of the communication system have less impact. Synchronization protocols include NTP (the network time protocol well known as the basis for synchronization in IP-based networks) and the IEEE 1588 standard. The latter originated in the instrumentation community and specifies the master-slave-type Precision Time Protocol (PTP) for delay measurement and clock synchronization. Issued only in 2002, it has been adopted by most of the real-time Ethernet approaches currently under standardization. The general problem is that the variability of the network delays between the nodes sets an upper limit to the achievable synchronization accuracy. If very high accuracy is needed, hardware modifications in the network interfaces can be used to either reduce the delay variations in frame processing or at least to explicitly measure the storage times of frames on, e.g., switches [20].

At the other extreme of distributed processes are *loosely coupled* systems. Here, application processes run independently of each other and need interaction only once in a while. This type of distributed system is typical for the higher layers of the automation hierarchy where real-time requirements do not exist or are not stringent. Coordination is mostly achieved by application-layer protocols using request-response mechanisms where latencies do not matter and timeouts are long. TCP/IP-based networks facilitate this type of communication, and actually most application-layer client-server protocols known from the Internet (like HTTP, SMTP, SNMP, etc.) can be used to implement autonomous, loosely coupled systems. Software agents, in particular multiagent systems such as those used in PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises (PABADIS'PROMISE) are other good examples. Their communication is based on the Foundation for Intelligent Physical Agents (FIPA) agent communication language, which defines a set of parameters for messages, the way data are put into containers and encoded, and ultimately uses HTTP to exchange data between the agents. The agents in PABADIS'PROMISE are fully autonomous, and only by means of this type of communication do they form a distributed automation system.

5.5 Security

For a long time automation systems used to be closed environments, and based on this philosophy fieldbus systems were standalone networks as well. Security features were therefore never a primary goal of fieldbus system development, even though the fieldbus was (at least in the IEC standardization work) recognized as part of a comprehensive networked company environment [9]. In recent years, however, particularly since the success of the Internet and the trend of using Ethernet and Internet technologies in automation, a connection of automation system and networks to an "outside world" has become feasible and (for flexibility and cost reasons) also reasonable. Security becomes an obvious necessity in such a situation, and solutions tailored to the specific boundary conditions in automation have to be sought.

Surprisingly, when it comes to discussing security in automation, an argument frequently heard is that methods known from the Internet world and firewalls in particular would be sufficient to protect an automation system. While this is in principle a good approach (following Kerkhoff's principle, a fundamental tenet especially in cryptography, that only proven methods should be employed), it neglects the peculiarities of the automation world. Taking them into account, it appears that firewalls are at most only one piece in the puzzle, but by no means a cure-all [21].

In the world of commerce, where security plays an important role, various security threats may be considered:

1. Tampering information
2. Stealing information
3. Destroying information
4. Destroying resources
5. Stealing resources

Not all of these aspects are as well important in the world of automation. Stealing information (e.g., by eavesdropping on a communication) is rather uninteresting. To steal resources makes no sense either, but to tamper with or to destroy information and to destroy resources are attacks that can lead to critical situations. Based on such a rough risk analysis, a first conclusion can be drawn: information confidentiality (which is usually enforced with encryption) is much less important than information integrity (including authentication of sender and receiver).

The second step after a risk analysis has to be the definition of a security policy for the actual system and the applications. This policy sets the constraints for the system components and gives guidelines for the selection of proper security measures to be used for system implementation. One such measure employed for the interconnection of an automation network and the Internet (or any IP-based network) can be a masquerading firewall, possibly with proxy functionality to separate the two networks logically and physically. Another constraint may be the physical protection of this gateway computer itself to prohibit manipulation.

When attacks do not originate only from outside the system but also from the inside, it is not sufficient to protect the (central) access points, e.g., by a firewall concept, but the network itself and all its components should to be secured as well. On the IP level, security mechanisms are readily available for this purpose. On the fieldbus level, things are much more difficult. As stated before, fieldbus systems are security-unaware, in particular those devised for industrial and process automation [22]. Those used in building automation have integrated security features, albeit with a rather low quality [23]. Although it is usually not recognized as such, perhaps the biggest potentials in the Industrial Ethernet movement would have been the integrated security features on a low level from scratch [19] – something that cannot be done for legacy fieldbus systems. Unfortunately, this opportunity was not seized upon in practice [24].

If there are no native built-in security mechanisms in the communication system, security can be introduced only indirectly and on an application level. In such cases, the hardware is also usually not prepared for security extensions. One possibility for coping with both problems is the introduction of dedicated security tokens like smart cards, which are widely used in, e.g., cellular phones. Such devices can be used in principle to implement application-level end-to-end security and also help if attackers have physical access to the security-critical device. Especially in building automation or systems for energy distribution with many accessible components this is a serious problem compared to e-commerce, where all activities are carried out over the network and it is comparatively easy to physically protect the relevant servers.

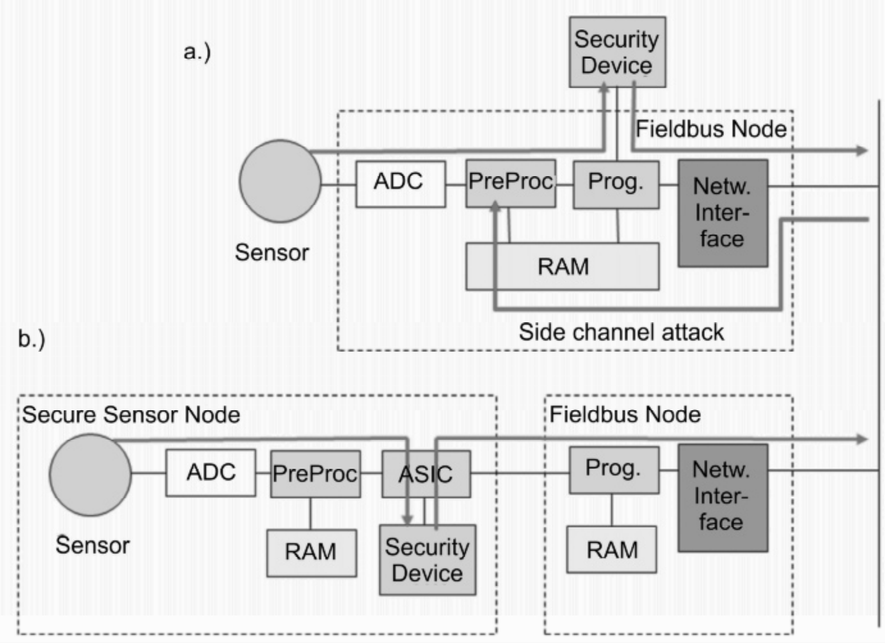


Fig. 5.2 Integration of a security device in a fieldbus node - *a)* Threat of side channel attack via network management - *b)* Separated secure sensor module

It must, however, not be overlooked that the integration of security devices is not at all straightforward. Rigorous security entails the restriction of communication relations, which is usually a problem for all broadcast- or multicast-oriented functions. The performance of fieldbus node processors and the secure serial communication links to smart cards is limited, so that the encryption or authentication may be too timeconsuming for real-time process data. Finally, the physical integration in the node hardware is problematic as well. Fieldbus systems have a highly sophisticated network management that often needs access to at least parts of the controller's internal memory, bearing the danger that the security device can be circumvented (Fig. 5.2a). Yet the straightforward solution to place the smart card together with some glue logic between the fieldbus controller and the sensor device would prevent fieldbus management from accessing the memory. However, since both components are integrated behind the fieldbus controller (seen from the fieldbus), a preprocessing of the sensor data in the controller is no longer possible and makes the whole device relatively inflexible (Fig. 5.2b).

In some cases an integration of security tokens is not feasible for technical or cost reasons. To still provide security mechanisms requires taking into account the resource limitations on the individual platforms and devising appropriate security architectures. This has been done in the PABADIS'PROMISE concept for Manufacturing Execution Systems, where the adhoc nature of the multiagent system poses an additional challenge. From the communication viewpoint, two different types of nodes and communication channels exist. The first relates to wired connections between stationary agents that are hosted on typically powerful platforms such as high-end servers or (industrial) PCs used for central services. These devices offer enough computational resources to execute state-of-the-art asymmetric security algorithms and sufficient communication bandwidth. On the other hand, the platforms for the mobile order agents have relatively small computing power and even less communication bandwidth because they mainly use Radio Frequency Identification (RFID) mechanisms. Therefore, simpler and computationally more efficient symmetric algorithms are used between these devices.

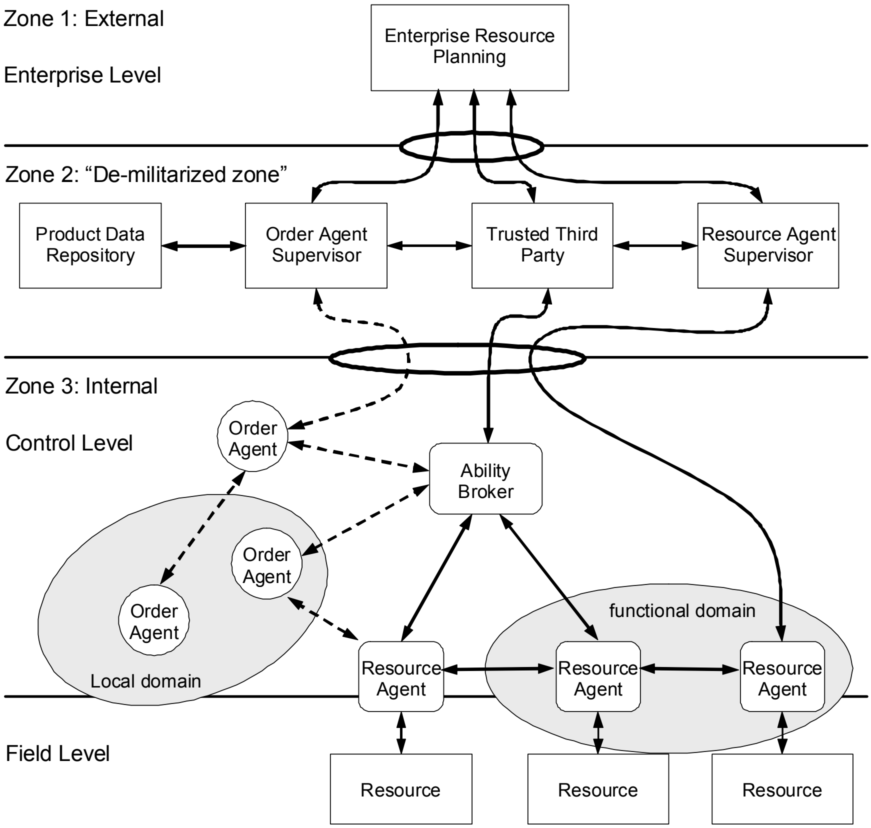


Fig. 5.3 Structured security architecture in PABADIS’PROMISE with rigid zones and flexible domains. The boundaries between the zones are protected by firewalls

Beyond the mere communication security aspect, large-scale and hierarchical systems also need a structured approach within the security architecture to group entities working on similar levels and to make data exchange more efficient within these groups (Fig. 5.3). For example, in PABADIS'PROMISE, the communication entities are grouped into zones that are related to the hierarchy levels of the automation pyramid [25]. The boundaries between the zones are particularly protected in order to relax security requirements inside. This is especially true for the RFID-based agent group, which uses only weak security mechanisms. Additional grouping to facilitate communication relations even further is done by means of a domain concept bundling resources with similar functionality or location. These domains can be changed dynamically (as opposed to the zones) and map the high flexibility of the system architecture to a correspondingly flexible security concept.

5.6 Future Automation Systems

It was already indicated in the introduction that the models conceived for fieldbus development are not sufficient to realize large-scale networks with thousands of devices in an efficient and low-cost way. Building automation systems with more than 10.000 nodes give a first flavor of future scenarios. Sensor networks will become even larger. Such networks need to be engineered and managed. More than that, also the actual process data transmitted inside such networks need to be structured and processed in a manageable way. The old hierarchical models that stood at the beginning of the fieldbus era were appropriate for small networks, and even the process pyramid model that will be standardized in CEN (TC 247 WG4) for building automation will not really be helpful. The abstract ISO/OSI model was defined with telecommunication applications in mind and does not fulfill the particular requirements of real-time communication in distributed systems. On a higher level, the layers introduced in the communication models in an attempt to find abstract and harmonized concepts have grown over the years. Companion standards and profiles are currently a must, but what will come on top of that? In any case, these layers will gain complexity and move closer to the application [3, 26].

New ideas, new views on communication networks are needed, and bionics can be one approach. Yet, focusing only on biological "IT systems" will not be sufficient to this end [4]. Communication systems are an integral part of automation and should not be treated separately. Rather, a holistic view to find new and more efficient concepts is more promising, in particular when it comes to finding solutions for distributed systems. Software agents and autonomous, loosely coupled systems are already a first step in this direction. Such systems, however, will need new models that go beyond adding just another level on top of the fieldbus profiles, perhaps by adopting "softer" models from psychology and psychoanalysis that explain in a nontechnical way how complex distributed systems operate [27, 28]. At any rate, modern distributed automation systems are gradually reaching a level of complexity and sophistication that limits the applicability of traditional methods. To tread new paths is not just a promising alternative, but a necessity.

References

[1] Sauter, T. 2005. Fieldbus Systems – History and evolution. In: R. Zurawski, ed. *The Industrial Communication Technology Handbook.* Boca Raton: CRC Press.

[2] Peschke, J., A. Lüder, A. Klostermeyer, A. Bratoukhine and T. Sauter. 2005. Distributed automation: PABADIS vs. HMS. *IEEE Transactions on Industrial Informatics,* 1: 31-38.

[3] Sauter, T. 2007. The continuing evolution of integration in manufacturing automation. *IEEE Industrial Electronics Magazine,* 1: 10-19.

[4] Dietrich, D. and T. Sauter. 2000. Evolution potentials for fieldbus systems. In: *3rd IEEE International Workshop on Factory Communication Systems.* September, Porto. pp. 343-350.

[5] Schutz, H.A. 1988. The role of MAP in factory integration. *IEEE Transactions on Industrial Electronics,* 35: 6-12.

[6] Shanmugham, S.G, S.G. Beaumariage, C.A. Roberts and D.A. Rollier. 1995. Manufacturing communication: the MMS approach. *Computers & Industrial Engineering,* 28: 1-21.

[7] Sauter, T. 2005. Integration aspects in automation – a technology survey. In: *Proceedings of the 10th IEEE International Conference on Emerging Technologies and Factory Automation*, 19-22 September, Catania, pp. 255-263.

[8] Thomesse, J.P. 1999. Fieldbuses and interoperability. *Control Engineering Practice,* 7 :81-94.

[9] Wood, G.G. 1987. Survey of LANs and standards. *Computer Standards & Interfaces,* 6 :27-36.

[10] Felser, M. and T. Sauter, T. 2002. The fieldbus war: history or short break between battles? In: *Proceedings of the 4th IEEE International Workshop on Factory Communication Systems,* 27-30 August, pp. 73-80.

[11] Leviti, P. 2001. IEC 61158: An offence to technicians? In: *IFAC International Conference on Fieldbus Systems and Their Applications*, 15-16 November, pp. 36.

[12] International Electrotechnical Commission. 2003. *IEC 61158-1, Digital data communications for measurement and control - Fieldbus for use in industrial control systems, Part 1: Introduction.*

[13] Sauter, T. 2009. Fieldbus systems – embedded networks for automation. In: R. Zurawski ed. *Network Embedded Systems Handbook.* Boca Ration: CRC Press.

[14] Thomesse, J.P. 2005. Fieldbus technology in industrial automation. *Proceedings IEEE*, 93: 1073-1101.

[15] Kalogeras, A.P., J. Gialelis, C.E. Alexakos, M.J. Georgoudakis and S.A. Koubias. 2004. Vertical integration of enterprise industrial systems utilizing web services. *IEEE Transactions on Industrial Informatics,* 2: 120-128.

[16] International Electrotechnical Commission. 2003. *IEC 61804-2, Function blocks (FB) for process control - Part 2: Specification of FB concept and Electronic Device Description Language (EDDL).*

[17] Flammini, A., P. Ferrari, E. Sisinni, D. Marioli and A. Taroni. 2003. Sensor integration in industrial enviroment from fieldbus to web sensors. *Computer Standards & Interfaces,* 25: 183-194.

[18] Decotignie, J.D. 2005. Ethernet-based real-time and industrial communications. *Proceedings IEEE*, 93: 1102-1117.

[19] Felser, M. and T. Sauter. 2004. Standardization of Industrial Ethernet – the next battlefield? In: *Proceedings of the 5th IEEE International Workshop on Factory Communication Systems,* pp. 413-421.

[20] Höller, R., T. Sauter and N. Kerö. 2003. Embedded SynUTC and IEEE 1588 Clock Synchronization for Industrial Ethernet. In: *Proceedings of the 9th IEEE International Conference on Emerging Technologies and Factory Automation,* pp. 422-426.

[21] Sauter, T. and C. Schwaiger. 2002. Achievement of secure Internet access to fieldbus systems. *Microprocessors and Microsystems,* 26: 331-339.

[22] Treytl, A., T. Sauter and C. Schwaiger. 2004. Security measures for industrial fieldbus systems with special respect to IP. In: *Proceedings of the 5th IEEE International Workshop on Factory Communication Systems,* pp. 201-210.

[23] Schwaiger, C. and A. Treytl. 2003. Smart card based security for fieldbus systems. In: *Proceedings of the 9th IEEE International Conference on Emerging Technologies and Factory Automation,* pp. 398-406.

[24] Peschke, J., D. Reinelt, Y. Wang and A. Treytl. 2006. Security in Industrial Ethernet. In: *Proceedings of the 11th IEEE International Conference on Emerging Technologies and Factory Automation,* pp. 1214-1221

[25] Khan, B.A., J. Mad and A. Treytl. 2007. Security in agent-based automation systems. In: Proceedings of the *12th IEEE International Conference on Emerging Technologies and Factory Automation,* pp. 768-771.

[26] Brainin, E., D. Dietrich, W. Kastner, P. Palensky and C. Rösener. 2004. Neuro-bionic architecture of automation – obstacles and challenges. In: *Proceedings of IEEE Africon*, pp. 1219-1222.

[27] Dietrich, D., W. Kastner and H. Schweinzer. 2004. Wahrnehmungsbewusstsein in der Automation – ein neuer bionischer Dekansatz. *Automatisierungstechnik,* 52: 107-116.

[28] Solms, M. and O. Turnbull. 2002. *The Brain and the Inner World.* New York: Other Press.

6 Applications of Agent Systems in Intelligent Manufacturing

Aleksey Bratukhin[1], Arndt Lüder[2] and Albert Treytl[3]

[1] Austrian Academy of Sciences, Research Unit for Integrated Sensor Systems, Viktor Kaplan-Strasse 2, 2700 Wiener Neustadt, Austria, e-mail: bratukhin@fiss-oeaw.at

[2] Otto-von-Guericke University of Magdeburg, Faculty for Mechanical Engineering, Institute for Ergonomics, Manufacturing Systems, and Automation, Universitätsplatz 2, 39106 Magdeburg, Germany, e-mail: arndt.lueder@ovgu.de

[3] Austrian Academy of Sciences, Research Unit for Integrated Sensor Systems, Viktor Kaplan-Strasse 2, 2700 Wiener Neustadt, Austria, e-mail: treytl@ict.tuwien.ac.at

Abstract Facing the fast growing needs regarding flexibility and adaptability of manufacturing systems, the decentralization of manufacturing execution control has attained high importance. Hence, in recent years different research and development activities have tackled the problem of decentralizing manufacturing execution control and implementing these decentralized systems within control architectures. One major result of these activities is a set of design patterns describing possibilities for decentralization including the description of major entities and interaction schemas.

These activities have also shown that agent systems are an appropriate means for the implementation of decentralized manufacturing execution control systems. They cope with decentralization by nature and (especially in the case of the Foundation for Intelligent Physical Agents (FIPA) - compliant agents) provide appropriate means for the implementation of the internal behavior of entities and entity interaction.

In this chapter some major design patterns for decentralized manufacturing execution control systems are described and mapped to three major approaches with Product-Resource-Order-Staff Architecture (PROSA) and MetaMorph (both based on Holonic Manufacturing Systems) and PABADIS (Plant Automation Based on Distributed Systems), and which finally are compared. Exploiting this comparison the PABADIS'PROMISE (PABADIS based Product Oriented Manufacturing Systems for Reconfigurable Enterprises) architecture is described as an architecture trying to incorporate the advantages of the different approaches and avoid its disadvantages.

6.1 Introduction

Conventional centralized control systems face challenges in adapting to the requirements of modern production systems [1]:

- Unpredictable order flow: customer and production orders are issued dynamically, often when production has already started.

- Dynamic shop floor: configuration of the resources on the field level changes during production, which makes it difficult to plan the execution of orders.
- Complexity of the shop floor and orders: modern production systems are characterized by increasing complexity of both the complexity and flexibility required by production orders as well as the variation in the shop floor configuration.

Due to their hierarchical nature, centralized systems are highly static and difficult to adapt to such changes as late modifications of customer orders or new/broken field level devices [2]. Additionally, the decision-making process is concentrated on the top layer of the automation pyramid [usually Enterprise Resource Planning (ERP) and related systems], and therefore the production planning is hardly able to react to changes or exceptions on the shop floor.

In environments where high flexibility is required such as highly customized small lot production, the absolute optimization of production can be partly neglected in favor of flexibility. The point is that in a permanently changing environment, the lack of flexibility would make it impossible to realize any optimization mechanisms.

Distributed control systems aim to solve such problems by providing two general mechanisms:

- Moving the decision-making process from ERP down to the Manufacturing Execution Systems (MES) layer, which has a shorter planning horizon and, hence, is able to react to the changes faster.
- Distributing control over a set of independently acting entities that take some responsibility for fulfilling the order. This provides concurrent processing and, therefore, minimizes the drawbacks of the hierarchical structures.

In order to provide higher flexibility of production control and planning, the paradigm of Distributed Control Systems (DCS) was defined and further developed into several concepts and architectures. Most notably is the Holonic Manufacturing Systems (HMS) [3] concept and its architectures such as Product-Resource-Order-Staff Architecture (PROSA) [4] and MetaMorph I and II [5].

The general idea of DCS is that the decision-making process as well as system functionalities are distributed among independently acting entities called "holon" in HMS. More commonly the concept of an "agent" can be used [6].

Multiagent Systems (MAS) became de facto a standard for DCS applications and has found slowly its way into the domain of industrial automation. Nevertheless, there are a few challenges the agent systems face in order to adapt to centralized and homogenous ERP systems. It seems unrealistic that in the near future the ERPs will also be able to adapt to the distributed paradigm. Therefore, the main burden for establishing a DCS lies with the MES layer.

6.2 Existing MES Solutions

The whole spectrum of solutions in the distributed automation system developed and researched ranges from partly centralized to totally distributed approaches. Some of them provide the complete architectures, integrating not only MES but also enterprise- and control-level systems. Others provide only supportive mechanisms that allow conventional systems to fit the requirements of the modern industry.

Most of the distributed MES architectures use multiagent technologies [7] and the term "agent" in particular, despite some principal differences in the origins of the MAS [6].

The most known architecture, PROSA, is a holonic reference architecture based on three types of basic holons: Product, Order and Resource [8, 9]. Additionally, typically to all distributed architectures, the need to observe the shop floor and provide an interface to enterprise-level systems evolved into the creation of a fourth special type of component. In PROSA, the Staff holon assist and supervise the basic holons. Another architecture based on the HMS concept – MetaMorph and the follow-up MetaMorph II – uses a mediatorcentric federation architecture for intelligent manufacturing [10]. It uses a term mediator that provides communication mechanisms to different systems and components, and takes over some of the functionalities of the MES. Yet it must be noted that the original Metamorph is more an integration tool for other systems rather than a complete solution.

Eventually, MetaMorph evolved into other multiagent architectures, that attempt to integrate the functionalities of a manufacturing enterprise within a distributed environment. For instance, Agent-Based Manufacturing Enterprise Infrastructure (ABMEI) is a hybrid agent-based architecture combining the mediator and the autonomous-agent approaches [11].

Similar agent-based architectures with slightly different implementation focuses are Autonomous Agents at Rock Island Arsenal (AARIA) [12] and Manufacturing Control Systems Capable of Managing Production Change and Disturbances (MASCADA) [13]. Such architectures as HOLOS/MASSIVE [32] and its followup Decentralized Decision-Making and Scheduling (DEDEMAS) [14] concentrate on decentralized decision making and scheduling.

Architectures like Advanced Fractal Companies Use Information Supply Chain (ADRENALIN) [15, 16] consider agent-based resource brokering functionality enabling a decentralized manufacturing order navigation based on local optimization strategies.

Opposite of that, Plant Automation Based on Distributed Systems (PABADIS) and its followup PABADIS'PROMISE (PABADIS based Product Oriented Manufacturing Systems for Reconfigurable Enterprises) provide agent-based architectures that cover the whole automation pyramid but with a primary goal of distributed MES [17].

6.3 A Generic Design Pattern for Manufacturing Execution Control

The organization of an MES and its interconnection to the ERP and field control layers has some general patterns that are followed by most of the solutions implementing MES in a distributed fashion. As a starting point these basic design pattern exploited in MES system design will be presented and exploited later on for the comparison of HMS and PABADIS approaches.

Design patterns are a widely used design aid in information sciences. They date from the initial work of Alexander in 1979. Alexander, who was an architect, came to the conclusion that building design always follows the same basic rules with culture and geography dependent implementations [18].

With the emergence of object-oriented programming the idea of design patterns as application-independent basic design principles has been adopted to software design in information sciences. The main research activity initiating this trend was the work of Gamma et al. [19].

Within the last few years the use of design patterns has been integrated in several disciplines including the design of control applications. Within control design the application of design patterns as a description of basic design principles has been extended to the design of complete control applications including control software and hardware, as well as the plant itself. Within this field valuable progress has been made. Overviews of the reached results are given in [20, 21, 22].

Within the field of distributed control systems the application of agents has gained wide acceptance [6]. That is why design patterns have act in this field as well been recognized and described [23, 24]. This chapter intends to sketch the MES-related design pattern briefly and describe how it occurs and is applied within the different agent-based approaches.

Within MES systems there are two main general entities with three categories of knowledge that are relevant for the execution process of manufacturing orders. These entities are order and resource usually accompanied by the categories product knowledge, order knowledge, and resource knowledge.

Equally, Resource Agents (RAs) are associated with production units of different types. Each agent represents one resource with certain production capabilities it provides to the agent community containing all resource-related knowledge. It is responsible for all resource-related control decisions including resource scheduling (based on cooperative scheduling algorithms) and resource control on the field control layer but also supportive actions like maintenance or life cycle activities.

Facing these entities and knowledge distributions, the overall system behaves in the following way. All agents in the system form an agent community containing all Order Agents (OAs) and RAs. This agent community also contains a special registration function for RAs. At startup of an RA it registers its manufacturing capabilities, which are represented by usable production functions, within the agent community. These production functions can then be searched, initialized, parameterized, started, and stopped by OA during the OA–RA interaction.

On the other hand, each OA covers a product order consisting of a set of production steps that are necessary to produce the ordered product. These production steps can be fulfilled by a set of production processes. The OA then has to negotiate for appropriate manufacturing capabilities offered by the RA to fulfill its order. Important and helpful with respect to flexibility, OAs and RAs both handle its own schedules and make its own scheduling decisions independently yet in a collaborative way. The resulting structure is given in the following class diagram and described in what follows (Fig. 6.1).

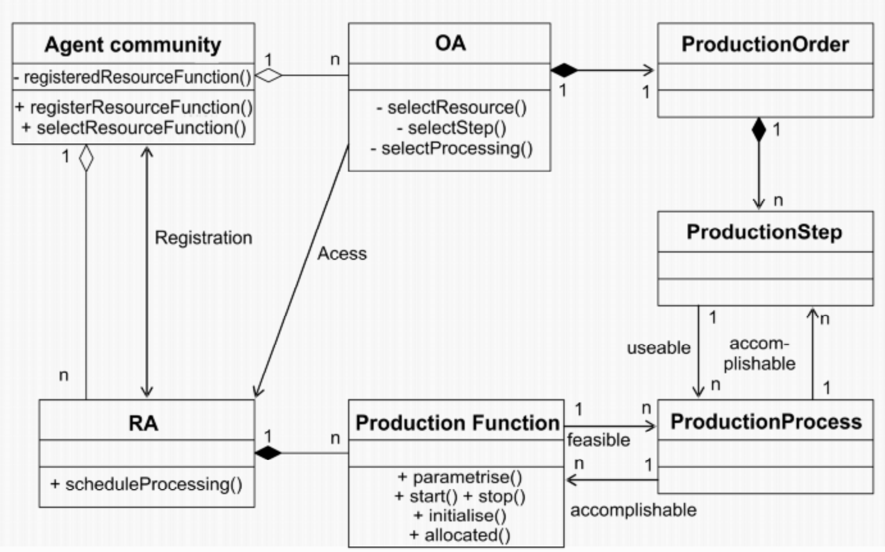


Fig. 6.1 Order-resource pattern

During the start of an RA, the RA makes itself known within the agent community. In this way, the agent community is informed about all production functions belonging to and being controllable by the RA. Each OA processes its production order step by step following the interaction scheme among OA and RA given in the next figure (Fig. 6.2).

1. The OA selects the production step to be executed next.
2. The OA determines within the knowledge of the agent community the set of RAs containing a production function that can be used for the next production step of the OA.
3. The OA negotiates with the determined RA set the next production function to be used and the usage schedule. Therefore, the OA asks all RAs for possible schedules, decides about the next production function to be used, and allocates

this function. The RA related to the selected production function reserves resources for this within its schedule.

4. At the agreed upon moment the OA accesses the RA to parameterize and start the expected production function.
5. If the processing is finished, then the processing function provides the processing result to the OA by using the RA. The OA can adjust its list of necessary production steps and proceed further with step 1.

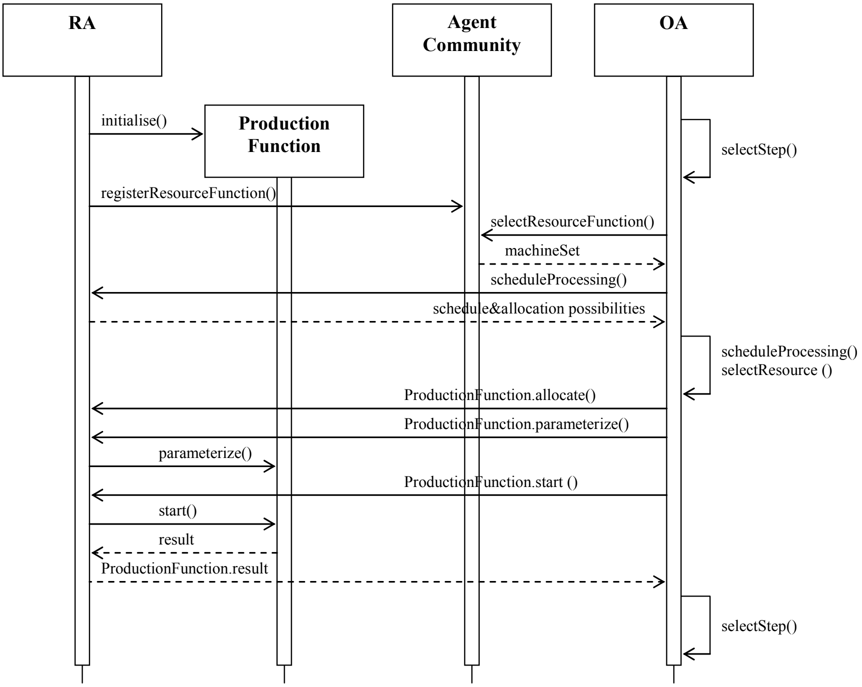


Fig. 6.2 Order and resource agent communication pattern

6.4 Distributed Approaches Analysis

Because different MES solutions usually focus on a specific area of implementation, e.g., scheduling, customer support, resource utilization, system integration, they often cannot be compared. For this chapter the focus is on systems that cover the complete manufacturing process. Hence HMS-based concepts, such as PROSA and METAMORPH, and MAS-based concepts, such as PABADIS and PABADIS’PROMISE, that cover wider fields of control systems and focus on the vertical integration of all layers of the automation pyramid from the ERP down to field control level are compared in the following chapter.

In order to have a unique name for the analysis the terms used by the HMS have been chosen because they are widely known. Furthermore, if HMS, which does not provide an architecture, lacks the required terms, PROSA terms (Order, Resource, and Stuff holons as well as MetaMorph mediators) are used to compare with PABADIS architecture with its respective Product, Residential and Plant Management Agents (PA, RA, and PMA).

6.4.1 Resource Holon and Residential Agent

Generally speaking, there are two main elements of distributed plant automation systems: resources and customers represented by orders as given in the above design pattern.

Resources in HMSs are represented by Resource Holons, which are responsible for machine-level representation. The Resource Holon consists of a physical part, namely, a production resource in HMS, and of an information processing part that controls the resource. This second part holds the methods to allocate the production resources, and the knowledge and procedures to organize, use, and control the physical production resources to drive production.

In terms of PABADIS, Resource Holon is a Co-Operative Manufacturing Unit (CMU) which is a building unit of the shop floor. The information processing part of the Resource Holon perfectly fits the concept of the Resource Agent, or Residential Agent as it is called in PABADIS, which represented the CMU and partly controls the function.

The main differences are that in PABADIS the Residential Agent is more or less an interface between the agent community and the CMU (machine, function unit), and the Residential Agent has a generic interface enabling the integration of an arbitrary machine hardware. In HMS the Resource Holon is more than this interface. It is rather a CMU-RA analog, where the whole functionality of the CMU (function, control level, agent communication level) is implemented. PABADIS distinguishes the standard (logical) part of the control level, which can be used for each CMU, from the "machine"-specific part, which has to be implemented by the customer of the system. This makes the system more flexible and encapsulates the MAS from the control level. That makes the process of adding new functionality easier, because the customer (industrial company) has to add a specific plant-dependent resource to the system and does not care about the interoperability of a new component with the existing control system. The customer just follows the interface, and incorporation into the system is provided by the generic Residential Agent, which is able to communicate within the system.

Additionally to this, Resource Holons do not just provide resources, but also manage the whole production facility. This means that they are able to communicate between each other in order to find the best use of machines. PABADIS does not allow RAs to communicate with each other. This is done in order to make the system product oriented and not machine oriented, which may be the biggest difference from the HMS concept. The product and its performance are the main goal of the system in PABADIS. HMS is more focused on machine utilization, where an Order Holon is simplified to a set of product parameters that has to be produced.

6.4.2 Order Holon and Product Agent

Compared to Resource Holons and Residential Agents the difference between definitions of the Product Agent in PABADIS and the Order Holon in HMS is stronger:

- A Product Agent is an instance that manages the whole production of a single workpiece. It bases its decisions, actions, and knowledge on the so-called Work Order (WO), which gives a full specification of the production activities regarding the tasks that have to be fulfilled in order to complete the product. At the same time, the WO does not assign exact machines; instead it describes the function that has to be used to do that. This principle gives the Product Agent the freedom to decide what machine to use and introduces the possibility of changing the machine during execution.
- An Order Holon is much simpler. It is more or less the customer request, where the requirements to the product are defined. It has an advantage in mass production, because an Order Holon is not attached to a single workpiece. An Order Holon does not do scheduling or resource allocation, because it simply does not have knowledge about the physical layout of the plant and the product specification.

6.4.3 Product Holon and Product Agent

The necessity of the Product Holon rides on the technology dependence of the product. That means that the Order Holon, representing the workpiece, does not have knowledge about the technological properties of the plant. This means that it has no information on how to produce the product. Order Holons contact the Product Holon in order to get information: how to produce this product. Based on the Product Holon's response, an Order Holon performs the product execution.

The standardization of the functions provided by the CMU's concept gives a generic description of the functionality that does not depend on the specific technology.

In PABADIS no analog to the Product Holon is required. Product Agents incorporate both Product Holons and Order Holons. The advantage of the Product

Holon is that the new product specification can be added to the system at any time, but on the other hand Order Holons cannot adapt their behavior to the unplanned changes in the system because they have no knowledge about the product performance. Product Agents have a complete specification of the product execution and knowledge how to analyze these data.

6.4.4 Stuff Holon and Plant Management Agent

The Stuff Holon fits the definition of the PMA in PABADIS. The main purpose of the Stuff Holon is to give the other components an overview of the system. Therefore, it is often used for supervision or support functionality implementation.

One of the typical main MES functions represented by this concept is scheduling. For example, PROSA uses a centralized unit called the Scheduler Stuff Holon that performs scheduling for the whole plant.

In PABADIS the scheduling is simplified and distributed in Product Agents. In contrast, this approach makes the system more flexible, yet it must be noted that it does not provide the optimal solution. In PABADIS, scheduling is done from the point of view of a single product, which makes it impossible to guarantee the optimum. PA negotiations and the benevolent behavior of Product Agents can increase scheduling performance, but this still does not guarantee the global optimum.

PABADIS minimizes the number of new instances by using existing basic architecture. That makes the components, basically Product Agents, more complex but keeps the architecture simple. In contrast, PROSA introduces the new instances and keeps the basic holons simple.

What is common to both architectures is that they try to avoid centralized units, which PMA and Stuff Holon are by definition.

6.4.5 Aggregation

Aggregation is a key point in HMS. Aggregation is structuring agents in a hierarchy. This is the appropriate solution to tackling complexity of independent Holons. This solution avoids complex communication and heavy network load in the system, but reverts back to the centralized systems, causing a loss of system flexibility and scalability. Aggregation introduces a new layer(s) in the control pyramid and makes the logic of holons more complex. In practical implementation even communication is not simpler because holons have to communicate on different layers, which brings complexity to the process in general yet simplifies one-to-one communications. A tradeoff between simpler single communication mechanisms and additional overhead due to aggregation has to be found.

PABADIS does not use aggregation of agents and tries to avoid the complexity of communication via giving the single instances more independence, which decreases the necessity of communication. This leads to a reduction in optimization yet efficiently increases system flexibility and scalability, which is the main goal of PABADIS.

6.4.6 Mediator

It is difficult to compare PROSA or PABADIS architectures with the MetaMorph approach. Mediators in MetaMorph are powerful tools to connect different systems together, but can hardly be considered as a manufacturing system architecture. To a certain extent, mediators behave as Staff Holons in HMSs or as PMAs in PABADIS. They provide centralized functionality to the system by coordinating the actions of other agents.

Mediators can successfully be used in cooperation with other concepts, such as PABADIS or PROSA, for the interconnection of different elements and for providing solutions for problems that require a temporal overview of the system. But individually, mediators cannot be a manufacturing-system-oriented architecture because they do not have dedicated MES functions (such as scheduling) or do not represent different actors in the plant (such as products and resources).

An attempt to create an architecture (MetaMorph II) is more or less a centralized approach with dedicated mediators for each function of the plant automation system and a strong hierarchy among them. Perhaps the most important achievement of the mediator concept is its ability to dynamically group entities into virtual groups according to the needs of the system. This ability can greatly improve the flexibility of a system, not only for MES applications.

6.4.7 Flexibility Versus Optimization

In conclusion of the previous sections, it can be said that PABADIS architecture is particularly usable in a highly turbulent environment where flexibility is crucial to system performance; PROSA is more suitable for systems were optimization is more important and flexibility is secondary.

Due to the fully distributed MES layer, PABADIS is more scalable than HMS since the latter has the remains of the centralized structure where each function is dedicated to a single entity. The same argument can be applied to the MetaMorph approach where, instead of distribution of intelligence, the intelligence is focused on the function-representation mediators.

Both PABADIS and PROSA provide comprehensive solutions to the problems in existing MES systems face, trying to adapt to modern trends in manufacturing, in particular mass customization. These trends require more flexible production management and planning to react to the turbulence in customer demand and shop floor configuration. They shift decision making from the level of business systems (such as ERP) to the middle layer of the MES. Therefore, they shorten the reaction time to changes in the plant and provide vertical integration of the automation pyramid, making it possible to bring the strategy made at the top level to the field level of control devices.

Both architectures provide distribution of the MES system but approach this problem from different sides. PROSA tries to imitate conventional centralized systems by providing entities for different functions. In contrast, PABADIS dissolves the functions of MES in the community of agents. The actual difference between the approaches lies in the balance between optimization and flexibility. It is clear that PROSA provides a higher level of optimization compared to PABADIS. It is also clear that PABADIS is more adaptive, scalable, and flexible than PROSA. PABADIS'PROMISE architecture developed later, attempts to find a balance between the above-mentioned concepts.

6.5 PABADIS'PROMISE Hybrid Approach

PABADIS'PROMISE is the followup of the PABADIS architecture that nevertheless combined the key features of all three afore mentioned concepts:

- General architecture PROSA and PABADIS - Order and Resource concept;
- Distributed functionality approach of PABADIS;
- Aggregation of resources of PROSA;
- Clustering and mediation of resources for scheduling of MetaMorph.

Being a followup of the PABADIS architecture, PABADIS'PROMISE comprises the general notions of Order and Resource agents used by PABADIS but improves its functionality by introducing concepts developed by the holonic architectures of PROSA and MetaMorph.

6.5.1 Resource Handling

Hierarchy of resources is a key distinction point of distributed concepts. On the one hand, such architectures as PROSA provide strict control of resources over each other. It improves resource utilization but makes it complex to implement an actual system and has problems in adapting to a changing environment. On the other hand, systems such as PABADIS make resources totally independent of each other (for instance, PABADIS Residential Agents do not even communicate with each other).

The PABADIS'PROMISE approach to Resource Agents is similar to the MetaMorph approach in terms of scheduling by the usage of clustering of related resources and makes it possible for Resource Agents to allocate other resources for certain tasks. This approach improves the flexibility of the shop floor compared to PROSA, because there is no direct static connection between two resources, and optimizes system performance by including Resource Agents in the decision-making process during scheduling.

6.5.2 Order Management

Another important aspect is order management. As with the other topics, there is always a balance between optimization and flexibility and the hierarchy of orders and management entities related to them. Where PROSA orders are organized in a rigid hierarchy and PABADIS has its minimization of interdependencies between orders, PABADIS'PROMISE provides what can be called "implicit hierarchy." There is no strict decision making, the organization is implemented via the structure of the Production Order that possesses interdependencies between different production steps (called Process Segments) via so-called node operators. The mechanism of order decomposition developed in PABADIS'PROMISE allows for the on sign autonomously acting Order Agents to each Process Segment. It is similar to the PABADIS approach with the difference that higher-level Order Agents can influence the agents responsible for subtask execution. For instance, the Order Agent that is responsible for the assembly of a car needs an engine to be produced before performing its tasks. Therefore, it assigns the deadlines for the engine agent, so the engine is delivered on time to assembly the complete car.

Another issue is the behavior of agents involved in production. PROSA with its hierarchy does not need to pay attention to this topic, but in more distributed systems such as PABADIS it has greater importance because it can increase performance efficiency. Product Agents in PABADIS are selfish in their "nature." As a result the system has low optimization of order execution compared to PROSA but much higher flexibility. PABADIS'PROMISE again tries to combine both benefits by introducing benevolent behavior of agents that is, agents have to obey a certain set of rules that benefit not only their local goals (finishing assigned Process Segments on time), but also to sacrifice their goals to help agents struggling to meet their deadlines. This approach considerably improves optimization without dramatically reducing flexibility, which guarantees higher overall system performance and stability.

6.5.3 Supervisory and Supporting Functionalities

As was mentioned above, both PROSA and PABADIS define a special entity (Stuff Holon and PMA, respectively) to cover functionality not reflected by order- and resource-related agents or holons.

PABADIS'PROMISE also provides supervision management entities, namely, Order and Resource Agent Supervisors (OASs and RASs) to implement functions that are centralized by nature such as supervision or reporting. But in contrast to the earlier architectures, PABADIS'PROMISE does not define a specific entity that deals with such functions as tooling or maintenance. Instead of creating an extra agent, the concept incorporates support functions via ontology and organization of agent communities that allow for implementing tooling or maintenance using the general mechanism of an order-resource relationship.

6.5.4 PABADIS'PROMISE Scheduling

The advantages of PABADIS'PROMISE as a hybrid approach compared to PROSA and PABADIS architectures (MetaMorph is more a concept than an architecture) can be summarized in the way it approaches scheduling. Instead of the single centralized concept of PROSA and the selfish order-oriented approach of PABADIS, PABADIS'PROMISE comprises resource-oriented scheduling on the shop floor with dynamic clustering for solution finding, and order-oriented benevolent rescheduling on the MES layer.

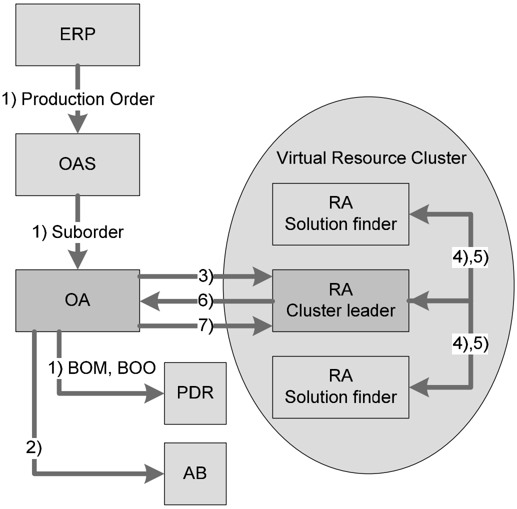


Fig. 6.3 Resource-oriented scheduling

Scheduling in PABADIS'PROMISE can be divided into two mechanisms: resource-oriented initial scheduling and order-oriented rescheduling [25].

The first mechanism (Fig. 6.3) consists of the following steps:

1. OA retrieves its first unscheduled Process Segment (PS);
2. OA asks the Ability Broker (AB) for an RA with the capability required by the PS;
3. OA requests an RA for allocation;
4. RA forms a cluster for scheduling;
5. RAs perform scheduling and find a solution;
6. RA sends a proposal to the OA;
7. OA accepts or rejects the proposal.

In steps 1 and 2, the initial processes of the OA parsing an order and discovering resources are executed. Steps 4 and 7 are negotiations between an OA and an RA. And the actual scheduling algorithm is applied in steps 4-6. Although the RAs implement scheduling algorithms for finding a solution to reserving resources, the OA is responsible for decisionmaking. That is in contrast to the MetaMorph original concept, where the cluster leader's decision is final. The reason to shift the decisionmaking to the OA is to provide optimization of the order execution and not just of resource utilization. In what followings the seven steps of the scheduling procedure are described in more detail.

6.5.4.1 Order Agent Receives the Production Order and Parses It

Depending on the "depth" scheduling, meaning the number of PSs scheduled in advance, the OA fills in the time schedule to create the framework for the execution of the Production Order (PO). The reason behind introducing "depth" is to avoid the snowball effect of the network overload caused by the rescheduling of the previously allocated task due to the requests from the higher-priority OAs.

6.5.4.2 Order Agent Asks the Ability Broker for Resources

In the next step, the OA sends a request to the AB that maintains the actual list of abilities and resources required by the OA to carry out the order. In PABADIS'PROMISE, ability is a certain function that a resource provides. It can be a physical operation, computation function, or an action of a human. It reflects the definition of a resource in the concept that varies from a robot to the entire production line or a plant and can even be a human being.

6.5.4.3 Order Agent Asks a Resource Agent for Allocation

After receiving the RA address that can perform the requested ability, the OA sends a scheduling request that contains the Process Segment and the time slot the OA desires for the ability execution.

6.5.4.4 Resource Agent Forms a Cluster for Scheduling

Upon receiving the request, it is the task of the RA to communicate to other identical resources and form a resource cluster. This communication can follow the sequence of actions as proposed in MetaMorph that is, the leader can first broadcast a message to all similar RAs, then in reply the RAs can join in the cluster, and this cluster can then participate in the process of scheduling under the leadership of this leader. However, the difference from the classical MetaMorph concept is that in PABADIS'PROMISE the leader does not have the pointers through which other identical resources can be accessed, therefore it needs to find out those pointers first. In order to find the similar abilities and respective resources, the cluster leader RA contacts the AB and receives the pointers. Then the leader broadcasts the request for clustering to the RAs with the same ability. In response, all the RAs to which a request was send evaluate the request message and reply to the leader about their decision; the request is then either accepted or dropped. The decision is based on the availability of a resource, meaning that if a resource is already allocated for a certain period, then it will not participate in the cluster. After that leader receives the responses, it forms a virtual cluster from the resources that responded positively. The important feature of the virtual clusters is their dynamism, meaning they are created on demand and are not permanent. A cluster is created for an order and then can be broken after completion of the scheduling activity of the order. Moreover, it is also possible that an agent that is participating as a leader in one cluster will also act as a participant in another cluster.

6.5.4.5 Resource Agents Perform Scheduling and Find a Solution

When cluster is formed, the mechanism of finding a quasioptimal solution starts. Within this mechanism a task leader asks the agents in the cluster for their proposals regarding the requested task execution and evaluates the results. There is also a local internal evaluation process at the RAs using the so-called evaluation function that considers the availability and costs of resources, as well as tooling and waiting time due to the gaps in the RA schedule [25]. Generally speaking, the evaluation function provides an optimal solution at the particular time for a single ability with respect to resource utilization optimization.

6.5.4.6 Resource Agent Sends a Proposal to the Order Agent

After the cluster leader RA receives all the bids from the cluster members, it verifies the proposals, and based on additional parameters given by the OA or general for the shop floor, chooses the best solution.

The general parameters for the shop floor are those that concern the optimization of the whole field layer and not focused on the local optimization of a single resource. Finally, the RA sends a proposal to the OA that had send the scheduling request.

6.5.4.7 Order Agent Accepts or Rejects the Proposal

Then it is up to the OA to evaluate the proposal and to accept or reject it. If the OA accepts the proposal than it allocates the resource on the proposed conditions. If the proposal is not suitable, then the OA can proceed in two possible ways:

- Starting the process from the beginning, meaning asking the AB for the given ability, asking the RA to form a new cluster, and so on. Due to the dynamism of the shop floor and of the order flow, the result of the new evaluation can be different from the old one.
- Asking the cluster for other solutions that fit the OA in a better way, but are less optimal for the resources. This mechanism depends on the configuration of the system that has to evaluate the importance of the shop floor optimization compared to the order flow optimization.

Eventually, if a solution cannot be found within the resource-oriented scheduling then the OA has to negotiate with the OAs or OASs to find a solution. This mechanism is based on the benevolence of the OA behavior, assuming that the OAs respect the needs of the others. An exact criterion for evaluation depends on a particular application, but for the actual implementation of the demonstrator the criterion for rescheduling is based on the due deadlines and due dates of the orders.

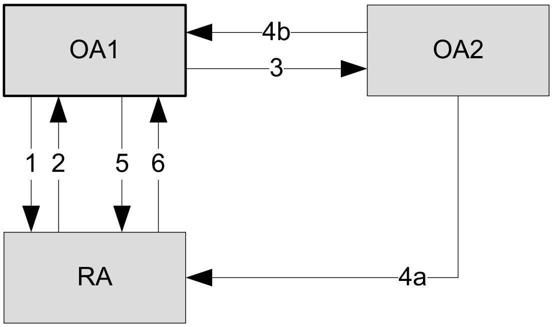


Fig. 6.4 Order-oriented rescheduling

Each order has a deadline and due date that are given to the OA by the ERP system and expected to be fulfilled. Each OA estimates the execution time for the order in general and for each activity in particular. Therefore, if the activity execution proposed by the RAs does not fit the deadline/due date, an OA contacts OAs that use the resources the RA is interested in. The information about such OAs is sent by the RA together with the refusal of allocation within the initial scheduling request protocol. Figure 6.4 shows the general communication mechanism of rescheduling and consists of the following steps:

1. OA1 sends a scheduling request with the possible execution time;
2. If the RA cannot find a solution that fits the required parameters, it responds with a message of refusal and informs the OA1 about the reasons for the refusal. In particular, it is the list of OAs (with the resource allocation identifications) that allocates the requested time slots of the RAs for its own activity execution;
3. OA1 sends a request to the OAs that reserved the resources. In the example request, OA1 asks OA2 to cancel a particular resource allocation and provides the deadline/due date of the order of OA1;
4. Depending on the applied constraints for the plant, OA2 evaluates the possibility for rescheduling. For instance, if OA1's deadline is in 1h and deadline OA2's is in 1 day, than the OA2 agrees to cancel a requested reservation. Before informing OA1, OA2 sends a reservation cancelation request to the RA and informs the RA that the cancellation is being done for another OA, namely, OA1. Therefore, the RA only allows OA1 to use this time slot;
5. OA sends a scheduling request to the RA again;
6. The RA confirms the reservation.

6.6 Summary

From a general point of view PABADIS'PROMISE, PROSA, and PABADIS can be summarized with respect to the following criteria:

Autonomy and aggregation: on the one hand, PROSA lacks flexibility due to the direct control over aggregated entities, and on the other hand, PABADIS lacks optimization because of total distribution. PABADIS'PROMISE defines Production Order decomposition as something that establishes rules of controlling autonomous entities without dramatically reducing flexibility.

Cooperation and hierarchy: where PROSA has an explicit hierarchy that causes rigidness of order changes, and PABADIS provides no hierarchy that implies overhead in the case of managing complex products, PABADIS'PROMISE offers implicit hierarchy (Production Order decomposition; flexible structure of orders; dynamic control of resources by resources) that finds a balance between two approaches.

Decisionmaking: while decision making in PROSA is centralized, meaning there is one control entity per functionality, PABADIS supports a completely distributed decision-making mechanism. PABADIS'PROMISE provides a semi-distributed approach based on clustering of resources at the shop floor and PO decomposition at the MES layer.

Data interoperability: with PROSA having implementation-specific data that cause difficulties for the system installation and PABADIS data that virtually do not have an established connection to the ERP system, PABADIS'PROMISE ontology covers the entire automation pyramid linking all three layers together.

Control flow: on the one hand, PROSA has a strict vertical control flow that lacks feedback to the upper layer of the ERP. PABADIS, on the other hand, has a limited feedback to the ERP, providing it only at the end of the production cycle. Therefore, PABADIS'PROMISE supports a permanent connection with the ERP via planned periodic or event-based reports during the Production Order life cycle.

Table 6.1 Distributed approaches comparison

	PROSA	PABADIS	PABADIS'PROMISE
Autonomy	Low	High	High
Aggregation	High	Low	Production order decomposition
Cooperation	Low	High, selfish	High, benevolent, dynamic resource control
Hierarchy	Explicit	No hierarchy	Implicit, flexible order structure
Decision-making	Centralized	Distributed	Distributed, shop floor clustering
Data interoperability	Implementation-specific data	Limited ERP connection	Common ontology throughout the system
Control flow	Vertical	Horizontal	Bidirectional
ERP feedback	No feedback	Limited feedback	Periodic and event-based reports

In conclusion PABADIS'PROMISE has achieved a balanced combination of the rigid PROSA architecture and the chaotic PABADIS approach with vital contributions from of MetaMorph dynamic optimization.

All the above-mentioned architectures approach manufacturing automation from the conceptual points of view and often overlook the application aspects that are brought by the distributed nature of the concepts. In particular, security, product identification, and data interoperability are vital aspects for practical implementation.

Distributed systems lack single point of control with decision making spread over the multiple entities that communicate with each other. This causes higher security risks, due to the intensive communication that such architectures require and the fact that there is no single entity that controls the system.

Another challenge of distributed systems is the lack of general overview of the processes and components on the shop floor. It is difficult to keep track of the products, work in progress, and materials in the plant, and it is often impossible to say where exactly a specific piece is located. Therefore, more advanced identification of the workpieces is required to guarantee the efficient operation of the system.

Last but not least, distribution of decision-making functionality requires interoperability of information flow over all three layers of the automation pyramid as well as mechanisms of distributed databases. Therefore, a common ontology with mechanisms of data abstraction for different control entities is required to guarantee the coordination within the system.

6.7 Practical Implementation Aspects

In order to address the practical implementation issues raised in the previous section, PABADIS'PROMISE defines features of data security, material identification, and consistent data management. The following sections describe the chosen solutions, namely, a three-zone security architecture, Radio Frequency Identification (RFID)-based product tracking, and XML-based automation ontology.

6.7.1 MES Security Architecture

Often overlooked by developers as a less important issue, a missing security concept always strikes back when it comes to actual implementation. It is especially vital in a distributed environment when the use of standard IT technologies, a collaborative agent system, and low-resource RFID devices introduces new threats to the usually closed automation environments. Security threats from the agent system point of view are:

- Modification of agent data and code during transmission;
- Abuse of a platform by a malicious or strayed agents including authentication theft;
- Misuse of resources (unauthorized access) or wrong pairing of entities, i.e., loss of origin or untraceable unitary (unauthenticated communication).

The PABADIS'PROMISE architecture integrates the security needs of the two loosely coupled agents with the main focus at the control and MES layers but also the interfaces to the surrounding layers.

The problem of securing a distributed MES is the use of devices with low resources in the MES layer and the field layer that are not capable of carrying the additional load of (strong) security measures. Hence, a hierarchical security concept is applied that is organized in three zones of different mutual trust. This is a common approach in industrial automation [26, 27].

For the purpose of the PABADIS'PROMISE system a security system with three main zones is a suitable solution (Fig. 6.5). The three zones match the three functional areas of MES: high-layer components of ERP; manufacturing execution with order and Resource Agents, and interfacing between the enterprise layer and the MES. Further subzones, called local and functional domains, are introduced that encapsulate operations such as real-time communication that conflict with the usual security measures.

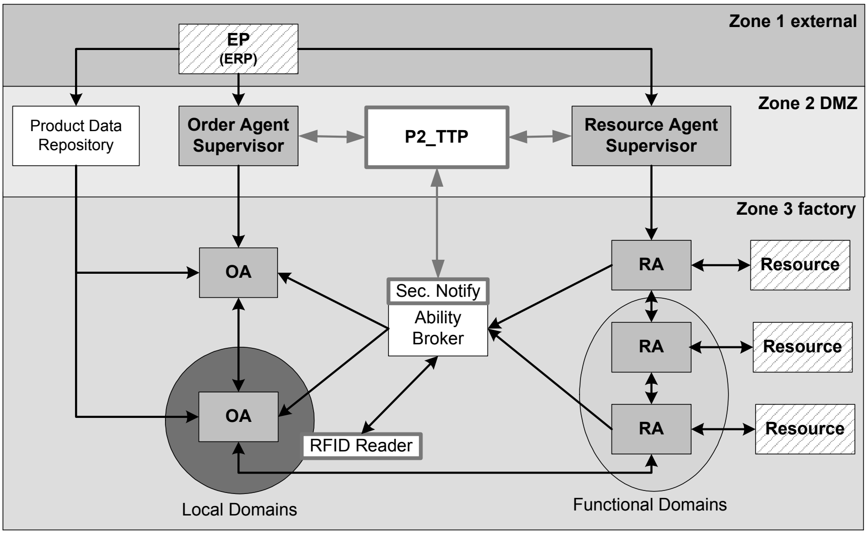


Fig. 6.5 Three-zone security model

The topmost is Zone 1: external meaning outside of the MES layer, including some ERP components of the enterprise layer.

In the middle of the three-zone model is Zone 2: Demilitarized Zone (DMZ). In the DMZ are located the supervising entities RAS, OAS, and PDR but also security management entities such as a Trusted Third Party (P2_TTP). These entities translate the semantics and syntax of the ERP payload to the factory layer. They are also responsible for establishing secure connections to the ERP, which is mainly based on standard Internet technologies such as SSL/TLS and XML encryption for Web services security to the limited security operations of the factory floor.

At the bottom of the three-zone model is the factory zone is placed. Due to hardware limitations of the embedded systems running OAs and RAs they are neither capable of doing strong computations nor can deal with high communication loads. For example, OAs are usually run on embedded devices that move around with their associated products and are often connected only through RFID communications. Hence, within the third zone only less resource consumption and, therefore, usually rather week security measures exist for authentication and access control. Nevertheless, overall security is maintained since the entrance to the zone is protected by strong security measures in the zone(s) above. Each data or request for operation must pass all zones on the way to its destination to allow weak authentication and encryption inside the inner zone, e.g., a request from the ERP first has to pass the Web-service security, then the checks at the OAS, the firewall to the factory zone, and the authentication inside the factory zone. If a security check fails, than the requested operation is not permitted and an exception handling takes place that is part of the P2 protocols.

The PABADIS'PROMISE security model allows one to apply a defense-in-depth concept that enables the system engineer to integrate weak components such as RFID tags and low resource Programmable Logic Controls that are not capable of implementing heavy security functions.

6.7.2 Radio Frequency Information Technology (RFIT)

The general problem with distributed system architectures is their abstraction from the real world, and that comes with a price when applying them to applications. One of the main challenges is tracking products, materials, and work-in-process pieces that are managed by purely software-based entities, namely, holons or agents.

PABADIS'PROMISE uses RFID tags to track the physical pieces in the plant and provides two types of RFID tags (PIT and PAHT): Product Identification Tag (PIT) – simple passive RFID tags used for identification of materials – and Product and Agent Host Tag (PAHT) – active RFID tag equipped with a Foundation for Intelligent Physical Agents (FIPA)-compliant agent platform that is used to identify products and host OAs that manage their production.

In the latter approach, RFID tags not only contain the product identification and product data, but also run an agent host providing an agent platform for the OA it hosts. It provides an environment for an agent to perform its action on the tag. This confers a considerable advantage in performance, because the OA is running constantly but also requires more tag resources (memory, processor), due to the fact that it has to run an agent host. Because standard agent platforms such as JADE [28] require a Java Virtual Machine (JVM) that consumes a lot of memory and processor capacity, PABAIDE'PROMISE developed a C-based agent environment called CARE (C-Agent Runtime Environment) that provides sufficient agent functionalities and is FIPA-compliant [29].

The main advantage of PAHT is that RAs do not need to retrieve product identification or product data at all, because the product and the OA are the same entity. There is no need for product identification at all because OA identification for PAHT is sufficient [30].

Both the physical and the logical connection of a product and an agent are archived, which gives a higher level of flexibility to the agent and product. Products can be freely moved from one environment to another, without regard to the actual location of the associated agent.

Due to the fact that products and their agents are the same entity, there is no need for additional intelligence of a tag other than an agent. That means that the tag does not need to analyze data or provide communication mechanisms, but rather serves as a database and as an agent host.

6.7.3 Data Interoperability

Last but not least is the challenge of data interoperability that many concepts (i.e., PROSA) shift to the realm of applications rather than including them in the conceptual core of systems.

This often leads to integration problems, especially when trying to combine ERP systems with shop floor solutions. PABADIS'PROMISE provides an ontology that incorporates standard structures that can be filled with specific data and can be decomposed in order to optimize performance and utilizes the resources efficiently. The best example of the PABADIS'PROMISE approach to data handling is the PO that is used by all three layers of the automation pyramid (Fig. 6.6).

The structure of the PO is designed to provide the possibility for its further decomposition as well as possibilities of the concurrent execution. The general description of the PO contains information about the type of product, its quantity, deadlines and due date and additional information specific for a particular instance of a product. The rest of the PO is a combination of PS and Node Operators (NOs).

The Process Segment (PS) is a basic construction element of the PO describing a single task or operation that the system has to fulfill in order to go further to complete the product. Each PS is specific to a PO but can serve as a reusable core. An Ability is a recipe of a single operation that is predefined and a set of parameters that are unique for each product or PO. In addition to the Ability description, material data are specified in a way that the consumed and produced materials are defined for each PS. That makes it possible to decompose the order into a set of suborders that can be executed concurrently, based on the principle that there is one OA that is responsible for a single piece of material/work in progress/product [31].

Finally, NOs are the logical links between different PS and combine them into a single hierarchical structure of the PO also facilitating order decomposition. There are several types of NOs declaring the type of the operator (Sequence, BranchOr, BranchAnd, JoinOr, JoinAnd) as well as input and output. There can be multiple inputs and outputs making it possible for different ways to execute the PO, which therefore gives extra flexibility to the production system by adaptation to the loss of a single type of a machine or transport line.

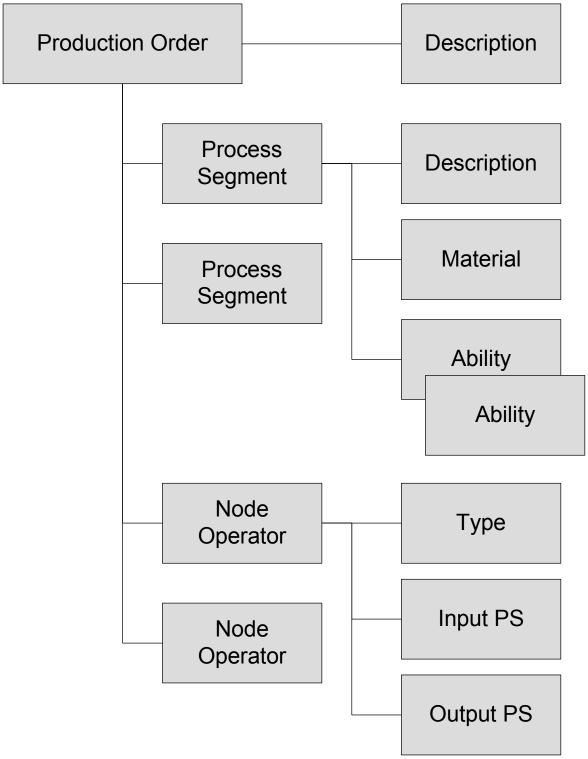


Fig. 6.6 Production order structure

6.8 Conclusion

Distributed Control Systems are a state-of-the-art approach. Based on agent technology, there are several implementations especially for the Manufacturing Execution Control layer.

In this chapter the basic structures used within the majority of approaches have been described. It has been shown using a comparative approach how they are reflected within the PROSA, the PABADIS, and the MetaMorph architectures. The benefits and drawbacks of each approach were discussed.

The PABADIS'PROMISE architecture has all the advantages of the above-named approaches, resulting in a new architecture most fitting for recent problems in factory automation on the MES layer.

It has been shown how PABADIS'PROMISE is incorporating the features and design decisions of PROSA, PABADIS, and MetaMorph. In addition, approaches addressing special requirements are addressed using the most recent technologies were presented.

References

[1] Cox, W.M and R. Alm. 1998. The right stuff; America's move to mass customization. *Annual Report Federal Reserve Bank of Dallas,* 3-26.

[2] Mönch, L. 2006. Agentenbasierte Produktionssteuerung komplexer Produktionssysteme. Wiesbaden: Deutscher Universitätsverlag.

[3] Wyns, J. 1999. Reference Architecture for Holonic Manufacturing Systems. PhD thesis, Katholieke Universiteit Leuven, Belgium.

[4] VanBrussel, H., J. Wyns, P. Valckenaers, L. Bongaerts and P. Peeters. 1998. Reference architecture for Holonic Manufacturing Systems: PROSA. *Computers in Industry,* 37: 255-274.

[5] Maturana, F. and D. Norrie. 1996. Multi-agent mediator architecture for Distributed Manufacturing. *Journal of Intelligent Manufacturing,* 7: 257-270.

[6] Shen, W., Q. Hao, H. Yoon and D.H. Norrie. 2006. Applications of agent systems in intelligent manufacturing: an update review. *International Journal of Advanced Engineering Informatics,* 20 (4): 415-431.

[7] Radjou, N. 2003. Software agents in business: still an experiment. Agent Link Magazine, issue 13, June.

[8] Bongaerts, L. 1998. Integration of Scheduling and Control in Holonic Manufacturing Systems, Ph.D. thesis, Katholieke Universiteit Leuven, Belgium.

[9] Deen, S.M. 2003. *Agent Based Manufacturing - Advances in the Holonic Approach, Advanced Information Processing*. Berlin: Springer.

[10] Maturana, F. and D. Norrie. 1996. Multi-agent mediator architecture for Distributed Manufacturing. *Journal of Intelligent Manufacturing*, 7.

[11] Shen W., D. Xue and D. Norrie. 1997. An agent-based manufacturing enterprise infrastructure for distributed integrated intelligent manufacturing systems. In: *Proceedings of the International Conference on the Practical Application of Intelligent Agents and Multi-Agents*.

[12] Van Dyke Parunak, H., A. Baker and S. Clark. 1998. The AARIA agent architecture: from manufacturing requirements to agent-based system design. In: *Proceedings of Workshop on Agent-based Manufacturing ICAA*.

[13] Heikkila, T., M. Kollingbaum, P. Valckenaers and G.J. Bluemink. 1999. manAge: an agent architecture for manufacturing control. In: *Proceedings of the 2nd International Workshop on Intelligent Manufacturing Systems*.

[14] Toenshoff, H., I. Seilonen, G. Teunis and P. Leitão. 2000. A mediator-based approach for decentralised production planning, scheduling and monitoring. In: *Proceedings of International Seminar on Intelligent Computation in Manufacturing Engineering*.

[15] Klostermeyer, A. 2002. *Agentengestützte Navigation wandlungsfähiger Produktionssysteme*. Thesis (PhD). Otto-von-Guericke-Universität Magdeburg.

[16] Walsh, P., A. Vontas, P. Koutsakas, A. Koumpis, J. Martinetz and A. Klostermeyer. *Building enterprise-wide information supply chains based on the fractal company concept – lessons learnt, Challenges and Achievements E-Business and E-Work, Part 1.* Leipzig: IOS Press.

[17] Bratoukhine, A., T. Sauter, J. Peschke, A. Lueder, A. Klostermeyer. 2003. Distributed automation - PABADIS vs. HMS. *IEEE International Conference on Industrial Informatics INDIN'03*.

[18] Alexander, C. 1979. *The Timeless Way of Building.* New York: Oxford University Press.

[19] Gemma, E., R. Helm, R. Johnson and J. Villisides. 1995. *Design Patterns, Elements of Reusable Object Oriented Software*. Reading: Addison-Wesley.

[20] Sanz, R. and J. Zalewski. 2003. Pattern-based control systems engineering – using design patterns to document, tranfer, and exploit design knowledge. *IEEE Control Systems Magazine,* June: 42-60.

[21] Perronne, J.-M., L. Thiry and B. Thirion. 2006. Architectural concepts and design patterns for behavior modeling and integration. *Mathematics and Computers in Simulation*, 70: 314-329.

[22] Thramboulidis, K. 2008. Challenges in the development of mechatronic systems – the mechatronic component. In: *Proceedings of the 13th IEEE International Conference on Emerging Technologies and Factory Automation,* September, Hamburg.

[23] Lüder, A., J. Peschke, A. Klostermeyer and H. Kühnle. 2004. Design pattern for distributed agent based factory automation. In: *IMS International Forum 2004 – Global Challenges in Manufacturing.* Cernobbio. Proceedings: 783-791.

[24] Lüder, A. 2006. *Strukturen zur verteilten Steuerung von Produktionssystemen.* Thesis (Habilitation). Otto-von-Guericke-University Magdeburg.

[25] Bratukhin, A., B.A. Khan and A. Treytl. 2007. Resource-oriented scheduling in the distributed production. In: *Proceedings of the 5th International Conference on Industrial Informatics*, pp 1091-1096.

[26] Dzung, D., M. Naedele, T.P. von Hoff and M. Crevatin. 2005. Security for industrial communication systems. In: *Proceedings of the IEEE*, 93: 1152–1177.

[27] Schwaiger, C. and T. Sauter. 2002. Security strategies for field area networks. In: *Proceedings of the 28th Annual Conference of the IEEE Industrial Electronics Society*. November Sevilla. pp 2915-2920.

[28] Bellifemine, F., G. Caire and D. Greenwood. 2007. *Developing Multi-Agent Systems with JADE*. Chichester: Wiley.

[29] FIPA consortium: FIPA web page. Available from: www.fipa.org. [Accessed: 19.03.2008].

[30] Heinze, M., A. Lüder and W. Gantner. 2008. Structure and Functionality of a PABADIS'PROMISE Agent System. In: *Proceedings of the 14th International Conference on Concurrent Enterprising*, June, Lisbon.

[31] Wuensch, D. and A. Bratukhin. 2007. Multilevel order decomposition in distributed production. In: *Proceedings of the 12th IEEE International Conference on Emerging Technologies and Factory Automation*, pp 872-879.

[32] Rabelo, R. and L. M. Camarinha-Matos. 1994. Negotiation in Multi-agent based dynamic scheduling. *International Journal on Robotics and Computer Integrated Manufacturing*, 2 (4): 303-310.

7 Utilization of Advanced Control Devices and Highly Autonomous Systems for the Provision of Distributed Automation Systems

Athanasios Kalogeras[1], Luca Ferrarini[2], Arndt Lueder[3], Christos Alexakos[4], Carlo Veber[5], and Michael Heinze[6]

[1] Industrial Systems Institute, PSP Building, 26504 Platani Patras, Greece,
e-mail: kalogeras@isi.gr

[2] Politecnico di Milano, Dipartimento di Elettronica e Informazione, 20133 Milano, Italy,
e-mail: ferrarin@elet.polimi.it

[3] Otto-von-Guericke University of Magdeburg, Institute for Ergonomics, Manufacturing Systems, and Automation, Universitätsplatz 2, 39106 Magdeburg, Germany,
e-mail: arndt.lueder@ovgu.de

[4] Industrial Systems Institute, PSP Building, 26504 Platani Patras, Greece,
e-mail: alexakos@isi.gr

[5] Politecnico di Milano, Dipartimento di Elettronica e Informazione, 20133 Milano, Italy,
e-mail: veber@elet.polimi.it

[6] Otto-von-Guericke University of Magdeburg, Institute for Ergonomics, Manufacturing Systems, and Automation, Universitätsplatz 2, 39106 Magdeburg, Germany,
e-mail: michael.heinze@ovgu.de

Abstract The importance of European manufacturing remains high for the European economy as it still accounts directly for 22% of EU Gross Domestic Product (GDP), while it is estimated that 75% of EU GDP and 70% of jobs indirectly depend on the manufacturing sector. Facing intense global competition, the European manufacturing sector has to increase its flexibility and promote advanced business models involving the customer in all phases of the product lifecycle, such as mass customization. Distributed Automation Systems enable the enforcement of such models. Formerly physically centralized hardware and software is distributed in smaller units within the automation system. Advanced control devices relying on a common modeling paradigm may be used in order to provide intelligence at the field/control level. On the other hand, autonomous acting systems may provide the needed middleware for enabling flexible manufacturing systems.

7.1 Introduction

European industry must focus on the weaknesses of the current manufacturing system structures and behavior in order to address them in an efficient way. The outcome of these considerations is the conclusion that successful manufacturing systems require an increasing amount of flexibility related to manufacturing capabilities, manufacturable products, and usable resources. Such flexibility will make it possible for manufacturer to better respond to increasing market demands and address new trends and requirements like mass customization or changes to customer orders that have already been scheduled for production (late order freeze). In this context the end customer is integrated in the overall manufacturing process throughout its lifecycle. As a consequence, production control systems have to be more flexible (quickly reconfigurable) and to integrate seamlessly with Enterprise Resource Planning (ERP) systems.

Future manufacturing will require high flexibility, adaptability, and speed with respect to the organization and control of production and of supply-chain management. These requirements especially concern the control and networking of embedded control systems of manufacturing enterprises at the ERP (office), Manufacturing Execution System (MES) (factory control), and field control level.

With reference to the field control level, the overall trend, which can be observed for any special application, is the increasing amount of Information Technology (IT) applied, leading to an increasing share of the value of IT in any kind of product related to the industrial automation sector. This increase in IT within single devices fosters the trend toward distributed automation, as it allows the single device to carry more intelligence [1, 2]. The consequence of a fully distributed automation system is a system consisting of only nonhierarchical modules that are linked together by a communication system. The application software that controls this system has to be distributed among the modules of the system as well. The single modules that make up this system will be small mechatronic modules such as drives, pneumatic actuators, sensors, etc. that will be delivered by the manufacturer, already equipped with a certain amount of verified software providing the basic functions of the device.

The provision of this basic functionality by the supplier will allow for the reuse of the control code. Besides fostering the trend toward distributed systems, the increase in IT within single devices also fosters the need for code reuse. With an increasing amount of IT, the amount of software also increases. As industrial control has a high emphasis on reliability, the only way to meet this need for more and more software (at a competitive price) is the reuse of existing and verified software pieces. However, this remains a great challenge since, in fully distributed applications, the reusability of large portions of control application functions and the independence of instrumentation vendors are far removed from the current industrial practice.

With reference to the MES level there is a major trend toward decentralization of its functionalities through the application of middleware solutions characterized by a high degree of autonomy and based on a multiagent system (MAS) architecture [3, 4]. The utilization of different agent-based paradigms in the industrial environment may lead to an increased level of autonomy and promote a more flexible manufacturing environment.

On top and with reference to the ERP level [5], services are provided to the customer relevant to, e.g., order release or order change on demand, an extremely dynamic concept addressing the needs of many manufacturing industries. Service-Oriented Architecture (SOA) may be utilized in order to provide appropriate interfaces between the different layers.

The PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises (PABADIS’PROMISE) project [6, 7] has developed novel concepts for the realization of manufacturing systems increasing manufacturing system flexibility and robustness, ultimately influencing all layers of the automation pyramid. The main outputs of the project are a novel field level architecture, a MAS architecture supporting increased autonomy at the MES level, and a next generation ERP system utilizing SOA interfaces to MES allowing the information flow in both directions to provide continuous supervision of and interaction with running processes. A PABADIS’PROMISE Meta Model and an associated PABADIS’PROMISE Ontology [8, 9] provide the necessary semantics for such a system.

7.2 Methodology Issues

The PABADIS’PROMISE project addresses a highly innovative control system design methodology, model systems facing the needs of this design methodology, and a most generic architecture as a starting point of the design methodology [10]. This design methodology aims at providing a means to gradually enrich and extend the most generic control system architecture provided by the project by application-case-related information and behavior in a first phase and implementation details in a second phase, finally resulting in a control system implementation facing all needs of the application case and following the PABADIS'PROMISE control paradigm.

Therefore, a three-level approach [11] for the design of a distributed control system is defined. This approach is based on a set of models relevant for the different levels and covering different abstractions of the overall system behavior. During the system design, the different models will be successively improved and translated/transformed to each other. Special relationships among the models, resulting from the different viewpoints they cover, allow maintaining coherence and consistency among model entities of the overall system.

This improvement/translation/transformation process will constitute a PABADIS'PROMISE adaptation process. This process will

- Start with the high-level architecture consisting of a PABADIS'PROMISE Role Model and a PABADIS'PROMISE Meta Model;
- Use the application cases as a delineation document;
- Integrate application-case-dependent general behavior and information-related details into a PABADIS'PROMISE Role Model and PABADIS'PROMISE Meta Model;
- Generate a PABADIS'PROMISE Agent Model and PABADIS'PROMISE Ontology as a low-level architecture specification in the sense of a functional specification; and
- Generate ultimately the PABADIS'PROMISE agent implementation, PABADIS'PROMISE data repositories, and PABADIS'PROMISE control system implementation.

The adaptation process has to ensure that the low-level architecture is generated in line with the high level architecture and to verify the coherence and the consistency of the resulting conceptual models of architecture and of ontology.

7.2.1 PABADIS'PROMISE High-Level Architecture

The PABADIS'PROMISE high-level architecture describes the general structure of the software and the relations and behavior of the components of a PABADIS'PROMISE system from a conceptual point of view and, therefore, covers only very abstract behavior descriptions and data structures. It consists of a most generic structure describing the decision process within a control system named PABADIS'PROMISE Role Model and a generic data representation covering all required data within the decision process named PABADIS'PROMISE Meta Model.

The PABADIS'PROMISE Role Model defines responsibilities, activities, and interactions without assigning them to dedicated implementation entities named roles. This makes it possible to adapt the system to different implementation approaches without changing the overall functionality. The PABADIS'PROMISE Role Model starts from the GAIA methodological approach [12] and adopts it for PABADIS'PROMISE purposes. The PABADIS'PROMISE Role Model will be realized through a MAS-based MES, realized on JADE [13], a Foundation for Intelligent Physical Agents (FIPA) [14] compliant framework completely implemented in Java^{TM}.

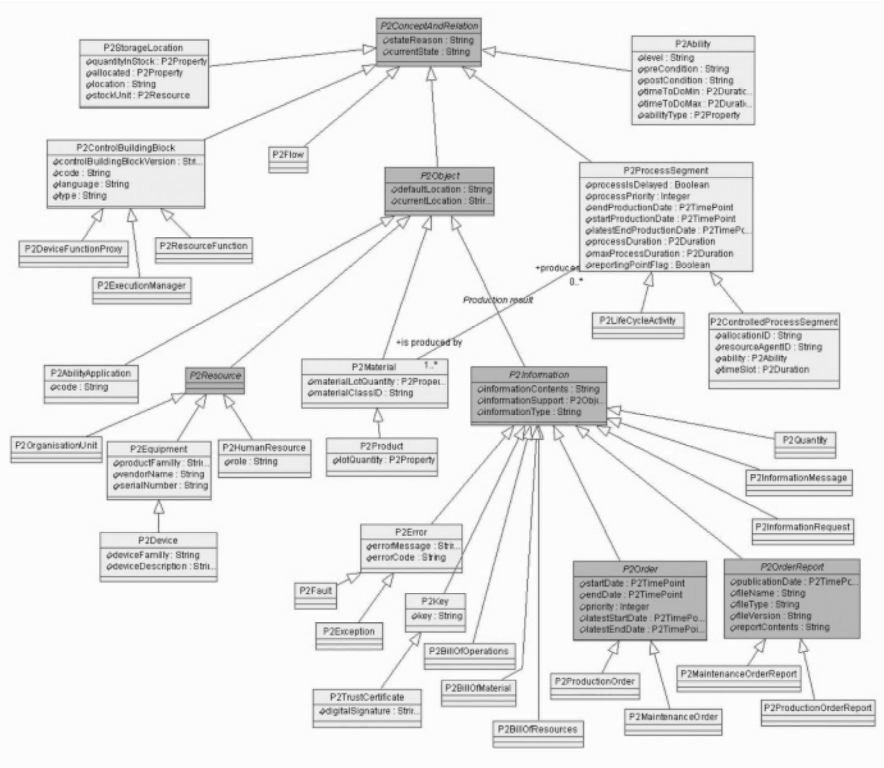


Fig. 7.1 PABADIS'PROMISE Meta Model

The second element of the PABADIS'PROMISE high-level architecture is the PABADIS'PROMISE Meta Model that is used to define, validate, and formalize the common meaning of a set of basic concepts, relations between concepts, and concept attributes required to describe a production system in the PABADIS'PROMISE project context to enable a consistent data representation within the PABADIS'PROMISE control system. The hierarchy of the PABADIS'PROMISE Meta Model is presented in Fig. 7.1.

7.2.2 PABADIS'PROMISE Low-Level Architecture

The PABADIS'PROMISE Role Model and the PABADIS'PROMISE Meta Model introduced by the PABADIS'PROMISE high-level system architecture concentrate on required functionalities and responsibilities. This view enables an implementation-independent description, which allows adapting the instantiation of the model to different characteristics and needs of the production system to control. When it comes to the actual implementation of the roles and meta-model entities, there is a need for their instantiation into concrete entities.

The PABADIS'PROMISE MAS [15] is composed of the following different types of agents, derived from the PABADIS'PROMISE Role Model (Fig. 7.2).

- Resource Agents (RAs) manage the manufacturing resources including scheduling and control. They provide manufacturing capabilities to other resource agents and order agents called abilities.
- Order Agents (OAs) are responsible for the execution of manufacturing orders including order scheduling and manufacturing process execution control.
- Order Agent Supervisor (OASs) are needed for the management of the OA lifecycle and supervision of order execution.
- The Resource Agent Supervisor (RAS) is responsible for the RA lifecycle management and supervision of this agent type.
- An Ability Broker (AB) is used for the collection of ability descriptions of RAs and for the provision of a yellow page service to request information about these abilities and its providers for OAs and RAs.
- A Product Data Repository (PDR) is a database for product information including manufacturing process descriptions in terms of required abilities and order-related control application building blocks called ability applications especially for OAs and RAs.
- An Information Collector (IC) is a generic entity for the collection of system information for external system entities.

On the other hand, a PABADIS'PROMISE Ontology presents a manufacturing ontology allowing future PABADIS'PROMISE components and applications to become fully interoperable with each other throughout the manufacturing process lifecycle. The PABADIS'PROMISE Ontology provides formal and unambiguous definitions of all the components and of their interactions with each other in an enterprise/industrial environment in order to establish a common "language" for exchanging and describing all the complex information that is related to the lower levels of an industry. The PABADIS'PROMISE Ontology aims to formalize conceptual information about:

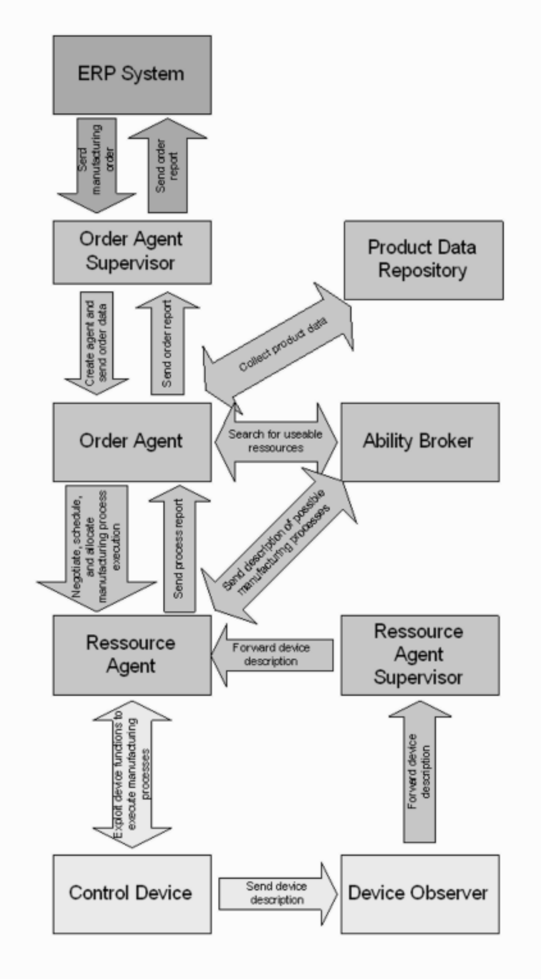

ERP System
Send manufacturing order
Send order report
Order Agent Supervisor
Product Data Repository
Create agent and send order data
Send order report
Collect product data
Order Agent
Search for useable ressources
Ability Broker
Negotiate, schedule, and allocate manufacturing process execution
Send process report
Send description of possible manufacturing processes
Ressource Agent
Forward device description
Ressource Agent Supervisor
Exploit device functions to execute manufacturing processes
Forward device description
Control Device
Send device description
Device Observer

Fig. 7.2 PABADIS'PROMISE system entities

- Each resource that can be used in a production line: machines, equipment, control systems, actuators, personnel, materials, etc.
- Each product that can be produced (i.e., transformed via a process) in this production line
- Each operation through the definition of each process (defined as a set of sequential or parallel operations): to drill, to move, to transport, to maintain, to measure, etc.

The PABADIS’PROMISE Ontology is derived from the PABADIS’PROMISE Meta Model.

7.3 Resource Agent Architecture

The RA is the agent responsible for the management of the lifecycle of resources. A detailed specification is given in [16]. It plays a key role in the PABADIS'PROMISE system to represent and access production resources. Using the managed devices this agent provides abilities to the MES layer. Therefore, the RA executes IEC 61499 [17] applications using functionality provided by field devices. These applications implement the abilities. The relation between abilities and applications is defined in the PDR and can be requested by the RA. The AB provides the services to register the abilities and for conveying them to the manufacturing system.

The RA is managed and supervised by the RAS, which initiates it when a device it is responsible for appears. The basic structure of the RA is shown in Fig. 7.3. To fulfil its tasks an RA realizes five main functionalities:

- *Device function access* is realized using a device-specific Device Proxy (DP). It enables communication and data exchange to the control device based on the communication protocol and technology required for a dedicated device. The DP is the access point for the integration of device functions in an Ability Application.
- *Device monitoring* enables the RA to track the current status of an RA, detect failures, and calculate the availability of device functions for ability provision.
- *Ability handling* provides the given ability to the MES layer by registering an ability in the AB. It handles the relation between abilities and the control applications implementing these abilities (called ability applications) using the set of device functions provided by the devices managed by the RA. For both cases information is requested from the PDR.
- *Ability application control* is the main functionality of the RA and carried out by executing an Ability Application in an appropriate runtime environment. It coordinates the execution of the required device functions in order to realize the ability provided by the RA.
- *Scheduling* is needed to coordinate the timing of the usage of abilities by OAs. Each RA maintains its own schedule, taking dependencies between abilities into account if multiple abilities are provided by the RA.

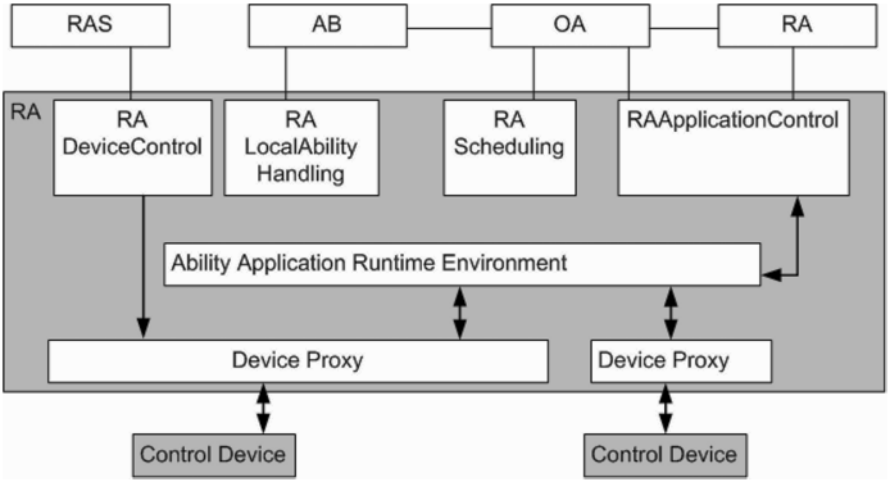


Fig. 7.3 Resource agent structure

7.3.1 Ability Application

As described above, the execution of certain abilities on request from other system entities is one of the main tasks of an RA. These system entities can be either other RAs or OAs managing orders requiring the given abilities. In order to control the execution of an ability, the RA executes an application that realizes a coordinated invocation of all required device functions executed on field level devices. These applications are IEC61499-compliant function block (FB) networks, which the RA can execute in its internal Ability Application Runtime. As this runtime is only responsible for the triggering of functionality in field level devices, it is normally not executed under real-time constraints. As IEC 61499 application the Ability Application consist of FBs, specifically, it uses three different types of FBs:

- Device Function Proxies (DFPs) implemented as Service Interface Function Blocks (SIFB), which enable the access to a DP and represent the underlying device functions inside an Ability Application.
- Resource FBs (Basic FBs) representing Ability-related parts of the Ability application. They define how the Ability Application invokes the device functions.
- Execution Manager (EM) FBs are used to enable the access for RA functionality to the application in order to pass parameters or report execution states and results.

Every ability can be realized by several applications that use DFPs accessing DFs provided by devices. They will be created by a control engineer, who is familiar with the structure and the responsibilities of the manufacturing plant. A more detailed description of the design process for the applications is given in [9].

The data belonging to the application realization are distributed among the PDR and the RAs. Thus the PDR contains the data about all applications implementing a particular ability, whereas an RA will always handle only the applications it is able to execute. Therefore it contains a runtime to execute ability applications. This runtime provides functionality to execute IEC 61499 applications.

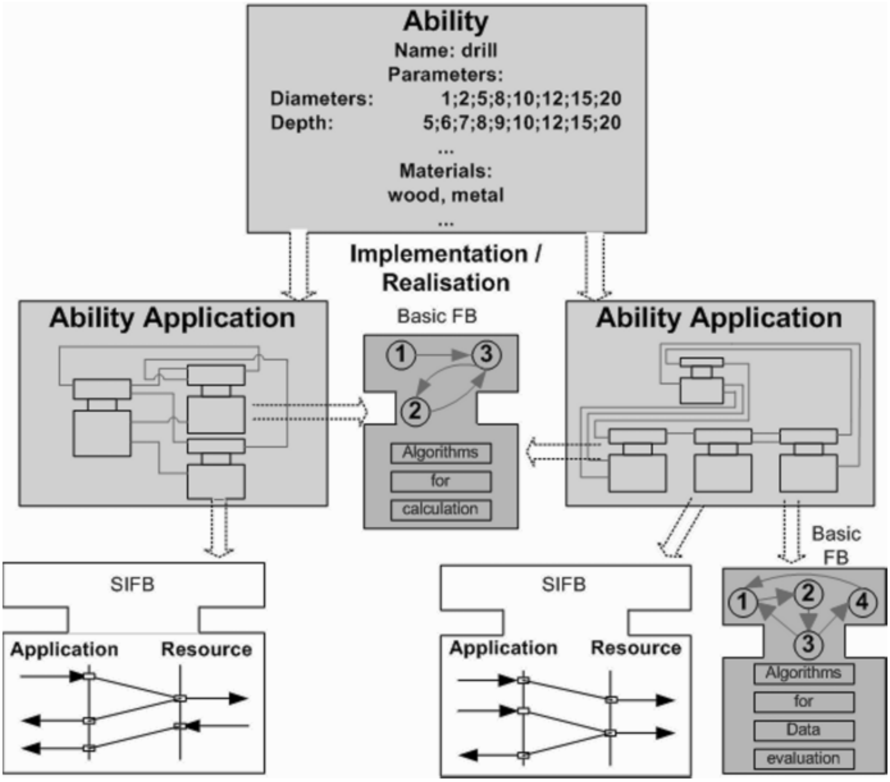


Fig. 7.4 Relation between ability and ability application

The descriptions of these applications are composed of

- An FB network description contained in an IEC61499-compliant resource describing an XML document. Only one network description is allowed in such an XML document;
- A set of XML documents describing Basic FBs. These Basic FBs are called Resource FBs in the PABADIS'PROMISE context. They can execute a set of algorithms implemented as Java classes. These classes either have to be part of

the descriptions or will be referenced by them and be loaded during FB creation;

- A set of DFPs, which will be instantiated during device integration. These DFPs will be linked to an application during instantiation;
- One EM used to enable the access for RA functionality to the application. All EMs have the same event connection structure. The data connections are application dependent and used to feed parameters into and from the application.

The dependence between ability and ability application and the use of FB types within them is depicted in Fig. 7.4.

7.3.2 Device Proxy

In order to control a resource, the RA needs to access the related control devices. This access can be either established via a network connection using, e.g., standard Ethernet or by the RA running directly on the device. In both cases the RA uses a DP establishing the connection to the counterpart in the control devices called the Device Manager (DM, see following section). The latter always runs directly on the device. To connect the DP with the Ability Application, the RA creates DFPs, one for each Device Function (DF) the device is able to provide.

As the communication technology used by the DP is device specific, it can be used to access a variety of device types. This ranges form device internal calls in case of an RA running on the a control device, over Ethernet-based protocols, to conventional field busses in the case of ,e.g., ordinary Programmable Logic Controllers (PLCs).

Besides providing access to DFs the DP allows the RA to constantly monitor the device capabilities provided by DFs. If a DF breaks down or needs maintenance, the RA has to recognize it and the DFP needs to be deleted. The availability of DFs influences the capability of RAs to provide abilities to the system because of their usage in the ability-implementing applications. So the RA needs to monitor the devices constantly to check the current state of the DFs. If the state of a DF changes, the RA has to update its own capabilities to provide abilities in interaction with the PDR. Afterwards, it has to inform the AB about its changed state. If the changed state influences already allocated processes, the RA has to inform the system entities, e.g., by canceling the allocation. The same has to happen if the whole device disappear from the system.

7.4 Field Control Architecture

In order to provide access to the field level, the RA has to be able to interact with a huge variety of control equipment at the field level. The flexibility in terms of used communication technologies is provided by the DP. Nevertheless a control device has to fulfil some basic requirements and provide a set of functions to enable integration in a PABADIS’PROMISE system. This basically concerns the provision of accessible device functions and support of the device integration process. Such a device is called a PABADIS’PROMISE Control Device (PABADIS’PROMISE-CD) (Fig. 7.5).

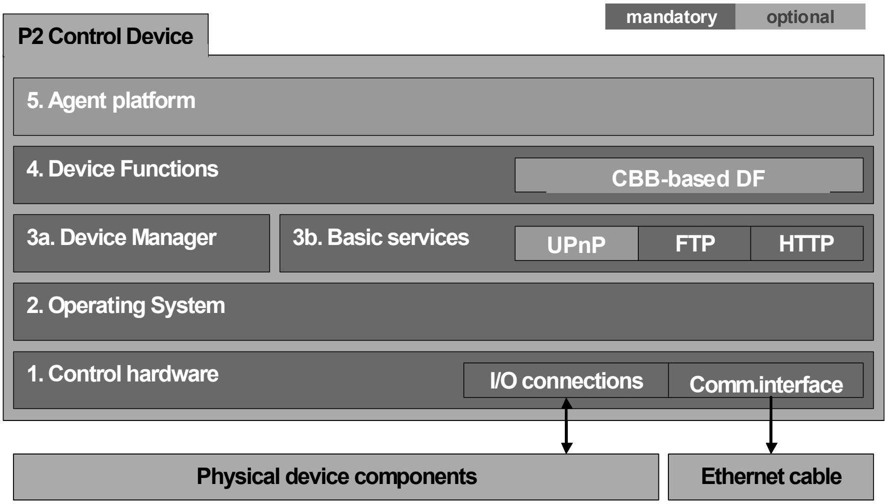


Fig. 7.5 Structure of PABADIS'PROMISE control device

7.4.1 PABADIS’PROMISE Control Device

A PABADIS’PROMISE-CD is an independent physical entity able to control one or more production process operations in a PABADIS’PROMISE system. This means a PABADIS’PROMISE-CD is an entity that may be composed by mechanical, electrical, and computational parts whose behavior cannot be influenced by external entities except from those interactions allowed by the interface. Figure 7.5 shows the basic structure of a PABADIS’PROMISE-CD. The core functionalities are the device functions provided to the RA and the DM controlling the behavior of the PABADIS’PROMISE-CD. The DM acts as an access point to the PABADIS’PROMISE-CD for the RA and provides the following features:

- Handles the PABADIS'PROMISE-CD startup;
- Starts/stops of Device Functions;
- Provides information during runtime (e.g. DFs availability); and
- Manages exceptions.

The basic services are software objects supporting the functionalities of the DM, e.g., a PABADIS'PROMISE-CD can provide an UPnP mechanism to support the automatic device integration process.

One or more RAs can be executed directly on a PABADIS'PROMISE-CD if it is able to provide all the necessary features required for an agent platform. This mainly concerns a JVM, a set of APIs, and the Jade platform.

Finally, the PABADIS'PROMISE-CD must also provide a description document, called PABADIS'PROMISE-CD Description, containing its characteristics in terms of computational capabilities, communication protocols supported, and DFs provided.

The PABADIS'PROMISE-CD builds on the device model elaborated in the framework of the Total Lifecycle Web-integrated Control (TORERO) project [18] and appropriately enriched in the framework of the PABADIS'PROMISE project and incorporated in the overall PABADIS'PROMISE Meta Model.

7.4.2 Device Observer (DO)

The DO is a system entity that is executed in the MAS, but it is conceptually located at the field level. It is responsible for the integration of new devices into the system. Therefore, it provides automatic hardware detection mechanisms like, for example, BootP or UPnP. To use this functionality the device has to support this kind of functionality. Furthermore it is possible to integrate a device using an HMI. In that case the integration will be done like a driver installation providing the data via a certain medium, e.g., a CD.

The DO reads and processes the device-related information. It will be converted into a system understandable format to allow the seamless integration into the PABADIS'PROMISE environment. The data will be sent to the RAS, which then assigns the device to the responsible RA.

7.5 Control Device Integration Process

One important process inside a PABADIS'PROMISE system is the integration of new devices. It involves several system entities and influences system entities in all layers of the automation pyramid. The process starts at the field level with the DO. This agent provides several behaviors allowing the recognition of appearance of new devices. Therefore, automatic hardware detection mechanisms like BootP or UPnP are provided. If a new device appears, a first connection will be established by one of these mechanisms. An XML document describing the capabilities of the device will be read. Afterwards, the DO fills the data in system understandable language and sends it to the RAS. The RAS receives the message and searches the content for data describing the RA responsible for managing the device.

Then, the RAS has to search its internal repository containing all RAs running in the MAS for the responsible one. If it is not yet present, it will be started and the description will be provided.

If the RA receives the message, it has to establish a connection with the DM. Depending on the location of the RA, the access will be realized via a network connection or directly. In both cases the RA has to instantiate the DP. During its initialization the DP establishes the connection to the DM. If the DP is running, the DFPs can be instantiated using the IEC61499-compliant XMLs contained in the device description. Therefore it is necessary to load and instantiate classes implementing the access to the DFs via DP. These classes enable calling the function and providing input parameters. After finalization of the call, the particular function returns with a set of output parameters. If all the DFPs are instantiated and able to work or at least the instantiation failed, the RAS will be informed about the RA state. In its repository it contains a list of all RAs and all DFPs the particular RA is able to use. Furthermore, the RA sends a message to the PDR requesting all realizable abilities. The PDR contains the information about applications realized as FB network descriptions as well as Resource FB descriptions and Service Interface FBs realizing the DFPs used in these networks. A link to abilities realized by this application closes the cycle. The PDR responds with all abilities realized with available networks using available Resource FBs and DFPs sent by the RA.

After receiving the PDR response, the RA checks for the capability to provide new abilities. They have to be registered at the AB with the RA as provider. Moreover, the ERP has to be informed if new abilities appear in the system. Figure 7.6 and the enumeration below recapitulate the described process and the involved entities.

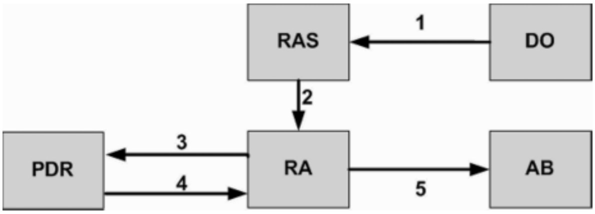


Fig. 7.6 Device integration, involved entities

1. DO sends device description after discovering a new device;
2. RAS sends the device description to the responsible RA after starting or at least searching for it;

3. RA requests list of realisable abilities after instantiating DP and all DFPs for provided DFs;
4. PDR sends list of abilities with references to implementing applications and composed DFPs, which are available at the RA;
5. RA registers the new abilities at AB.

7.6 Conclusions

The chapter presents innovative concepts for flexible manufacturing systems addressed in the framework of the PABADIS'PROMISE project. It makes it possible to address in an efficient way modern needs in the industrial environment leading to adaptive production of highly customizable products that is robust enough to anticipate unexpected events generated both by the end customer (change of order) or possible failures of the participating resources.

The chapter focuses on the representation and access of production resources in a PABADIS'PROMISE system. The resource management in PABADIS'PROMISE enables a synchronization of executable device functions at the field control level and appropriate ability representation on the MES layer. Furthermore it defines concepts and interfaces to access field devices from the MAS. This is done in a very flexible way in terms of technologies and implementation and provides the following advantages:

- Autonomous acting agents for resource control allowing a self-organizing production with high efficiency, high flexibility to react to changing demands, and small system downtimes;
- On-demand design of control code using innovative concepts for FB-based applications like IEC 61499;
- Mechanisms for automatic hardware integration like UPnP to allow fast reactions to changing demands and reduce maintenance time;
- Modular structure of manufacturing systems for on-demand combination of needed capabilities; and
- Scheduling mechanisms between autonomous system entities to allow the most efficient use of production system capabilities.

The PABADIS'PROMISE project opens new horizons for the use of innovative IT-like agent systems and intelligent devices/components. The application of such concepts to manufacturing systems will provide industry with ways to react to future challenges resulting from worldwide competition.

Acknowledgments This chapter was supported by the PABADIS'PROMISE project (ref. nr. FP6-IST-016649) funded by the European Commission.

References

[1] ARC Advisory Group. 2001. PLCs*: Is There a Future?.* ARC Strategies, Dedham: ARC.

[2] Lorentz, K., A. Lüder and J. Peschke. 2001. PABADIS – Fabrikautomatisierung basierend auf verteilten Systemen. *In:* Schraft, R., K. Bender and G. Brandenburg, eds. *Tagungsband SPS/IPC/Drives 2001.* Heidelberg: Hüthig.

[3] Shen, W., Q. Hao, H. Yoon and D.H. Norrie. 2006. Applications of Agent Systems in Intelligent Manufacturing: An Update Review. *International Journal of Advanced Engineering Informatics,* 20 (4): 415-431.

[4] Lüder, A., J. Peschke, A. Klostermeyer and H. Kühnle. 2004. Design Pattern for Distributed Agent Based Factory Automation. In: *Proceedings of the IMS International Forum 2004 - Global Challenges in Manufacturing.* Cernobbio. pp. 783-791.

[5] Rode, J. and D. Wünsch. 2007. A research agenda for adaptive manufacturing. In: *Proceedings of the 12th IEEE International Conference on Emerging Technologies and Factory Automation*, 25-28 September, Patras.

[6] PABADIS'PROMISE Consortia. 2008. *White paper 1 – Structure and behaviour of a PABADIS'PROMISE system.* Available from: www.pabadis-promise.org [Accessed 19.03.2008].

[7] Ferrarini, L., C. Veber, A. Lüder, J. Peschke, A. Kalogeras, J. Gialelis, J. Rode, D. Wünsch and V. Chapurlat. 2006. Control Architecture for Reconfigurable Manufacturing Systems: the PABADIS'PROMISE approach. In: *Proceedings of the 11th IEEE International Conference on Emerging Technologies and Factory Automation,* September, Prague.

[8] PABADIS'PROMISE Consortia. *Deliverable 3.1, Development of Manufacturing Ontology.*

[9] Kalogeras, A.P., L. Ferrarini, A. Lueder, J. Gialelis, C. Alexakos, J. Peschke and C. Veber. 2007. Ontology-driven Control Application Design Methodology. In: *Proceedings of the 12th IEEE Conference on Emerging Technologies and Factory Automation,* 25-28 September, Patras. pp. 1425-1428.

[10] PABADIS'PROMISE Consortia. *Deliverable 2.1, Concept of Overall PABADIS PROMISE Architecture Components.*

[11] Lüder, A., J. Peschke, A. Bratukhin, A. Treytl, A. Kalogeras and J. Gialelis. 2006. Order Oriented Manufacturing Control, The PABADIS'PROMISE approach. In: *ANIPLA 2006 International Congress Methodologies for Emerging Technologies in Automation*, November, Rome.

[12] Wooldridge, M., N. Jennings and D. Kinny. 2000. The GAIA methodology for agent-oriented analysis and design. *Autonomous Agents and Multi-Agent Systems,* 3 (3): 285-312.

[13] Bellifemine, F., G. Caire and D. Greenwood. 2007. *Developing multi-agent Systems with JADE.* Chichester: Wiley.

[14] FIPA consortium. 2008. Available from: www.fipa.org [Accessed 19.03.2008].

[15] PABADIS'PROMISE Consortia. *Deliverable 4.1.* Available from: www.pabadis-promise.org, [Accessed 19.03.2008].

[16] PABADIS'PROMISE Consortia. *Deliverable 6.1.* Available from: http://www.uni-magdeburg.de/iaf/cvs/pabadispromise/dokumente/del_6_1_final.pdf [Accessed 19.03.2008].

[17] Vyatkin, V. 2007. *IEC 61499 Function Blocks for Embedded and Distributed Control Systems.* O3NEIDA – Instrumentation Society of America.

[18] Lorentz, K., A. Kalogeras, T. Bauten, L. Ferrarini, C. Schwab, J. Thieme, G. Fogliazza and A. Vontas. Next Generation Integrated Development of Automation Control Code in TORERO. In: *International Symposium on Industrial Electronics IEEE-ISIE'2003*, June, Rio de Janeiro.

8 Design Patterns for Distributed Control Applications

Arndt Lüder[1], Jörn Peschke[2] and Ricardo Sanz[3]

[1] Otto-von-Guericke University of Magdeburg, Institute for Ergonomics, Manufacturing Systems, and Automation, Universitätsplatz 2, 39106 Magdeburg, Germany
e-mail: arndt.lueder@ovgu.de

[2] Otto-von-Guericke University of Magdeburg, Institute for Ergonomics, Manufacturing Systems, and Automation, Universitätsplatz 2, 39106 Magdeburg, Germany
e-mail: joern.peschke@ovgu.de

[3] Dept. Business Administration. School of Industrial Engineering, Universidad Politécnica de Madrid., C/José Gutiérrez Abascal, 2. 28006 Madrid, Spain
e-mail: ricardo.sanz@etsii.upm.es

Abstract Design patterns are an appropriate means to code solution knowledge within different areas of science and practice. They cover a description of the problem with the problem context, a description of a possible solution to the problem, and ancillary conditions of this solution. Initially design patterns were invented in the building architecture sciences but were quickly applied to information science and other disciplines.

This paper deals with the possible application of design patterns within industrial control. It describes how design patterns can be used by the example of design patterns for industrial field control systems. Therefore, initially the paper presents requirements for field control systems, maps the design pattern approach to them, and describes three basic design patterns for distributed field control systems.

8.1 Introduction

Control applications have been applied since a few tens of centuries. From simple mechanical controllers in water systems and clocks to mechanical controllers for steam engines or completely hard-wired relay control systems up to the current programmable logic controllers, different technologies for control systems have been developed and applied. In many cases these systems have been based on recurring structures that can be observed as fundamental design patterns for control systems. These design patterns can be used as design guidance in the case of the design of a new control architecture or at least in the case of the design of a new control application.

Design patterns are a currently used widely design aid emerging from building architecture but made more widespread by information sciences. The term design pattern first appeared in the initial work of Alexander in 1979 [1]. Alexander worked as an architect for building design and construction. He recognized that building design always follows the same basic rules but was adapted to the real projects following cultural and geographic implementation specialties. Hence, in his basic work he describes some basic design patterns for building design.

With the emergence of the programming paradigm of object orientation and the software engineering processes and technologies based on it, the idea of design patterns as application-independent basic design principles has been adopted to support engineering process efficiency and quality. The main research activity initiating this trend was the work of Gamma et al. [2].

The main intention behind the application of design patterns in computer science was to provide means to describe and reuse known problem solutions with proven quality and usability and enabling the description of structures behind the original problem solution and dependencies among system components and behaviors. Since 1995 several design patterns have been developed addressing general and special problems [3, 4].

Within the last few years the use of the design pattern concept has been integrated into several scientific disciplines, including the design of control applications. Initially, it was integrated within the design processes of control application software. But it was quickly extended to the engineering of complete control systems including control software, control hardware, and the controlled plant itself. In this field valuable progress has been made. Overviews on the attained results are given in [5, 6, 7].

The application of design patterns was supported by the increasing trend toward the application of mechatronical units within manufacturing system design. Mechatronical units are seen as manufacturing system components comprised of hardware, electrical systems, and electronics including control software designed with the aim of providing special functionality to the overall system [8]. With respect to the application of design patterns within control system engineering, mechatronical units enable the combination of controlled processes with its control device and its controlling software structures. In this way, completely new engineering processes are enabled exploiting libraries of mechatronical units and, hence, libraries of control application building blocks [9].

The description and application of design patterns in control design is based on the principle of describing a solution to a problem within a special context. Therefore, a design pattern includes the description of the basic problem, the context of the problem (i.e., the field where the problem has occurred), drivers important for the solution, the solution of the problem itself, and the background of the origin, and, finally, the possible application fields of the solution. The descriptions within this paper will follow this structure. The particular structure used to document the design pattern – as that suggested previously – constitutes what is referred-to as a pattern schema. Many pattern schemata have been proposed so far for many concrete domains including control systems [10, 11].

8.2 Requirements for Field Control Systems

The design of field control systems is driven by a set of requirements emerging from technology and economics. Generally, in the control architecture the Field Control Level aims to control individual manufacturing processes, i.e., to control the physical execution of manufacturing processes and, therefore, to drive them via actuators and measure them via sensors. Hence, the core individual entities of this level to be considered are the manufacturing process, sensors, actuators, and control devices. Following the idea of mechatronical units they are combined with control software running within control devices, sensors, and actuators realizing the required manufacturing process functionality.

All in all, the basic entity structure of field control systems is oriented towards the most economic execution of manufacturing processes against the background of a turbulent surrounding of product variants, manufacturing volume, and exploited technologies. Hence, the design of field level control systems forces special requirements on the design process, on the control system architecture, and on the integrated devices. The most recent are listed below.

1. *Low building and maintenance costs:* Building and maintenance costs influence significantly competitiveness and customer satisfaction and, hence, the profit margins and market share of the company using the manufacturing system. Moreover, productivity of manufacturing systems is always a ratio between the production obtained and the production costs where the production costs depend on the costs of the design and application (including maintenance) of the

manufacturing system. So in order to increase productivity, we have to improve performance but also reduce overall costs.

2. *Robustness with respect to operability under malfunctions:* Manufacturing systems need to operate in a predictable way under all circumstances. This includes also proper behavior in the case of malfunctions. Independently of its size and complexity, a manufacturing system has to be continuously available or at least prevent dangerous behavior in the case of faults. This requires overall reliability and maintainability, but also some degree of fault tolerance.
3. *Flexibility with product types, product volume, and equipment:* Flexibility in manufacturing is a very broad concept covering varying types of flexibility. With respect to manufacturing systems most recent types of flexibility are as follows:

   – Product type flexibility describing the ability to manufacture different kinds of products and different versions of similar products or the same product,
   – Product volume flexibility describing the ability to manufacture different lot sizes of products, and
   – Equipment flexibility describing the ability to add, delete, and change equipment within the manufacturing system without additional efforts during system runtime.

   Hence, flexibility is the capacity of a system to adapt to new manufacturing requirements as well as to new manufacturing devices and technologies.
4. *Application-dependent control system development:* The development of the control system should be guided by the application it is intended for, i.e., by the functions and functionalities it has to allow in the controlled system and, thereby, by the manufacturing process the control system has to control. The control engineer should be guided in his work by the desired behavior expected from the manufacturing system and, hence, the controlled system that has to be mapped to available mechatronical units with its inherent manufacturing-process-related capabilities and control code fragments.
5. *Component-based development approach:* Usually manufacturing processes are structured in a modular/hierarchical way consisting of manufacturing process steps, substeps, subsubsteps, etc. Each of the steps within this hierarchy is based on the execution of a physical process provided by a mechatronical unit. In addition, manufacturing systems are today large and complex real-time systems. They can benefit significantly from a component-based development approach where new systems are constructed by composing reusable, documented, and previously tested concurrent objects. The re-use of components is a very powerful instrument for reducing development time and related costs, but also for increasing the control system reliability.
6. *Human integration and friendliness:* In automation systems, human integration is more than just a user interface. Control systems usually require human inte-

gration for supervisory control. Hence, it covers aspects related to the active cooperation of human and semiautonomous systems to execute coordinated tasks required to successfully execute manufacturing processes. Here, problems of clear representation of current manufacturing system states and possible supervisory control activities have to be solved.

7. *Compliancy with existing system and standards:* Standardization of modules and integration protocols can actively support the long lifecycle of reconfigurable automation systems and can provide a means to reduce building costs. Compliancy with existing standards is also required by the need of an enterprise to integrate all its automation systems in a larger factory organization.
8. *Integration with existing control devices and legacy systems:* Inside an organization an automation system is not an island but a component of a larger system. Control applications must be easy to integrate in more complex systems, providing and receiving a flow of data toward the legacy enterprise management system. On the other end the control architecture must be able to easily interface existing control devices.

This set of requirements needs to be properly addressed within control system design, resulting in an overall increase in design and construction complexity. Thus, it is obvious that it is more necessary than ever to provide some basic design patterns to assist control system designers and end users to design, build, and operate a safe, efficient, and economical control system.

8.3 Rationales of Design Patterns

The basic problem of control is the necessity of controlling the physical values influencing the controlled system based on the current state of the controlled system. By economic, social, legal, and other reasons the controlled system has to follow a specified sequence of states over time. Which system structure of controlled system and controller enables the most efficient way of control following the specified state sequence? This problem arises in all fields of economy and society including process automation, building automation, and factory automation.

In this light, the complete history of control technology can be seen as the search for a general solution to the named problem and the development of concrete, practical solutions to more tangible and restricted problems. One of the best known general solutions is that of the closed-loop, feedback-based control system. In general, we can say that the usability of this solution for the general problem is not limited to a particular case; hence the feedback control design pattern can be applied – in principle – to any automation system.

As can be seen in this basic example, the main aim of a design pattern is the description of basic structural and behavioral principles of a problem solution that can be reused in a wide domain. Nevertheless, the description of the solution itself is not sufficient [5]. It has to be accompanied by

- A description of the problem the solution is targeting on,
- The surrounding and environmental conditions the design pattern is valid within, and
- The drivers and ancillary conditions that imposed this solution.

In addition to this directly pattern-related information, the integration of the design pattern in the overall "world of design pattern" must also be characterized, i.e., other design patterns that are related to the considered design pattern must be identified, as well as the type of relationship must be given.

If design patterns are described in this way, they can be easily used to catalog and document design knowledge. In this way, knowledge exchange among scientists and practitioners, system integrators and end users, and people of different manufacturing sciences can be improved by the common knowledge base and the common description framework that pattern languages provide. This communication problem is very relevant within the automation world. Here specialists from information sciences, mechanical engineering, process industry, electrical engineering, and other fields have to work together towards a common goal.

8.4 Existing Design Patterns for Field Control Systems

There have been several developments in the field of application of design patterns in the control domain. Obviously there are two main sources of patterns specifically focused on the two sides of the technological spectrum that goes from *control theory* to *computer science*.

On one side control patterns are the basic knowledge that all control engineers learn in any control textbook. There are plenty of examples from the feedback control pattern representing the commonly used closed-loop control cycle or the simple Proportional integral differential (PID) pattern describing the most used continuous controller structure exploiting proportional, integral, and differential control reactions on measured controlled system states, to the more complex model-based adaptive-predictive control or expert-fuzzy patterns. These control textbooks, however, lack an important aspect that is vital for the effective use of the pattern at large: the systematic method of the presentation that a pattern schema provides.

On the other side of the spectrum, pattern schemas are usually the very well investigated structures of design patterns in the computer science community. The books [3, 12, 13] offer valuable material for the implementation of computer-based control systems related to the field control layer. Computers and software are the building materials of control systems, and software design patterns are very valuable in the efficient implementation of them.

In this last case, however, most of the patterns lack some of the critical information and guidelines that are necessary to be applicable in the control field: those related to the real-time operation of the pattern. There are few examples of design patterns that fully address these issues, and in most cases they are focused on very specific domains; for example, [14] is focused in the avionics software domain and [15] is specifically focused on small-scale embedded systems on microcontrollers. In the newest approaches the aspect of design methodologies and engineering procedure has attracted greater interest. As examples [6] focuses on the methodological aspect of the integration of design patterns in control system engineering, while [7] is mainly targeted at the application of design patterns in the engineering process of control systems based on mechatronical units.

Nevertheless, it is necessary to extend the systematic application of a unified pattern schema across the entire disciplines of control, computing, and communications to be effective in the capture of the design knowledge that is necessary for the adequate implementation of the complex distributed controllers that are required in plantwide control of manufacturing plants and can follow the needs of the manufacturing systems sketched above. An example of such a schema is presented in [5].

8.5 Design Patterns for Distributed Field Control Systems

In the following section selected design pattern dedicated for Distributed Field Control Systems will be described. The main aim of these design patterns is the enforcement of the paradigm of distributed control systems based on distributed intelligence.

8.5.1 Design Patterns – Distributed Control Applications

Distributed control applications are based on the cooperative solution of a control problem by a set of independent but cooperating intelligent and non-intelligent control devices as usually given in the case of interacting mechatronical units within a manufacturing system. Within this context the problem of assignment of control application parts to control devices has to be solved.

In factory automation the aspect of distributed intelligence on different control devices is growing fast. Control applications and the intelligence inherent within these applications are being increasingly distributed among interacting and cooperating control devices. In parallel, this increases the control application execution capabilities of control devices resulting from the increase in computational capabilities.

This distribution problem requires the allocation of different parts of the control application to control devices. Following this distribution the different parts of the control intelligence are distributed to control devices, and the necessary interactions among the intelligent units and, thereby, the necessary relationships between devices are defined. The distribution has to follow the control system requirements and has to enable a safe, efficient, and economic control system design and application.

The main drivers for a solution to the above-mentioned problem are the reusability of control application parts as well as the maintainability and flexibility of the control system itself. In addition, the application of mechatronical units hosting the control application and executing the controlled process is intended.

The solution has to ensure that the necessary knowledge about products that have to be produced, including knowledge about necessary manufacturing processes that have to be executed and the necessary knowledge about manufacturing resources that have to be applied (production system knowledge) is integrated within the control design process and, ultimately, within the control application. Here, the mechatronical units will play an important role since they usually have to convey the necessary control application parts relevant for the control of the physical manufacturing process in the layer of sensor and actuator access, and the further control application parts related to product manufacturing are distributed among mechatronical units.

The solution of the mentioned problem is based on the definition of control functions containing control system building blocks and their unique allocation to control devices. The combination and interaction of these control functions will establish the control application.

The control functions associated to the control devices can be classified within two main function classes.

The first class consists of process functions. These functions are used to control the manufacturing process and its progress. They provide a complex system of manufacturing process steps that largely enable the manufacture of the desired products. The process functions are sequenced in a so-called half-order relation. Each process function may have predecessor steps, successor steps, and parallel (concurrent) steps. This half-order, together with the individual process functions, represents the overall product knowledge contained in the control application.

The second class of functions are safety functions. These functions contain all control activities necessary to safeguard machines, material, environment, and human beings against dangerous events. Safety functions interact with other safety functions as well as with process functions. They act as observers for process functions to prevent dangerous situations caused by the activities of process functions. These interactions, together with the safety functions and the process functions, will represent the production system knowledge contained in the control application.

Each control function needs to be able to interact with other control functions crossing device borders. Hence, the control system has to provide a device-independent interaction mechanism for control functions enabling a control function interaction without the knowledge of the function location on control devices. This device-independent interaction mechanism has to be provided by the control device interaction system.

The described solution is depicted in Fig. 8.1.

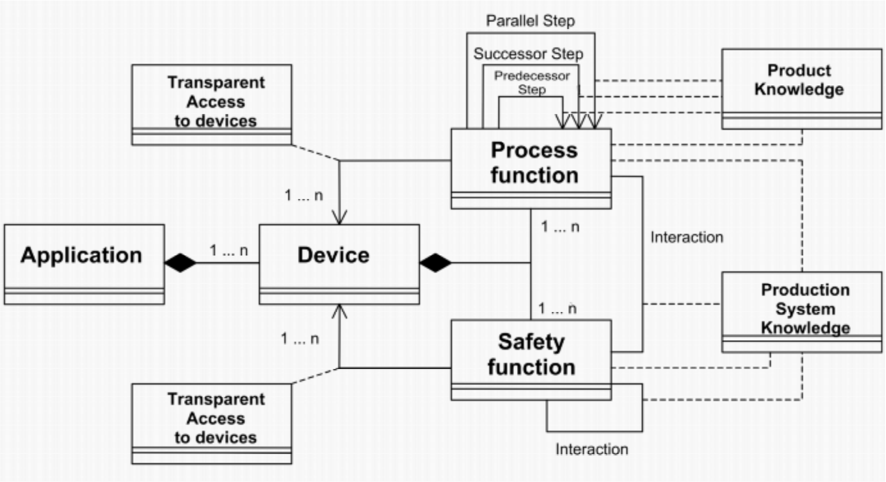


Fig. 8.1 Design pattern – Distributed Control Applications

The described solution enables an easier handling of control functions within a distributed control application related to the devices hosting the control application. This property is especially important for device-independent designed and automatically distributed control applications as often generated within Model Driven Engineering (MDE)-based control application design processes [16, 17]. In this case a set of control functions will be distributed among a set of devices without user intervention. Here it is necessary to automatically establish the necessary communication paths among functions to enable their interaction. The provided solution will enable this automatic integration of interaction mechanisms.

A main influence on the described solution has been the International Electrotechnical Commission (IEC) 61499 standard [18]. This standard describes a function-block-oriented structure for specification, implementation, and analysis of control systems using event-based interaction mechanisms for control function interaction. It is useful for the implementation of distributed control applications since it enables a direct realization of the above-described solution. The IEC 61499 standard is widely considered in research resulting in various applications and control engineering design methodologies [19].

The solution is based on the experiences made during the realization of distributed control applications and the development of new control design tools for the

Field Control Level. The main drivers have been the research activities within the research projects TORERO (Total lifecycle web-integrated control) [20, 21], JAKOBI (Java und komponentenbasierte Industriesteuerung) [22], and PABADIS'PROMISE (PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises) [23, 24] as well as practical experiences within industrial projects.

The application of the described solution is not limited to a certain manufacturing area. In addition it is not limited to a certain programming language- but object-oriented or function-block-oriented languages are preferred.

8.5.2 Design Patterns – Reusability of Control Software Building Blocks

In the same context of distributed control systems based on intelligent and nonintelligent devices solving a common control problem, the problem of control software building block reuse also emerges.

Distributed control applications have to reflect the modular design of manufacturing systems based on mechtronical units. System designers enforce the standardization and the reuse of these mechtronical units. This concerns manufacturing modules as well as other system parts necessary for the functionality of a manufacturing system like communication systems or data management systems.

Following this trend and based on the aim of economical efficiency and correctness of control systems it is useful to design control applications based on control application building blocks and to reuse existing/tested building blocks within other control projects.

But this objective leads to the problem of control application decomposition enabling a most efficient reuse of control application building blocks.

The main driver for the solution of the reuse problem is the possibility of different viewpoints for the decomposition of manufacturing systems and manufacturing system control applications. The decomposition into control application building blocks can be based on the following decomposition rules:

- Assignment of different production functions and safety functions to different building blocks
- Assignment of different mechatronical units to different building blocks
- Assignment of different technologies to different building blocks
- Assignment of different technological parts of a device to different building blocks
- Assignment of different devices to different building blocks.

The intended solution of the decomposition problem has to be as flexible as possible to enable all these decomposition rules and, thereby, enable the automatic decomposition/composition of control applications and its automatic distribution among devices.

In addition, the intended solution has to enable the easy replacement of building blocks to make it easy to upgrade/redesign the system. This will increase the application fields of control building block reuse.

The solution to the above-mentioned decomposition problem is based on the distinction of equipment (control devices, sensors, actuators, communication system, material, human intervention, etc.)-dependent and equipment-independent control application building blocks. The equipment-dependent control application building blocks can be seen as the drivers of the physical capabilities of the mechatronical units involved covering the control of the physical manufacturing processes as well as the control of the involved devices. The equipment-independent control application building blocks can be seen as the control application parts resulting from the control of the product-oriented manufacturing process execution. Together all these building blocks reflect the internal logic of the control application.

The set of equipment-independent control application building blocks contains the control logic that is independent from the technologies applied in the controlled system. They reflect the technology-independent product and production process knowledge of the control system. An example of the contents of these building blocks is the sequence of necessary manufacturing steps to create a certain product within a manufacturing system.

The set of equipment-dependent control application building blocks contains three subsets for equipment control functions, for example controlling a special drill within a drilling machine, communication system access, for example the Ethernet-network-card-based access to a MODBUS TCP communication system, and hardware access, for example to direct input/output cards. These three types are similar in their functionality since they provide access to the plant but they have different internal and application characteristics in their functionality as proxies for plant access.

The control logic in the equipment-independent control application building blocks uses the equipment control functions for the realization of their control tasks. They access the equipment functions via a generic equipment function interface that will be configured with respect to their cooperative use by more than one control logic building block. In addition the control logic building blocks will use the communication building blocks for communication with other control logic building blocks and with upper-level control systems.

The equipment functions themselves will use the communication system access building blocks as well as the hardware access building blocks for the completion of their control tasks. Again the access to the communication and the hardware function building blocks is made by a generic and configured interface mapping logic variables and functions to physical access.

The described structure is given in the following class diagram (Fig. 8.2).

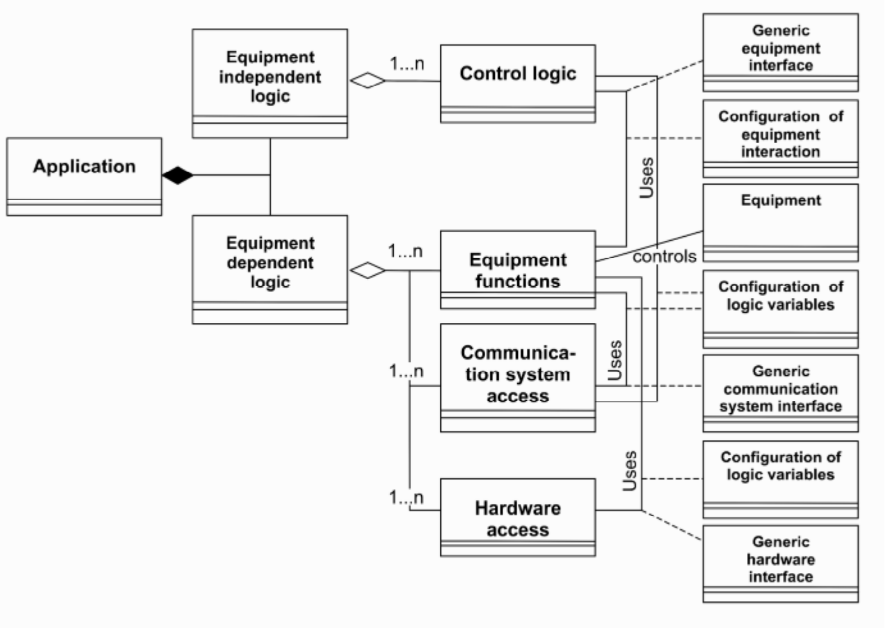

Application
Equipment independent logic
Equipment dependent logic
1...n
Control logic
1...n
Equipment functions
1...n
Communica-tion system access
1...n
Hardware access
Uses
controls
Uses
Uses
Generic equipment interface
Configuration of equipment interaction
Equipment
Configuration of logic variables
Generic communication system interface
Configuration of logic variables
Generic hardware interface

Fig. 8.2 Design pattern - reusability of control software building blocks – application view

This structure enables to a large extent the decoupling of the different layers and their interconnection by generic and configurable interfaces. This is depicted in the following example of a transportation system (Fig. 8.3). This transportation system consists of different turntables and conveyers. Within the overall control logic a transport sequence can be implemented independently of the used conveyers and turntables for their control. The control of the turntables can be implemented independently of the access path to the physical devices (drives and position sensors). The actual sequence of turntables and conveyers controlled by the control logic will be configured within the generic interface between control logic and turntables/conveyers, and the association of turntables/conveyers to physical devices will be configured, for example, in the interface to the communication system enabling the physical system access (Fig. 8.3.).

The combination of different building blocks within a control application follows three main application scenarios. These three scenarios are based on the interaction of a control logic building block with a plant, another control logic building block, or an upper-level control system like MES or ERP.

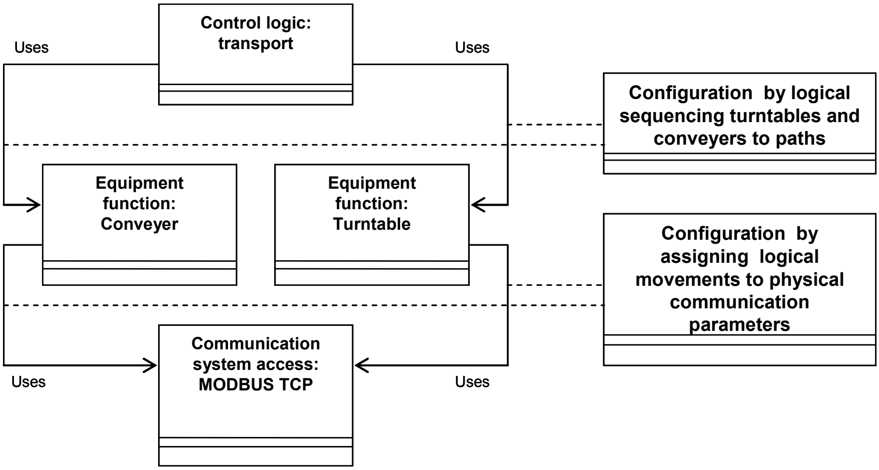

Control logic: transport
Uses
Uses
Configuration by logical sequencing turntables and conveyers to paths
Equipment function: Conveyer
Equipment function: Turntable
Configuration by assigning logical movements to physical communication parameters
Communication system access: MODBUS TCP
Uses
Uses

Fig. 8.3 Design pattern - reusability of control software building blocks – access view

The interaction between control logic building block and plant is carried out by using equipment function control building blocks and its access to hardware or communication building blocks depending on the possible access paths to sensors and actors. This interaction structure results in a decoupling of the data transmission between plant and control application that is carried out by the hardware access and the communication system access building blocks from the measurement value preparation which is carried out by the equipment function control building block and from the application of the gained data from the plant within the control logic for the overall process. This structure is depicted in the left part of the following collaboration diagram (see Fig. 8.4).

The interaction between control logic building block and upper-level control systems like Manufacturing Execution Systems (MESs) and Enterprise Resource Planning (ERP) systems is initiated by the upper-level control systems. The interaction is realized by using the communication building blocks as interface. This structure enables the decoupling of the accomplishment of the required action of the control logic from the data transmission between both system parts that is carried out by the communication system access building block. This is depicted in the middle of the collaboration diagram in Fig. 8.4.

Finally the interaction between two control logic building blocks is also based on the application of communication building blocks as generic and configurable interface. In this way, the decoupling of data transmission between control logic building blocks (executed by the communication building blocks) from the accomplishment of control actions (executed by the control logic building blocks) is realised. This is depicted in the right hand side part of Fig. 8.4.

The described solution enables a widely independent development of control system components with respect to the correct control of manufacturing system building blocks, the correct application of communication systems, and the correct interaction of system components. The very generic interfaces of the different building blocks enable an automatic composition of control applications based on these building blocks integrating predefined or vendor-delivered control system components.

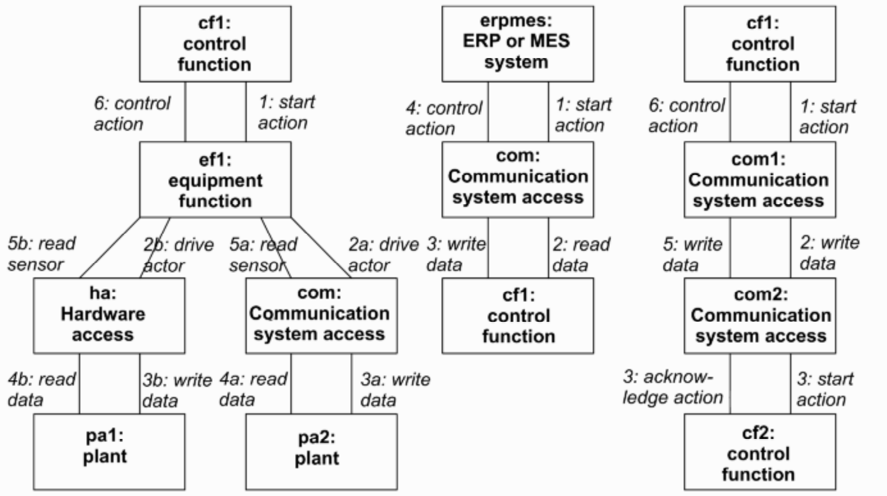


Fig. 8.4 Design pattern - reusability of control software building blocks – building block interaction

Like the design pattern “Distributed Control Applications”, the described solution is based on the experience gained during the realization of distributed control applications and the development of new control design tools for the Field Control Level. The main drivers have been the activities within the research projects TORERO [25], JAKOBI [26], and PABADIS’PROMISE [24], as well as practical experiences within industrial projects. Additionally, experiences in the field of software design specifically the layers pattern [5, 6] had a primary influence on the solution.

Like the pattern “Distributed Control Applications”, the application of the described solution is not limited to a certain control field or to a certain programming language. Its optimal application will come from using object-oriented programming languages.

8.5.3 Design Patterns – Devices Within Distributed Control Systems

Within the context of distributed control systems based on intelligent and non-intelligent devices solving a common control problem, the problem of the efficient structure of control devices dedicated to the distributed nature of the control system also arises. Additionally, the control device structure has to reflect the integration of control devices within mechatronical units. Here the devices have to be able to interact within the mechatronical unit to control it as well as to interact with other control devices across mechatronical unit borders.

Hence, distributed control systems require control devices that enable a cooperative solution of a common control problem by using a distributed control application based on control system building blocks. Therefore, the control devices have to be prepared from the technological, structural, and architectural points of view. The emerging requirements for control devices are based on the necessity of running control application building blocks in a cooperative and coordinated way and the necessity to enable communication between control building blocks by enabling communication among control devices.

This problem has to be considered in view of the existing control device technology and the current trend for control system device improvements. These devices have to be adapted to the necessities of distributed control applications. Existing generations of devices should be incorporable within distributed control systems without major modifications.

Since the described solution implies the application of software building blocks on the control devices, intelligent devices as intelligent Programmable Logic Controllers (PLCs), industrial PCs, or embedded control devices within sensors and actuators are essential for the solution of the problem. They have to be considered as the basis of the solution described below. The solution of the mentioned problem is based on a bipartite structure of hardware and software entities that are functionally related. The hardware part consists of the following five parts:

- One or more processors providing the necessary computing power,
- One clock for temporal coordination of the interaction and cooperation of the different control devices with respect to necessary temporal interrelated control actions,
- One or more memory building blocks containing all necessary information like control code, data at runtime, documentations, etc.
- One or more communication networks (resp. network cards) used for physical access to communication media, and
- Zero, one, or more physical inputs and outputs (the physical connection).

The software part consists of two main component classes of entities. The first class contains one or more control application building blocks running on the control device. The second one contains

- The operating system,
- One ore more communication protocols, and
- Software drivers needed to access the physical Input/Output (I/O) interfaces.

Each control device is uniquely associated to one ore more plant building blocks that will be exclusively controlled by this device. In this way, plant building bock(s) and control devices comprise a mechatronical unit. To control its plant building blocks the control devices have to contain the software implementation necessary for control logic.

This control logic has to be implemented using control software building blocks as expressed in the design pattern above. These application building blocks will run on top of the operating system using all its functionalities and will use for its interaction with the plant and other control application building blocks running on other devices the communication protocol drivers and the I/O access drivers. These two types of drivers will be governed by the operating system, which will provide an interface to these drivers to the control application building blocks.

The operating system will use for its activities the processor(s), memory bocks(s), and the clock. Each communication protocol driver will use the communication network attached to it, and the drivers for local I/O access will use the physical I/O interfaces.

The access of the control application building blocks to other devices will be provided using the communication protocol drivers and the communication networks. Access to the plant will be obtained using the communication protocol drivers and the communication networks as one access path and the local I/O access drivers and the physical I/O interfaces as the other access path. This structure is depicted in Fig. 8.5.

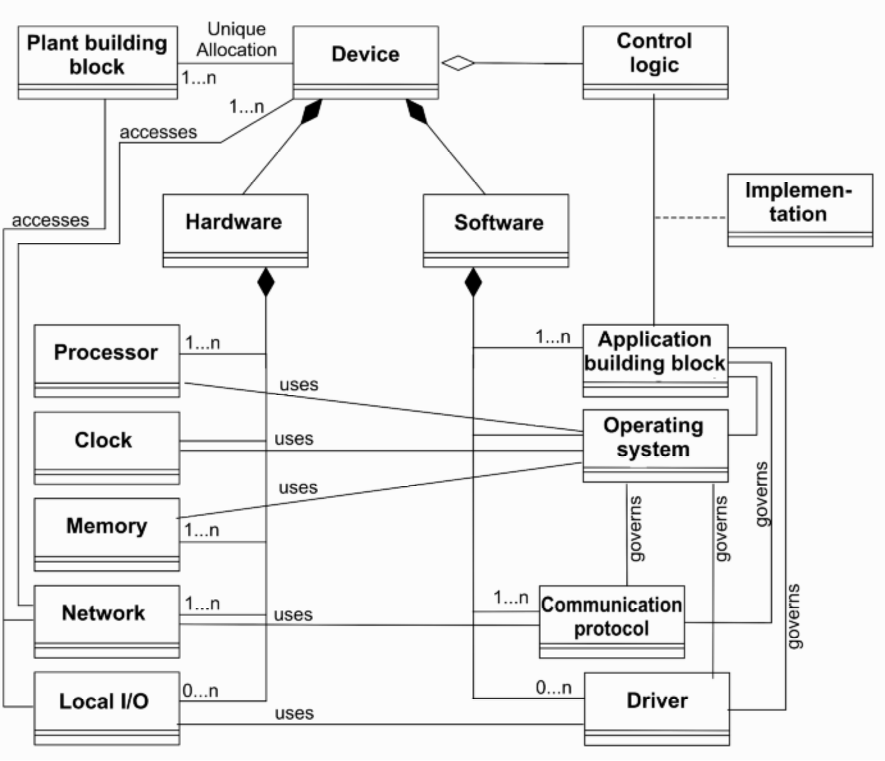


Fig. 8.5 Design pattern - devices within distributed control systems

As in the design pattern "Reusability of Control Software Building Blocks", the described solution will provide a layered structure of hardware and software building blocks establishing control devices within a distributed control system. It extends the pattern "Reusability of Control Software Building Blocks" by hardware and hardware near software building blocks.

Like all previous patterns, the described solution is independent of an application field or a special industry. It is also independent of a special control system implementation language. But it has been developed especially for devices having a PC-like architecture and being integrated into a factory communication network. In this context mainly fieldbusses and Ethernet-based factory communication systems are used. Hence, the solution is dedicated to systems implementing an Ethernet-based industrial communication protocol.

Like the design pattern "Distributed Control Applications", the described solution is based on the experience gained during the realization of distributed control applications and the development of new control design methodologies for the Field Control Level. The main drivers have been the activities within the research projects JAKOBI [26] and PABADIS'PROMISE [24], as well as practical experiences within industrial projects.

8.6 Application of the Design Patterns Within the PABADIS'PROMISE Project

Within the PABADIS'PROMISE project the described design patterns have been used to develop a methodology and an architecture usable for the design and execution of product-related control applications on demand. Here, control applications will be developed only if a product needs to be manufactured.

Based on the design pattern "Reusability of Control Software Building Blocks" the PABADIS'PROMISE field control architecture is based on a layered structure with three main components.

The first component sitting on top of the architecture is the so-called ability application executed by an ability application runtime environment. This ability application constitutes the equipment-independent control logic within so-called Resource Function Blocks coding how the ability application invokes the underlying manufacturing functions as well as equipment-dependent functions within so-called Device Function Proxies (DFP). DFPs implemented as Service Interface Function Blocks (SIFB) that enable access to a Device Proxy (DP) and represent the underlying device functions inside an Ability Application.

These ability applications are IEC61499-compliant function block (FB) networks, which are executed within the Ability Application Runtime. As this runtime is only responsible for the triggering of functionality in field-level devices it is normally not executed under real-time constraints.

The second component of the three-layer structure is constituted by the set of DPs. DPs implement communication system access components of the mentioned design pattern realizing the interaction of the ability application with the underlying manufacturing system controlling (non-)intelligent devices.

These underlying manufacturing system controlling (non-)intelligent devices constitute the third and lowest layer of the structure.

Following the design pattern "Distributed Control Applications" the ability application as well as the underlying manufacturing system controlling (non-)intelligent devices are divided into Process Function Blocks and Safety function Blocks. Especially within the ability application the Resource Function Blocks plays the role of Process Function Blocks while the DFPs take over both roles of Process Function Blocks and Safety Function Blocks.

The design pattern "Devices within Distributed Control Systems" was the main driver for the design and implementation of the ability application runtime environment. It constitutes a type of operating system executing the model-based ability applications and providing communication and access functionalities over the DPs.

The resulting structure is given in Fig. 8.6.

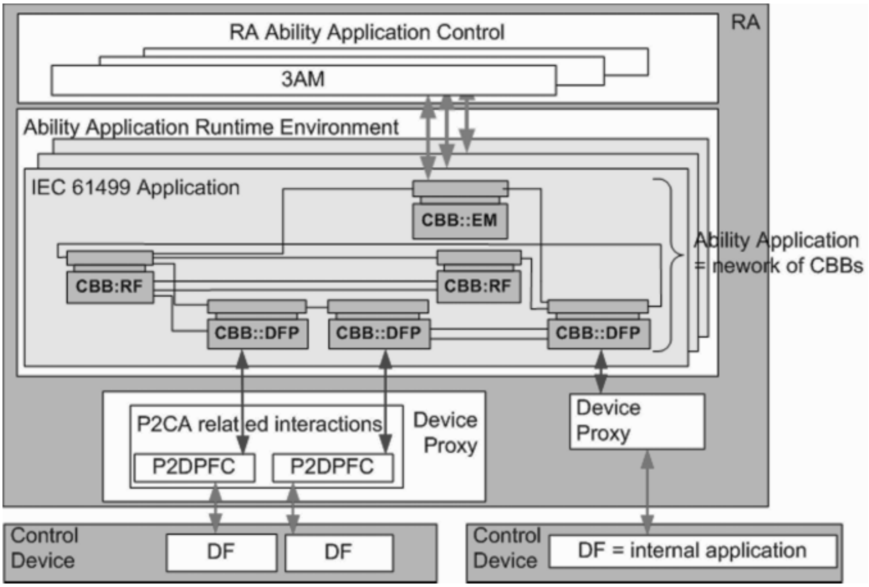

RA Ability Application Control
3AM
RA
Ability Application Runtime Environment
IEC 61499 Application
CBB::EM
CBB:RF
CBB:RF
CBB::DFP
CBB::DFP
CBB::DFP
Ability Application = nework of CBBs
P2CA related interactions
P2DPFC
P2DPFC
Device Proxy
Device Proxy
Control Device
DF
DF
Control Device
DF = internal application

Fig. 8.6 Application of the design pattern within the PABADIS'PROMISE control architecture

The essential benefit provided by the PABADIS'PROMISE field control architecture is the capability of the system to handle field control applications in a vary flexible way. These capabilities include

- The ability to add, change, replace, and delete manufacturing system controlling (non-)intelligent devices to the system without changing the ability applications during runtime and, thereby, to make the underlying manufacturing flexible;
- The ability to change ability applications during runtime enabling product-related flexibility; and
- The ability to change manufacturing system controlling (non-)intelligent devices-access structures and, thereby, enabling a resource-application- and manufacturing-process-related flexibility, for example, to increase process optimality.

All in all, this field control architecture enables a maximal flexibility based on the application of the three described design patterns.

8.7 Conclusion

In this chapter we have described a set of fundamental design patterns that can be used in the field of distributed automation. These design patterns cover the field of distributed applications, reusability of control application building blocks, and the device structure and functionality necessary for distributed control systems in light of the recent requirements of manufacturing system control as well as the strong trend toward the application of mechatronical units.

Following this design pattern it will be possible to design control systems and control applications in an efficient way.

The design patterns introduce a layered structure for distributed control systems that enables a structured design for distributed control systems. In a first design step a manufacturing system resource-independent control application part will be designed. This control application part will be combined in a second step with predesigned control application building blocks necessary for the control of the used manufacturing system resources and control application building blocks necessary for access to communication systems and physical I/Os.

A control application that has been designed in this way is much easier to maintain and to change with respect to new communication systems, manufacturing system resources, or products. In addition, it can be used for automatic control application building block distribution among devices, which is a huge benefit for distributed control. Finally, this procedure can be exploited within an approach designing control applications on demand related to manufacturing orders as successfully developed within the PABADIS'PROMISE project [24].

References

[1] Alexander, C. 1979. *The Timeless Way of Building*. New York: Oxford University Press.

[2] Gamma, E., R. Helm, R.E. Johnson and J. Villisides. 1995. Design Patterns, Elements of Reusable Object Oriented Software. Reading: Addison-Wesley.

[3] Gamma, E., R. Helm and R.E. Johnson. 2004. *Entwurfsmuster - Elemente wiederverwendbarer objektorientierter Software*. München: Addison-Wesley.

[4] Freeman, E., E. Robson, B. Bates and K. Sierra. 2004. *Head First Design Patterns*. O'Reilley.

[5] Sanz, R. and J. Zalewski. 2003. Pattern-based control systems engineering – using design patterns to document, transfer and exploit design knowledge. *IEEE Control Systems Magazine*, June: 42-60.

[6] Perronne, J.-M., L. Thiry and B. Thirion. 2006. Architectural concepts and design patterns for behavior modeling and integration. *Mathematics and Computers in Simulation*, 70: 314-329.

[7] Thramboulidis, K. 2008. Challenges in the development of mechatronic systems – the mechatronic component. In: *Proceedings of the 13th IEEE International Conference on Emerging Technologies and Factory Automation*. pp. 624-632.

[8] Habib, M. 2007. Mechatronics – A unifying interdisciplinary and intelligent engineering science paradigm. *IEEE Industrial Electronics Magazine*, 1 (2).

[9] Wagner, T., A. Schertl, J. Elger and J. Vollmar. 2008. Evaluation of Effectiveness and Impact of Decentralized Automation. In: *Proceedings of the 13th IEEE International Conference on Emerging Technologies and Factory Automation*. Proceedings: 1128-1137.

[10] Sanz, R., A. Yelaand and R. Chinchilla. 2003. A Pattern Schema for Complex Controllers. In: *Proceedings of IEEE Conference on Emerging Technologies for Factory Automation*, October, Lisbon.

[11] Lüder, A., J. Peschke and R. Sanz. 2006. Design Patterns for Distributed Control Applications. *ATP International*, 3: 32-40.

[12] Buschmann, F., K. Henney and D.C. Schmidt. 2007. Pattern-Oriented Software Architecture: *A Pattern Language for Distributed Computing*. Wiley.

[13] Schmidt, D., M. Stal, H. Rohert and F. Buschmann. 2000. Pattern-Oriented Software Architecture: *Patterns for Concurrent and Networked Objects*. Wiley.

[14] Lea, D. 1994. Design patterns for avionics control systems. State University of New York, Oswego, Tech. Rep. ADAGE-OSW-94-01, November, dSSA ADAGE Project.

[15] Pont, M. 2001. Patterns for Time-Triggered Embedded Systems: *Building Reliable Applications with the 8051 Family of Microcontrollers*. London: Addison-Wesley.

[16] Schmidt, D. 2006. Model-Driven Engineering [online]. *IEEE Computer*, 39 (2): 41-47. Available from: http://www.cs.wustl.edu/~schmidt/PDF/GEI.pdf [Accessed October 2008].

[17] Gérard, S., F. Babau and J. Champeau. 2005. Model Driven Engineering for Distributed Real-time Embedded Systems. Wiley.

[18] IEC 65/240/CD 61499. Function Blocks for Industrial Process Management and Control Systems - Part 1: *Architecture. International Standard*. Available from: www.iec.ch.

[19] Vyatkin, V. 2007. IEC 61499 Function Blocks for Embedded and Distributed Control Systems Design. O3NEIDA and ISA.

[20] Tangermann, M., C. Schwab, A.P. Kalogeras, K. Lorentz and A.S. Prayati. 2003. Aspect-Orientation of Control Application Code for Distributed Automation Systems: The TORERO Approach. In: *Proceedings of the OTM Confederated International Workshops Java Technologies for Real-Time and Embedded Systems*. November, Catania. Lecture Notes in Computer Science - LNCS 2889, Heidelberg, Berlin: Springer.

[21] Tangermann, M., C. Schwab, A. Lüder, L. Ferrarini and C. Veber. 2004. Encapsulation of IEC 61499 Function Blocks Using Real-Time Java according to the RTSJ. In: *Proceedings of the 2nd Workshop on Java Technologies for Real-Time and Embedded Systems*, Larnaca, Cyprus. Heidelberg, Berlin: Springer.

[22] Peschke, J. and A. Lüder. 2005. Java Technology and Industrial Applications, The Industrial Information Technology Handbook In: *Zurawski, R., ed., Industrial Electronics Series*, CRC Press: 63/1 – 63/15.

[23] Lüder, A., J. Peschke, A. Bratukhin, A. Treytl, A. Kalogeras and J. Gialelis. 2008. Order Oriented Manufacturing Control – The PABADIS'PROMISE approach. In: *Raabe, M. and P. Mihok, eds., New technologies for the Intelligent Design and Operation of Manufacturing Networks*. Fraunhofer IRB Verlag.

[24] Heinze, M., J. Peschke and A. Lüder. 2006. Resource Management and Usage in highly flexible and adaptable Manufacturing Systems. In: *Proceedings of the 13th IEEE International Conference on Emerging Technologies and Factory Automation*, September, Hamburg.

[25] Schwab, C., M. Tangermann, A. Lüder, A. Kalogeras and L. Ferrarini. 2004. Mapping of IEC 61499 Function Blocks to Automation Protocols within the TORERO Approach. In: *Proceedings of the 2nd IEEE Conference on Industrial Informatics*, Berlin. pp. 149-154.

[26] Peschke, J. and A. Lüder: The JAKOBI architecture – a distributed dynamic execution environment in Java. In: *Proceedings of the 3rd International IEEE Conference on Industrial Informatics*, August, Perth.

9 Conclusions and Outlook

This edited book has presented a selected range of contributions, based on the results of the EU Plant Automation Based on Distributed Systems (PABADIS) and PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises (PABADIS'PROMISE) projects: from methodological to theoretical to address the issues surrounding Distributed Manufacturing. The contributions neither claim to have completed all the research necessary on the field nor to be the final stage of development of principles, methods and instruments resulting from network thinking in manufacturing. It must be noted that the "distributed" perspective on manufacturing, implicitly introducing the modes of iteration, parallelism, emergence, behaviour and encapsulation, reveals a surprising number of novel functions, creates new problems for methodologists, and generates demands for better information and communication technologies (ICT) pervasion. However, for now this recognition can only partly result in coherent descriptions of relevant (interdisciplinary) contributions for Distributed Manufacturing.

This volume will not conclude without an attempt to synthesise all approaches outlined and to summarise key challenges that further research needs to address in Distributed Manufacturing and the implications that this way of interpreting manufacturing and resulting research has on practice.

9.1 Contributions of the Book

What follows from the contributions is that the potential of Distributed Manufacturing is manifold. It covers a number of different areas, all enhancing flexibility, efficiency and performance.

Firstly, many approaches as well as contributions appear from different disciplines, a fact that is generally cited as being very fruitful, as it nearly guarantees novel solutions and mind opening insights. Aside from the complexity thinking there are also substantial inspirations from neurobiology, topology, computer science, social behaviour studies, and artificial intelligence. This may be interpreted as a lack of theory as a solid base for manufacturing networks in general and Distributed Manufacturing in particular. Challenges will be the resistance to overcome systems thinking while at the same time transferring all comfortable properties, such as decomposition, formalisms to describe input/output and behaviour, or the selection of aspects and attributes for the networked world.

Secondly, the involvement and the exploitation of complexity principles for manufacturing networks are inevitable and favourable at the same time. Their consideration synthesises new philosophies and progress in technology. The challenges for an organisation operating loosely connected units and actors as customer-oriented networks bring us back to the potentials of ICT and the web. Distributed computing, data warehousing and knowledge bases are contributing research fields on the data processing side; on the communication side, security issues, pervasive technologies, augmented virtual reality, blogging and shared data applications are important fields.

Thirdly, the harmonisation of models and procedures used for management and successful coordination of network behaviour can be achieved by reasonable input of resources. Practitioners that are fully exposed to the challenges of Distributed Manufacturing networks still have to rely upon accustomed instruments that are developed for one time static setups' management. The contributions confirm the evidence that next generation instruments will have to support the optimisations of 'transvariable' indicators (such as value addition, time) instead of 'intervariable' attributes (such as output, volume). Network performance is measured by the quality and the speed of linkage. The specific experiences with prototypes underpin the fact that next generation management will operate also on methods originating from Distributed Manufacturing as described.

Distributed Manufacturing is a powerful structure for coping with market volatility by sharing knowledge, resources and risks in new product developments. For the success of these activities, the concurrency of operations is a vital issue. In Chap. 1 Hidalgo thoroughly explores concurrency in product development in the much tight interfirm context of the extended enterprise (E2). At this level he sees a field of distributed product development as the aggregate pattern of product introductions emerging over time. In order to cope with volatility he calls for robust product families for the creation of numbers of derivative products, founded on a common core technology – the product platform. This idea constitutes a red thread throughout the book as design pattern in Chap.6 or as ability application environment in Chap.7. Success results from concurrent evolution of product families, platforms and product creation. Only close collaboration and intensive standardisation may provide efficient common processes as well as successful innovation. The resulting way of value creation assumes new instruments, other coordination mechanisms and interlinking ICT tools, which are outlined in principles and examples. This statement is repeated in Chap. 3 with respect to factory design.

More generalised and motivated by biological evolution, and not confined to new product development, Dekkers discusses the very "prototype" of collaboration and concurrency, co-evolution. The basic idea is to identify collaboration as a strategy for the phenotype to survive, whose success is expressed in the fitness of an agent or an entity. It is the basis for descriptions of dependencies and mutual developments, which yields additional insights on the interaction and interrelation of entities. Companies engage in new relationships in order to increase their fitness underpinning the importance of new and more effective models to support the dynamics of collaborative relationships, as proposed in Chap. 4 by different highly collaborative factory set-ups or in Chap. 5 by several "X"-reality approaches or community approaches. The theories developed there are very valuable ingredients for a synthesis to advance our understanding of concurrent enterprising and how distributed structures address collaborative challenges in industrial networks. Therefore they define the framework for the metrics and indicators to be applied to the network topology that can only logically describe network set-ups and processes representing valuable input for software design and software agent programming.

Wünsch et al. explore the technical and operational interpretations (focusing on ICT) of distribution and concurrency by synthesising various aspects of flexibility, product, technology and resource. Three prevailing information technology paradigms – object orientation (OO), service oriented architecture (SOA) and agent oriented architecture (AOA) – are mirrored in the most important manufacturing principles that have emerged in leading companies in order to better cope with the concurrency needs of multistream processes. These principles may be seen as responses to environmental problems (green manufacturing), network thinking (collaborative manufacturing, visual manufacturing, mobile manufacturing, and reconfigurable manufacturing) and decentralised parallel computing possibilities (open manufacturing, harmonised manufacturing, event-driven and real-world manufacturing). All items discussed are viewed as constituents of mechatronic systems as the preferred application area for next-generation Enterprise Resource Planning (ERP). The authors could verify that the "PABADIS world", which serves as the foundation of all contributions of this volume, perfectly enables one to exploit the three information technology paradigms and can be engaged to successfully cope with all identified challenges. It is concluded that next-generation planning and execution systems will be based on architecture like PABADIS fully integrating process and management levels. This comes up in all Chaps. 4-8 as a result of commonly used ICT standards, but especially for coherently applied methods and platforms.

It is generally anticipated that people will be working more as dynamically assembled groups of diverse and complementary skilled professionals within an enhanced collaboration environment. Therefore, a number of communication means have been successfully introduced, each offering specific profiles for support. Pallot and Bergmann take a critical look at such implementations revealing that most of the set-ups do not need teams working together in one place, although most of the solutions are applied in this way. The team members may be distributed at a number of different sites. Moreover skills and benefits as well as adoptions of such communication media prove to co-evolve with all other ICT applications, a key point in Chap. 3 as well. These findings again confirm instinctive feelings of managers strongly signaling that especially the gains of the prefabricated expensive "X"-reality set-ups could lag far behind expectations even in the long term. Companies which are enthralled by the potential of augmented reality (AR) to communicate e.g assembly instructions in real time in their factories should consider the potential damage that could arise if skills learning were not enabled. Chap. 4 endorses other recent experiences and field study outcomes that, contrary to all predominant views, the formulation and communication of information should be designed using ICT, and distributed decision support may be added using agent architecture as such multi-agent systems (MAS) set-ups outlined for manufacturing execution systems (MES) in the Chaps. 5-7. In particular, the failure to develop and harness human expertise can lead to companies becoming uncompetitive, because of expensive and powerful ICT investments as well as the role of agents to better meet the requirements is one important strand of research, outlined by Bratukhin et al.. Agents are introduced as means to decentralise decision making by detaching it from the top levels. This is a precondition for the decentralisation of control. The design patterns used to describe the control set-ups are interpreted as generically consisting of two main elements: resources and costumer orders. To cope with complexity the agents' structure is aggregated or disaggregated hierarchically, by implicit use of self-similarity principles as well as synchronisation methods to ensure concurrency options. The basic structures addressed by the majority of the most important approaches are derived from these generic set-ups. The advantages of all approaches are synthesised for advanced MES solutions in factory automation which is compatible with the resource planning levels as discussed in Chap. 3.

Sauter and Tretyl take up Distributed Manufacturing from the fieldbus level and the variety of solutions that already exist, exploring the limitations of real-time applications. A major issue is finding suitable mechanisms for synchronisation and security. The historical review leads, over important communication paradigms – client-server, producer-consumer, and publisher-subscriber models – to the potential of the Internet with its multiple ways of synchronising and providing for security. Tightly coupled as well as loosely coupled units may be connected by the World Wide Web and installation of rigid zones and flexible domains. The set-up proposed by PABADIS is strongly emphasised as adequate and most instructive for future automation systems. Moreover, bionics is recommended as the general view to be followed and softer models, as outlined in by Bergmann and Pallot in Chap. 4, have to be increasingly included if higher levels of complexity for distributed manufacturing are to be covered.

Lüder et al. thoroughly analyse the distribution issues on the lowest control level implemented by field control systems. Here the focus is on knowledge preservation related to the most efficient and often used structuring paradigms valid for distributed field control applications. Therefore, the design pattern as an appropriate means to code solution knowledge within different areas of science and practice is explored. Design patterns were initially developed in architectural sciences for describing generic solutions in the problem area as well as the solution context. They quickly have been transposed to information science and a number of other disciplines and serves as a perfect base of integration for all preceding contributions. With this last chapter the most appropriate design pattern for the segmentation of field control applications to distributable entities, for enforcement of the reusability of field control application segments, and for positioning of these segments on control devices is outlined. An application case making use of these design patterns is explained highlighting the greatest benefits of the approach.

9.1.1 What are the philosophies to manage Distributed Manufacturing? Which paradigms and metaphors should be emphasised and encouraged for support?

The concept of the monolithic manufacturing company is definitely obsolete. The "certainties" and the full "hierarchic controllability" of companies have always been comfortable pretences or even most welcome excuses for inflexibilities or insistence on organisational standards. Under pressure and in dynamic environments such illusions vanish, and – upheld through inertia – will even develop into a fatal threat for companies. As successors several new concepts have arised. These concepts are not just static and systematic; they draw from such dynamic backgrounds as the theory of constraints (lean manufacturing), biology (bionic manufacturing), nature (holonic manufacturing) and geometry (fractal organisation). None of these concepts may fully cover the Distributed Manufacturing field. Complexity modes (encapsulation, iteration, behaviour) as well as concurrency structures (parallelism, emergence) had to be emphasised to furnish adequate grounds for engineering and management of Distributed Manufacturing. Collaboration frameworks (e.g. company footprints), facilitating self-organisation and guidance of self-interested interrelated actors, accelerate implementations and instant stabilisations of reliable structures and help to define adequate measuring and monitoring indicators.

9.1.2 Which disciplines and models are likely to further develop the methods and instruments for Distributed Manufacturing structures? How about the trends in information technology and their effects on coordination and management of inter-organisational value chains?

Among the contributing disciplines we find complex adaptive Systems theory, decision theories, sociology, theory of multi-agents, evolutionary biology, zoology, organisation theory, topology, artificial intelligence, and network management. Within all these disciplines there have been successful attempts to investigate and describe the phenomena of collaboration and networks and the changes that are taking place in relation to industrial organisations. However, to date there has not been published an edited comprehensive account of the different perspectives that exist among the various academic and industrial research communities striving for an emerging new network science.

Ambient intelligence and mobile connection of (distributed) expertise, knowledge and creativity will be the important next ICT features. Open data sharing networks and social webs could link people, units and organisations. Business legal entities will be complemented by social legal entities. Cutting-edge knowledge and high-skilled individuals will be increasingly organised outside of manufacturing companies e.g. in professional virtual communities (PVCs) or comparable organisations. Products and value-adding services will increasingly rely on embedded intelligent devices, affecting the users' individual security and privacy. As increasing numbers of people will neither be able to understand the mechanisms of the (also networked) products nor classify the risks of the products and processes, deeper involvement of people in development and innovation, e.g. by offering open source or living laboratories, might become an important company activity. Anticipating problem areas, synthesised with company-specific sustainability priorities, might strongly direct companies' public relations activities. Accounting may have to provide transparency in all areas pointing out successful contributions to higher societal goals as well in order to maintain the highest acceptance of its behaviour. Surely, a manufacturing world of collaborative ICT is emerging. Human resource policies should put a premium on collaboration skills, ICT capabilities and readiness for IP connection anywhere and anytime, as these will be the decisive factors.

9.1.3 How can companies self-position in times of vanishing distinction of organisations from their environment? Do organisation theory and management science need to be extended by a number of new chapters covering decentralised and distributed processes and value chains?

The scope of collaboration among enterprises has widened substantially. In Distributed Manufacturing the creation of value will take place simultaneously and at various locations. Knowledge about the capabilities of potential partners is becoming a part of the know-how of an enterprise. A company thus needs to continuously determine its position in the market as well as within the network. One key attribute is the core capability, as the Distributed Manufacturing approach relies on the implications of variation, selection and retention. In the manufacturing networks arena, there are specific capabilities in a network of manufacturing plants above and beyond those at the factory plant level. The extension of capabilities to beyond the firm calls for a process-based analysis of capability development, as well as an outcome-based assessment at the strategic level. A central issue is the development of metacapabilities, involving "selective resource picking"; the selection process seeks to achieve a particular outcome or alignment to the business model and thereby provides for a unique "metalevel capability" that is not yet understood. These metacapabilities seem to be company (or network) specific, customer relevant and business aligned.

Apart from the basic trust generally expected by business ethos, the level of trust in networks is closely linked to the partners' behaviour. Real trust builds e.g. under extraordinary circumstances in which one partner is willing to meet exceptional requests, above and beyond the agreed terms of business. The mechanisms for establishing and keeping high levels of trust, absolutely crucial for manufacturing networks, still are very little understood. Trust is the foundation of collaboration on the MES level as well, where most of the security problems in the ICT fields remain unsolved. Technical as well as organisational solutions are far advanced however, important lessons have not yet accessed the inner body of knowledge in manufacturing and management. Interesting possible advances, such as e.g. web-based manufacturing operating globally distributed technical units via Internet communication, have therefore not been made. ICT education and training on software and devices has not reached maturity levels permitting to enable entire staffs of manufacturing networks skilled and efficient use at reasonable cost. Most productive technologies and solutions remain restricted to ICT expert use so the broad effects lag far behind expectations.

Whereas concurrency thinking is well established in engineering, its extension to enterprise networks is a very recent phenomenon. This fruitful extension takes place gradually as many benefits and achievements in distributed manufacturing structures fertilising quasi-unconscious and unarticulated approaches in numerous fields originate in concurrency set-ups. All chapters in this book indicate that the work is not yet finished. In fact, all authors point to avenues of future work.

9.2 Implications for Practice

What follows for industrial practice is that non-hierarchical views of manufacturing will become fully established. In the first place, the concepts of planning and control will replace central, sequential, rhythmic and time-sliced procedures by event-driven parallel distributed evolving logics. Manufacturing will introduce and apply new types of methods and tools, supporting linkage and reconfiguration as well as acquisition of high-level plug and produce, plug and participate and concurrent work skills. Atomisation of production equipment and flows into intelligent units will enable every unit to manage and control its flow process autonomously. Humans and units as well as units and units will communicate or even negotiate.

Concerning ICT in industry, maximum attention has to be paid to all developments of networkable devices, as they will increasingly influence organisational structures as well as decision-making procedures. This will be the most important challenge for interoperability and data security resulting in the complete revision of enterprise resource planning system architecture and man-machine interfaces. New worker types may emerge making use of all ICT options and collaborative working environments relying on permanent Internet connectivity and social touch via web applications. An expected shift from individual productivity towards interpersonal productivity could engender novel intellectual property rights (IPR) situations and might force manufacturing companies into stronger involvement in communities of practice and on-line communities. A shift of power towards Professional Virtual Community structures is a possible consequence, so companies are likely to experience that individuals' working contracts will be managed by these organisations and communities, models far beyond any actual management scope.

In addition, the practical work with the prototypes has once more proven the high potential of new ways of collaboration in distributed manufacturing. Mastering complexity will be one of the major topics for future manufacturing. A specific production network management science for developing and incorporating new networking instruments may appear. As the new ways of looking at manufacturing are not yet fully settled in widespread methods and tools, the challenges compel managers to contribute further insight and collaborate with academics to advance both practice and theory.

Reponses to new order volumes will be rather relinking or renegotiating links with network partners and contributing to the common trust base with the highest reliability. Dedicated training activities aimed at promoting teamwork on a worldwide basis are necessary for the emergent concurrent way of working.

On the way to concurrent enterprising (CE), following up beyond Distributed Manufacturing, more research needs to be put on the agenda. As the papers invited for this book indicate, the unanswered questions can be examined through a wide variety of approaches, both theoretically and methodologically. We hope that the papers and themes outlined will spur more research to further examine this fascinating area.

Bibliography

ARICON EU RTD Project (2005) *European Handbook for Virtual Enterprises*.

Arnaud R, Barnes MC (2006) *Collada – Sailing the Gulf of 3D Digital Content Creation*. Transatlantic, Wellesley.

Barabási A-L (2002) *Linked: The New Science of Networks*. Perseus, Cambridge, MA.

Bellifemine F, Caire G, Greenwood D (2007) *Developing Multi-Agent Systems with JADE*. Wiley Series in Agent Technology. Wiley, New York.

Bonabeau E, Dorigo M, Theraulaz G (1999) *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press.

Borghoff UM, Schlichter JH (2000) *Computer-supported Cooperative Work: Introduction to Distributed Applications*. Berlin, Heidelberg, New York: Springer.

Camarinha-Matos LM (2004) *Virtual Enterprises and Collaborative Networks*. Springer, Netherlands.

Camarinha-Matos LM, Afsarmanesh H (2008) *Collaborative Networks: Reference Modeling*. Springer, New York.

Camarinha-Matos LM, Afsarmanesh H, Ollus M (2008) *Methods and Tools for Collaborative Networked Organizations*. Springer, New York.

Child J, Faulkner D, Tallman S (2005) *Cooperative Strategy*. Oxford University Press, New York.

Davis E, Spekman R (2004) *The Extended Enterprise: Gaining Competitive Advantage Through Collaborative Supply Chains*. FT Prentice Hall, New York.

De Berg M, van Kreveld M, Overmars M et al (2000) *Computational Geometry: Algorithms and Applications*, 2nd edn. Springer, Berlin.

Deen SM (2003) *Agent Based Manufacturing - Advances in the Holonic Approach, Advanced Information Processing*. Springer, Berlin.

Dekkers R (2005) *(R)Evolution, Organizations and the Dynamics of the Environment*. Springer Science and Business Media, New York.

Dooley K (2004) Complexity science models of organizational change. In: Poole S, Van De Ven A (eds) *Handbook of Organizational Change and Development*. Oxford University Press.

Foley JD, van Dam A, Feiner SK et al (1996) *Computer Graphics*, 2nd edn. Addison-Wesley Professional, Amsterdam.

Gloor PA (2006) *Swarm Creativity; Competitive Advantage Through Collaborative Innovation Networks*. University Press, Oxford.

Goldman SL, Nagel RN, Preiss K (1995) *Agile Competitors and Virtual Organizations*. Van Nostrand Reinhold, New York.

Kühnle H (1995) L'entreprise fractale. In: Braesch C, Haurat A (eds) *La modélisation systémique en entreprise*. Pôle productique Rhône-Alpes, Paris.

Liker J (2003) *The Toyota Way: 14 Management Principles from the World's Greatest Manufacturer*, 1st edn. McGraw-Hill, New York.

Lomi A, Larsen ER (2001) *Dynamics of Organizations – Computational Modeling and Organization Theories*. MIT Press, Cambridge, MA.

Luecke R, Katz R (2003) *Managing Creativity and Innovation*. Harvard Business School Press, Boston.

Lunze J (2008) *Automatisierungstechnik*. Oldenburg.

Miller K (2004) *Communication Theories: Perspectives, Processes and Context*, 2nd edn. McGraw-Hill, New York.

Oestereich B (2002) *Developing Software with UML*. Addison-Wesley.

Op 't Land M, Proper E, Waage M et al (2009) *Enterprise Architecture – Creating Value by Informed Governance*. The Enterprise Engineering Series/Springer.

Pawar K, Sharifi S (2001) Product Development Strategies for Agility. In: Gunasekaran G (ed) *Agile Manufacturing: The 21st Century Strategy*. Elsevier, USA.

Reichwald R, Möslein K, Sachenbacher H et al (2000) *Telekooperation, verteilte Arbeits- und Organisationsformen*. Springer, Berlin.

Santoro R, Bifulco A (2005) *A Conceptual Framework for "Professional Virtual Communities"*. IFIP International Federation for Information Processing 186: 417-424. Springer.

Schnieder E (1999) *Methoden der Automatisierung*. Braunschweig, Wiesbaden: Vieweg.

Slack N, Chambers S, Harland C et al (1998) *Operations Management*, 2nd edn. London.

Slater M, Steed A, Chrysanthou Y (2001) *Computer Graphics and Virtual Environments: From Realism to Real-time*. Addison Wesley.

Sommerville I (2007) *Software Engineering*. Pearson Studium - Addison-Wesley, München.

Stacey RD (1996) *Complexity and Creativity in Organizations*. Berrett-Koehler.

Stahl T, Völter M, Efftinge S (2007) *Modellgetriebene Softwareentwicklung – Techniken, Engineering, Management*. dpunkt.verlag.

Vyatkin V (2006) *IEC 61499 Function Blocks for Embedded and Distributed Control Systems Design*. O3NEIDA-Instrumentation Society of America, ISA.

Wang L, Nee AYC (2009) *Collaborative Design and Planning for Digital Manufacturing*. Springer.

Warnecke H-J, Hüser M (1993). *The Fractal Company – a Revolution in Corporate Culture*. Springer, Berlin.

Watts D (2003) *Six Degrees: The Science of a Connected Age*. Norton, New York.

Weilkiens T (2008) *Systems Engineering with SysML/UML*. Morgan Kaufmann, USA.

Wigand R, Picot A, Reichwald R (2004) *Information, Organization and Management: Expanding Markets and Corporate Boundaries*. Wiley, Chichester.

Zaremba AJ (2003) *Organizational Communication: Foundations for Business & Management*, 1st edn. Thomson/South-Western, Mason.

Zurawski R (2005) *The Industrial Information Technology Handbook*. Industrial Electronics Series. CRC Boca Raton, FL.

Index