DISSERTATION

System architecture for cost-effective automation of complex enterprise-spanning business processes

ausgeführt zum Zwecke der Erlangung
des akademischen Grades eines Doktors der
technischen Wissenschaften unter der Leitung von

Em.o.Univ.Prof. Dipl.-Ing. Dr.techn. Gerfried Zeichen
E376
Institut für Automatisierungs- und Regelungstechnik

eingereicht an der Technischen Universität Wien
Fakultät für Elektrotechnik und Informationstechnik

von

Dipl.-Ing. Thomas Schmidt
Matr.Nr.: 9326306
Langobardenstrasse 191/55
1220 Wien

Wien, im November 2003

Th. Schmidt

Kurzfassung

Gesteigerter Wettbewerb im globalem Markt und beschleunigte Entwicklungen der Technologien erhöhen den Druck auf die Unternehmen, hochqualitative Produkte und Dienstleistungen mit immer kürzerer "time-to-market" und "time-to-money" zu liefern. Die Effizienz lässt sich mit der Integration und Automation von internen und unternehmensübergreifenden Geschäftsprozessen steigern. Die logische und konsequente Weiterentwicklung der Kommunikation im Bereich e-Business führt zu Extended Enterprises. Darunter versteht man unabhängige Firmen, die sich zusammenschließen und dadurch deren Wettbewerbsfähigkeit steigern.

Das Ziel dieser Dissertation ist die Entwicklung und Implementierung eines Softwaresystems, das die vollautomatische Zusammenarbeit zwischen Geschäftspartnern ermöglicht. Dafür muss das neu entwickelte System durch die Verwendung einer ganzheitlichen und flexiblen Systemarchitektur die derzeit vorhandenen isolierten Integrationsinseln innerhalb der Firmen - entstanden durch den Einsatz unterschiedlicher Unternehmenssoftware, die jeweils nur einen Teil des Produktlebenszykluses abdecken kann - hinter sich lassen.

Die Hauptkomponente des vorgestellten Ansatzes ist die Business Process Engine, die die inner-/zwischen-organisatorischen Geschäftsprozesse der Extended Enterprise abarbeitet. Bei der Ausführung einer Prozessaktivität werden Web Services verwendet, die von Unternehmenssoftwaresystemen und/oder Teilprozessen zur Verfügung gestellt werden. Ziel ist die Unterstützung des gesamten Produktlebenszykluses. Diese auf Web Services basierende Softwarearchitektur ermöglicht erstmals neben der horizontalen Inte-gration zwischen den einzelnen Geschäftspartnern auch die durchgängige vertikale Integration durch alle Unternehemensebenen.

Die Verwendung von Open-Source Softwarekomponenten ist eine gebräuchliche Methode, um die Anforderung "kostengünstig" zu erfüllen.

Damit die bestgeeignetste und ausgereifteste Workflow Engine als Basis für die Business Process Engine ausgewählt werden kann, ist im Rahmen dieser Arbeit eine ausführliche Studie durchgeführt worden. Aufgrund von aufgestellten Entscheidungskriterien fiel die Wahl auf die Workflow Engine "OFBiz", die in dieser Arbeit weiterentwickelt und in die Softwareumgebung eingebunden wird.

Der letzte Teil der Dissertation umfasst die Realisierung eines Prototypen, der im Rahmen des dreijährigen Europäischen Forschungs- und Entwicklungsprojekts "FLoCI-EE" entstanden ist. Für diesen wird ein beispielhaftes und praxisnahes Geschäftsszenario zuerst modeliert und dann implementiert. Die wichtigsten Realisierungsergebnisse und eine grobe Wirtschaftlichkeitsabschätzung des entwickelten Softwaresystems schließen die Dissertation ab.

Abstract

Increased competition in global markets and accelerated technology development raises the pressure on companies to deliver high-quality products and services within decreased time-to-market and time-to-money. Higher efficiency can be achieved by the integration and automation of internal and enterprise-spanning business processes. The logical further development of e-business communication leads to extended enterprises where independent enterprises join forces to be more competitive.

The aim of this thesis is the development and implementation of a new software system, which enables the fully automated collaboration between business partners. For that the designed system has to put the presently isolated integration islands of companies - originated by the employment of several business information systems supporting only parts of the product life cycle - behind ones by the use of a holistic and flexible system architecture.

The core component of the presented approach is the business process engine which processes the intra-/inter-organizational business processes inside extended enterprises. During the execution of each process activity, Web services of enterprise information systems and/or sub-processes are utilized to support the whole product life cycle. For the first time, this Web service-based software architecture enables both the horizontal integration among business partners and the continuous vertical integration of all levels inside one enterprise.

The usage of open-source software components is a well-established option to comply with the requirement "cost-efficient". To find the most suitable and mature workflow engine as foundation for the business process engine, a substantial survey is done in the course of this work. Considering the decision requirements, the open-source project "OFBiz" is selected, enhanced, and deployed into the software environment.

The final part of this thesis outlines the realization of a proof-of-concept prototype, which is carried out within the three-year European research and development project "FLoCI-EE". For it an exemplary, practical business scenario is modeled and implemented. The most important realization fruits and a profitability appraisement for the developed software system complete the thesis.

Acknowledgment

This dissertation came into being during my employment as a research assistant at the Institute for Flexible Automation which is now part of the new Automation Control Institute at the Faculty of Electrical Engineering and Information Technology, Vienna University of Technology. My research work deals with enterprise integration and collaboration including the underlying core technologies.

My sincere thanks are given to Em.o.Univ.Prof. Dipl.-Ing. Dr. Gerfried Zeichen for his steady encouragement of my work at the institute and the supervision of my dissertation.

Furthermore, I thank o.Univ.Prof. Dipl.-Ing. Dr. Alexander Weinmann for the review of my dissertation as second assessor and for his valuable suggestions and enhancements.

I would like to express my graditude to the members of the Automation Control Institute for the excellent cooperation. Special acknowledgment goes to my former superior Dipl.-Ing. Dr. Karl Fürst as well as to my colleagues Dipl.-Ing. Klaus Glanzer and Dipl.-Ing. Gerald Wippel for many stimulating discussions and fruitful teamwork.

Many affectionate thanks go to my family, in particular to my parents, for their unending support during my years of education. Finally and above all, I would like to thank my wife Susanne for her incessant encouragement.

Contents

1 Motivation, Aim of the Work **7**

1.1 Motivation 7

1.2 Aim of the Work 12

2 Holistic Integration of Complex Industrial Processes **16**

2.1 Evolvement of Mass Production 16

2.2 Basic Principles of Holistic Manufacturing 19

3 Definitions, Subsystems and Standards **24**

3.1 Business Systems supporting the Product Life Cycle 25

3.1.1 Product Data Management (PDM) 27

3.1.2 Enterprise Resource Planning (ERP) 29

3.1.3 Supply Chain Management (SCM) 31

3.1.4 Customer Relationship Management (CRM) 32

3.2 Theory of Enterprise Networking 34

3.2.1 Types of Business Collaboration 35

3.2.2 The Extended Enterprise Definition 37

3.2.3 Life Cycle of Extended Enterprises 39

3.3 Workflow Management System Basics 43

3.4 Approved Standards for Business Process Modeling 46

3.4.1 Unified Modeling Language (UML) 47

3.4.2 XML-based Standards describing Business Processes . . 49

3.4.2.1 XML-based Process Definition Language (XPDL) 49

3.4.2.2 Business Process Modeling Language (BPML) . 51

3.4.2.3 Business Process Execution Language for Web Services (BPEL4WS) 52

3.5 Concepts and Standards behind Web Services 53

3.5.1 Simple Object Access Protocol (SOAP) 56

3.5.2 Web Services Description Language (WSDL) 57

3.5.3 Universal Description, Discovery, and Integration (UDDI) 58

4 State of the Art **60**

4.1 Methodology for Enterprise Application Integration 61

4.1.1 Remote Procedure Call (RPC) 63

4.1.2 Message-Oriented Middleware (MOM) 64

4.1.3 Transaction Processing Monitor 64

4.1.4 Message Broker 65

4.1.5 Distributed Object 65

4.1.6 Application Server 68

4.2 PROFInet: Emerging Conception for Automation 69

4.3 Extended Enterprise Projects and Systems 70
4.3.1 US Project NIIIP 70
4.3.2 European Research Project PRODNET II 71
4.3.3 IMS Research Project GLOBEMEN 73
4.3.4 SAP R/3 System 76
4.3.5 e-Business Systems 79
4.3.5.1 mySAP.com 80
4.3.5.2 Siebel CRM Solutions 81
4.3.5.3 Microsoft Business Solutions 83
4.4 Classification of the Dissertation 84

5 New Approach enabling Extended Enterprise Collaboration 89
5.1 Extended Enterprise Requirements 90
5.1.1 Functionality 93
5.1.2 Integration 94
5.1.3 Usability 95
5.1.4 Security 97
5.1.5 Flexibility 98
5.2 Integration Concept for Knowledge-driven Industry 99
5.3 Fundamental Conception for Problem Solving 104

6 Technical Specification of the Software System 108
6.1 System Architecture 109
6.1.1 Client Tier 110

6.1.2 Role-based Presentation Tier 110
6.1.3 Business Process Automation Tier 112
6.1.4 Service Tier 113
6.2 The Key Component: Business Process Engine 115
6.2.1 Motivation for Open Source Engine 115
6.2.2 Workflow Engine as the Principal Component 117
6.2.2.1 Selection of appositely Workflow Engine 117
6.2.2.2 Characteristics of selected Workflow Engine . . 120
6.2.3 Deployment of the Business Process Engine 121

7 Proof-of-Concept Prototype Realization 126

7.1 Exemplary Business Case 127
7.1.1 Goal and Background Information 127
7.1.2 Scenario Description 128
7.2 Modeling the Business Case 129
7.2.1 Definition of Required Roles 129
7.2.2 Specifying the Prototype Functionality 130
7.2.3 Modeling the main Business Process 137
7.2.3.1 Descriptive Model 137
7.2.3.2 Utilizable for Business Process Engine 142
7.3 Implementation Environment 147
7.3.1 User Interface Framework 148
7.3.2 Enterprise JavaBeans Container 151

7.4 Realization Fruits 152
7.4.1 Performance Measurement of Web Services 152
7.4.2 User Acceptance of Implemented Prototype 154
7.4.3 Experiences with Open Source Software 157
7.5 Profitability Appraisement for Developed System 160
7.5.1 Development Costs 160
7.5.2 Unique Selling Proposition 160
7.5.3 Target Market 162
7.5.4 Competition Situation 163
7.5.5 Realization Risks 163
7.5.6 Introduction Costs 164
7.5.6.1 Primary Acquire Costs 164
7.5.6.2 Adjustment Costs 165
7.5.6.3 Operating Costs 166
7.5.7 Benefits 166
7.5.7.1 Direct Benefits 166
7.5.7.2 Indirect Benefits 166
7.5.8 Break-Even Analysis 167

8 Summary and Outlook **169**

A List of Abbreviations **194**

B Glossary **198**

C List of Publications **203**

D Curriculum Vitae **208**

Chapter 1

Motivation, Aim of the Work

1.1 Motivation

The more and more sophisticated information and communication technologies enable - and sometimes even force - an all-embracing integration of all business partners involved in complex industrial business processes. To solve those extensive problems, it is essential to develop a multidisciplinary, holistic approach including information technology, automation, engineering, logistics, product design, etc. The intention of this dissertation is to bridge the existing gaps between these fields, which have been open by focusing only on the traditional domains. Especially the study of electrical engineering transfers a profound knowledge to deal with such complex industrial problems enforcing holistic solutions.

Throughout the twentieth century, the traditional starting point for strategic business thinking had been the individual enterprise. Such enterprises have tried to accomplish the whole product life cycle by oneself. But nowadays in highly competitive and rapidly changing global economy environment, busi-

ness organizations need to collaborate to achieve common business goals, and to support the product life cycle in a more flexible and effective way than ever before. By the need of collaboration between either intra-organizational departments or independent enterprises, the presently appeared communication disruptions within existing quality control loops, which connect the various stages of the product life cycle, have to be prevented (Figure 1.1).

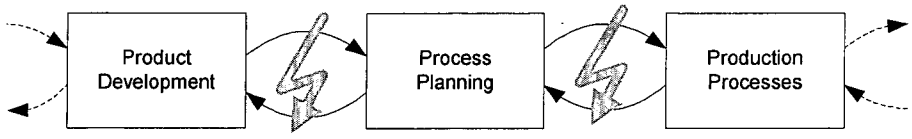


Figure 1.1: Communication Disruptions between Stages of Product Life Cycle

The solution on this problem is the automation of complex systems. A very demonstrative example in this context is the flight control system "voice communication system" developed by the Austrian company Frequentis to enable, among other things, the undisturbed, standardized communication between pilot and tower.

In the field of enterprise collaboration, nowadays a new era starts where systems of suppliers, service providers, manufacturing companies, and customers use the Internet as the basis for business communication and automation [KR99]. Plug-and-play collaboration between independent enterprises with different core competencies will be key for staying competitive in such turbulent markets.

Within every business, dynamic activity generally takes place at a number of different management levels. From an information technology point of view, these levels are characterized by their data volumes and response times. Different information requirements exist at field level (where operational, visible processes are executed) and management level (where strategic - mostly invis-

ible - processes are executed). These requirements are specified numerically in Table 1.1. [ZF00]

Levels	Data Transmission		
	Volume	Response Time	Frequency
Top Management Level	GByte	Hours	Day
Manufacturing Level	MByte	Minutes	Hours
Operational Level	Byte	Seconds	Minutes
Shopfloor Control Level	Bit	100 ms	Seconds
Field Level	Bit	Millisec.	Millisec.

Table 1.1: Data Transmission Structure [ZF00]

In spite of increasing complexity, each employee of co-operating enterprises has to accelerate the process flows and the decision processes. The necessary information demand (Table 1.1) for that purpose exceeds massive the information processing capacity of a human. Therefore, a sophisticated support through information systems is indispensable to reduce the lack of information (Figure 1.2). The importance of information technology shows the facts that the average enterprise spends more than 4.2% of revenues annually on IT, and that those investments account for more than 50% of the total capital budget [WSB02].

The automated data processing (point "B" in Figure 1.2) was the first step to reduce the gap between information demand and human data processing capacity. Examples for that are enterprise business systems like enterprise resource planning (ERP) for automated data processing inside the company, and electronic data interchange (EDI) for the electronic data exchange between co-operating companies. Especially the traditional EDI, using different standardized protocols such as EDIFACT, ANSI X.12, or ODETTE, is only

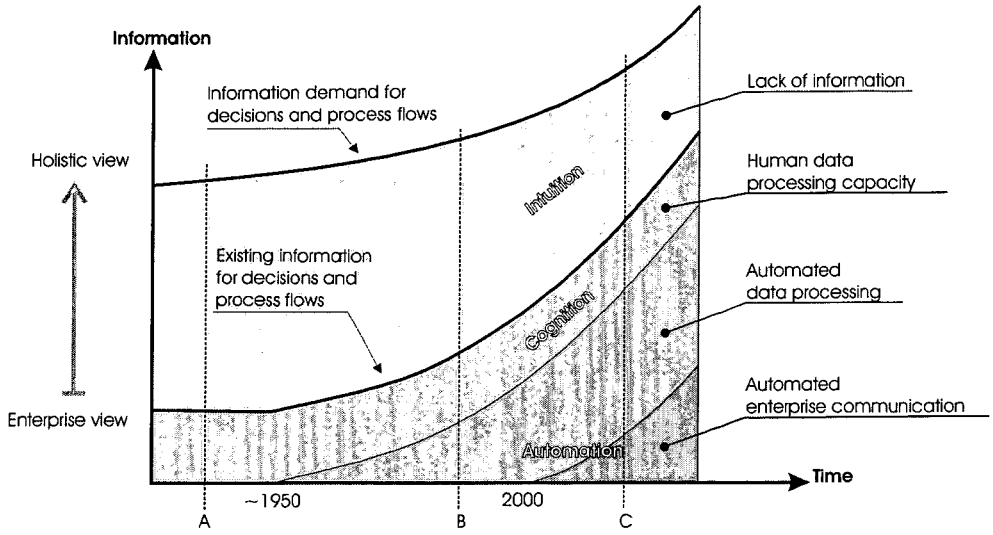


Figure 1.2: Information demand for decisions and process flows [FH02][Häb03]

practical to control the material flow along the supply chain - and not for flexible business-to-business integration, because there are still problems integrating data and services of different business information systems. The main reason for that is the nonexistence of consistently standards to access these systems.

The example of the company Oracle is very illustrative. *"In the year 1997, Oracle had 97 e-mail servers running seven incompatible programs. Each country had its own enterprise resource planning system and its own Web site in the local language and adapted to the local manager´s taste. And to find out how many people Oracle employed worldwide on any given day, someone hat to scout through 60 differently formatted databases and consolidate the number."* [GG02] As Oracle CEO Larry Ellison explained, *"If you want to buy a car, would you get an engine from BMW, a chassis from Jaguar, windshield wipers from Ford? No, of course not. Right now with the software out there, you need a glue gun."* [GG02]

By now Oracle has solved this problems internally through centralize the local systems to global systems. But this is not possible in the global market where different global and local players use different business systems. There is still the comparable "software hell" (see also Figure 4.1) explained in the above example.

To solve these problems, the next evolution step - automated information procurement respectively the automated communication and integration between information systems - is essential. The aim is the totally automation and integration of industrial business processes that transcend enterprise boundaries (point "C" in Figure 1.2).

After early fears of company managers that the Internet would disrupt the business models of many companies (for example, by enabling customers to switch suppliers easily) have subsided, nowadays many of them see the Internet as a significant opportunity [Joh02]. Furthermore, another barrier can be overbeared: the technological feasibility. The most important change is the Web and the associated information technology capabilities. Information sharing has always been at the heart of integration, but now technology allows organizations to respond to integration needs in ways that were unavailable even five years ago [GG02]. Some indications for that are:

- nearly complete distribution of the Internet as an optimal, cost-effective infrastructure for enterprise-spanning communication and integration;
- more and more existing and sophisticated standard technologies like XML (eXtensible Markup Language, [GP99]) and Web services (see chapter 3.5) to enable a common "language" to describe data, information, and services;
- extensive offering of Web-based information inside the enterprise using

the Intranet.

Therefore, this integration step using the Internet and Web-based technologies is now technologically possible, and for the global economy very beneficial. This dissertation provides a holistic system architecture therewith independent enterprises can integrate and automate their enterprise-internal and -spanning industrial business processes in an open, flexible, and cost-effective way using the latest Internet technologies. For most of enterprises this will be the key for their economic survival, because the detected proposition by Charles Darwin is also valid for companies:

"It is not the strongest of the species that survive, nor the most intelligent, but the one most responsive to change." [Pyk01]

1.2 Aim of the Work

The aim of the work is to develop a holistic, flexible, and open system to realize the vision of the totally integrated enterprise. This approach has to eliminate the problems described in chapter 1.1 and make the automated communication and integration between different enterprises possible. The characteristic of this new system is high flexibility combined with high time-to-market (Figure 1.3), compared to bought standard software such as SAP and self-built in-house software solutions.

To cast the innovative approach in software and to position system against the competitors, the developed IT architecture has to satisfy the following requirements:

- enabling the horizontal integration of enterprise-spanning industrial business processes inside the extended enterprise where independent enter-

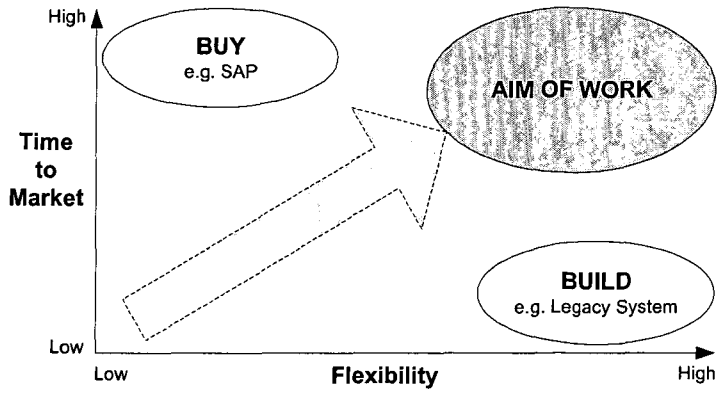


Figure 1.3: Positioning the Emerging System

prises can join forces to be competitive in the global market;

- enabling the vertical integration inside one enterprise to solve the integration problems with different, incompatible business software systems ("Oracle example", see chapter 1.1) in the face of facilitating the collaboration inside the extended enterprise;
- providing one "entry point" for the enterprise roles (note: enterprise users take in one or more enterprise roles e.g. project manager, software developer, etc.) to support them with all needed business processes and information needed for their daily work;
- using open, international established standards and technologies like XML to avoid a software vendor specific solution;
- using sophisticated open source solutions as often as possible to get a cost-effective software product which is also appositely for small and medium sized enterprises;
- realizing the software components with J2EE (Java 2 Enterprise Edition) to be independent from the different used operation systems [Sel93] in

the heterogeneous environment inside extended enterprises.

The most important part of the IT architecture is the business process engine to process the complex, networked industrial business processes inside the extended enterprise. On the one hand, the business process engines have to provide all role-specific business processes for the different enterprise roles to support their daily work. On the other hand, the business process engines have to enable the collaboration and integration inside the extended enterprise through using either

- services which are available inside the own enterprise, or
- services which are provided by enterprises inside the extended enterprise, or
- business processes which are available inside the extended enterprise.

Therefore, the business process engine is the key component for the realization of the totally integrated enterprise.

A further aim of this dissertation is the realization of a proof-of-concept prototype. This prototype implements a typical, industrial scenario inside an extended enterprise to proof the usability of the developed approach and IT architecture.

To give the aim of this work more systematics, the following overview is given (according to [Pun95]):

Purpose: This work is a new scientific contribution in the area of extended enterprises in respect of enterprise-spanning business process integration and automation. Additionally to this horizontal integration aspect, the vertical integration difficulties inside one enterprise are discussed. The key to solve

these integration problems is a business process engine processing all complex, networked, industrial business processes inside such an extended enterprise. The developed IT architecture makes the flexible creation of extended enterprises and - above all - the intensive collaboration of independent enterprises possible.

Customer: This scientific work should be a valuable contribution to the international scientific community and a foundation for further research work in the area of extended enterprises. Additionally, the developed IT architecture and business process engine can be used by software vendors to implement software components for an flexible, cost-effective software system enabling extended enterprises.

Newness: Due to the holistic, scientific approach, the complex problems of enterprise collaboration are systematizes. At present there are only some isolated problems are solved like the automated data exchange between enterprises, but not the totally integration between them. In this work not only the horizontal integration between independent enterprises are solved, but also the vertical integration difficulties between different business information systems inside one enterprise. For this purpose the business process engine for extended enterprises has been developed and implemented within a proof-of-concept prototype.

Chapter 2

Holistic Integration of Complex Industrial Processes

The aim of this chapter is the description of the evolution from the strong reductionism in F.W. Taylor's theory to holistic, open systems where the market, the demand for cooperation, and not the partial optimization play the leading parts. Furthermore, this development effects the replacement of the strict control thoughts to servo loops where the achieved results affect the further decisions to enable robust fruits.

2.1 Evolvement of Mass Production

Frederik W. Taylor (1856 - 1915) introduced his book, the *Principles of Scientific Management* [Tay98] in 1911 that has changed the workplace forever. These principles became known as "Taylorism". For nearly one century this theory made possible a tremendous growth in living standards and wealth [Zei95a]. By splitting the whole process of product manufacturing into the smallest possible single steps on the one hand, and through sophisticated sys-

tem planning on the other, Taylor's concept enabled even untrained people to contribute to a complex task [ZF00][Pun95]. Additionally, relatively low-skilled workers could produce a high volume of products in comparison to single piece production by highly skilled craftsmen [ZH98].

Taylorism also began to change how organizations functioned. Before this time organizations were usually setup in homes or in formal businesses where the workspaces were open. There were no barriers for communication and ideas could flow freely among employees. Taylorism abruptly changed this feature of organizations: [KF98]

- Hierarchical leadership
- Split locations for manufacturing and office work
- Work became specialized with divisional labor
- Product/outcome focused - not customer focused
- Demand exceeded supply
- Manufacturing and industrial companies were the main company types

The strict application of the Taylorism's principles led to the creation of huge number of interfaces between specialists, monotonous work, and a very restricted information horizon for the employees.

Henry Ford (1863 - 1947) applied Taylor's principles to making automobiles, and founded one of the largest industries in the United States. The first moving assembly lines, which took the automobile frame from one worker to another, put into operation at Ford's plant in Michigan in 1914. The assembly line increased labor productivity tenfold and permitting stunning price cuts in Ford cars [Wil03]: from USD 780 in 1910 to USD 360 in 1914. This low

price made the Ford Model T widely available to most American families. In 1914 the Ford Motor Company with 13,000 employees produced 267,720 cars; the other 299 American auto companies with 66,350 workers produced only 286,770 cars.

The increased consumption due to lower prices through standardized mass production embosses the term "Fordism". Americans and Europeans both took this idea and started to mass produce everything from clothing to furniture to machines. Since mass producing an object usually lowers it's cost, everything became more available to the masses.

As a consequence of that strict reductionism, the manufacturing flexibility was lost because the manufacturing plant was specialized for the Model T. It has never been proven that Henry Ford ever said, *"You can paint it any color, so long as it's black"* [Hen03], but exactly this weakness was the opportunity for the competitors to break the market dominance of Ford - through offering different colors, sizes, or models [GS02].

Beginning of the 1970s, the standardized mass production has been replaced more and more through a quality-based production, which can be characterized through customer orientation and lower piece of parts with higher quality [Pre03]. One representative for this evolution is the so-called lean production developed by Toyota ("Toyotism"). The main characteristics are: avoidance of overproduction, customer integration into the product development, team work, and outsourcing to the suppliers. Thereby, new concepts like just-in-time have been established. Just-in-time (JIT) is a logistics concept for the reduction of material, semi-finished and finished goods stocks in and between the individual operations involved in the product creation process. A distinction is made between JIT delivery, JIT distribution, and JIT production [ZF00].

The increasing complexity of industrial processes during the last years has

led to competitive aims of different functional units inside an industrial enterprise. Due to the optimization of partial processes, the approach of lean production cause more and more target conflicts. A typical example is the conflict between quality, time, and costs [Esc96]. Solutions to solve that conflicts have been developed by [PT02] finding out an aggregate value for the whole utility value, and by [Tra02] finding out new methods for optimisation of an integrated production plant.

Through the separate contemplation of partial processes it will be not possible to optimize the whole process. Therefore, during the last ten years a new approach has been developed to establish a holistic view of industrial processes.

2.2 Basic Principles of Holistic Manufacturing

Today, the problems lie not in the performance of individual tasks and activities, the units of work, but in the processes, how the units fit together into a whole [Ham96]. Therefore, in the 1990s new management concepts view industrial enterprises not in terms of functions, divisions, or products, but of department-spanning key processes. According to DIN 66201,

"a process is the totality of activities in a system which are influencing each other and by which material, energy, or information is transformed, transported, or stored."[ZF00]

Another definition for the term "process" can be found in [Dav92]:

"a process is simply a structured, measured set of activities designed to produce a specified output for a particular customer or market. It implies a strong emphasis on how work is done within an organization, in contrast to a product focus's emphasis on what. A process is thus a specific ordering of

work activities across time and place, with a beginning, an end, and clearly identified inputs and outputs: a structure for action."

According [ZF00], *"an industrial enterprise is a company involved in the design of a product as well as in the production and sales of this product"*. Therefore, *"industrial processes can be said to be equivalent to business processes, however, the term business process also applies to trading firms and commercial enterprises"*. Thereby, the terms "business process" and "industrial process" are frequently used in the same context.

Most companies, even large and complex ones, can be broken down into fewer than 20 major processes. IBM, for example, has identified 18 processes. Key business processes in manufacturing firms are: product development, customer acquisition, customer requirements identification, manufacturing, integrated logistics, order management, post-sales service, performance monitoring, information management, asset management, human resource management, planning and resource allocation. [Dav92]

Thus, the essential difference to Taylorism is that not separated functions inside an enterprise are considered, but processes cross existing organizational boundaries. While Taylor's theory was based on reductionism, orientation on single tasks, a linear causality, different planning and operating, and single working, the holistic approach is interested to: [Zei95a]

- see the whole process, including shop floor, planning and business systems
- be orientated on customer and therefore be flexible
- non-linear controlled procedures
- behave like bionic system
- look for job enlargement and team work

With the holistic approach, the unneeded walls between the Tayloristic divisions of specialized work are dragged down, and information technology is applied to improve the procedures in industrial communications.

Because a process perspective implies a horizontal view of the enterprise that cuts across the organization, major processes such as product development include activities that draw on multiple functional skills (holistic product development [Rau01]). New product designs are generated by research and development, tested for market acceptance by marketing, and evaluated for manufacturability by engineering or manufacturing. [Lan98] has developed a holistic approach to plan and optimize the whole product development process. The creative work of the product developer in such a process-oriented manner [Lei02], from the product idea up to implementation, is called "holistic engineering". *"Holistic engineering enables the inventor or developer of a new product to clearly and efficiently define the structure of and accept responsibility for the production and marketing subprocesses that follow product development"* [ZF00]. Furthermore, holistic engineering is implemented in a feedback loop which, on the one hand, transfers responsibility for the subsequent production functions to the product developer and, on the other hand, enables him to receive a direct response regarding acceptance of his development in the form of feedback.

Figure 2.1 shows the interaction between multiple disciplines and the market, where the utility value actual is the total added value within the "Big Manufacturing" processes, and the utility value targeted is the expected customer value of the product. This model embeds the manufacturing processes into a control loop where the market constitutes the plant. In holistic engineering, the controlling approach is not only restricted to qualitative models, but also to quantitative models where the research work and results by [Wei94] are the foundation.

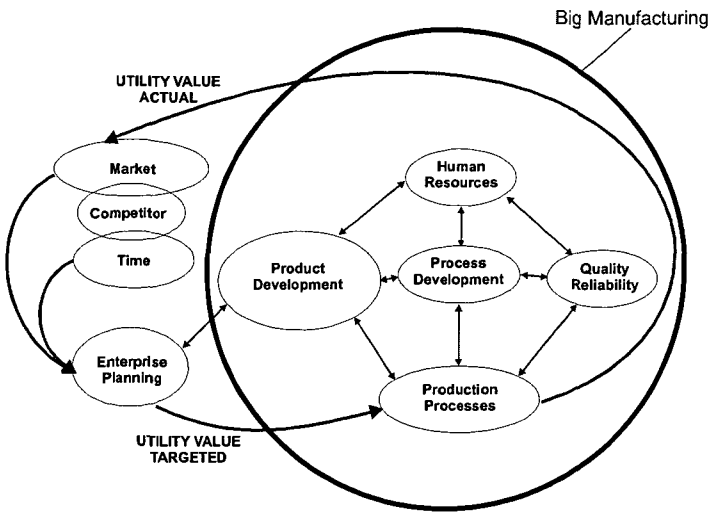


Figure 2.1: "Holistic engineering" as the interaction between multiple specialist disciplines from the engineering sciences and economics [ZF00]

The term "Big Manufacturing" was introduced by the Massachusetts Institute of Technology (MIT) for integrating all necessary functions for industrial manufacture of goods and services [Zei95b]. Especially in turbulent market situation as in times of mega mergers, the manufacturing companies have to develop completely new architectures. Even small and medium-sized enterprises are developing new strategies like virtual cooperations between business partners and vertical integration inside the enterprise. Both holistic engineering and holistic manufacturing support this new trend by considering all needed functions of an industrial company. Due to the holistic view, a exact definition of the not international standardized term "manufacturing" is necessary. *"Manufacturing is defined as the sum of all subfunctions that are necessary for the manufacture of a product:*

- *product development and design (realised in the product specification list and relevant drawings),*

- *process development (defining the parts production and assembly procedures),*
- *operational tasks such as parts production, assembly, filling, quality assurance and packaging for the whole product,*
- *order processing and communication with sales and suppliers."* [ZF00]

The trail to holistic manufacturing is now opened because of many needs:

- Market driven holistic manufacturing and engineering
- Integration between the Tayloristic departments to support industrial processes
- Support of cooperation and collaboration between business partners
- Integration of multi-vendor hardware and software components

Enablers for the holistic approach are the firm organization, employee acceptance, and the information technology with the objective to provide the *right information, at the right place, at the right time.*

Chapter 3

Definitions, Subsystems and Standards

In this chapter all definitions, subsystems, and standards, that are used in this dissertation, are discussed in detail.

At the beginning the main established information systems supporting the product life cycle will be described. In this enterprise-centric view, computer technology has supported computing and data storage inside the firm.

The next evolution step was the development of strategies for cross-enterprise collaboration. Many organizations and research communities have evolved diverse approaches and concepts for enterprise networks like virtual enterprise or extended enterprise. The focus in such networks lies in the horizontal integration between independent enterprises.

To enable the automation and integration of business processes, two things are necessary: on the one hand the business processes have to be modeled in a standard way, and on the other hand a business process engine has to step through the business process activity by activity. Therefore, in this chap-

ter the standardized business process modeling languages and the workflow management systems to carry out the business processes are discussed.

For the realization of the new concept of enterprise-spanning automation and integration over the Internet, the latest XML-based standard named Web services is essential to bridge the existing gaps between information systems.

3.1 Business Systems supporting the Product Life Cycle

Therewith a company can achieve one of the most important success factor *the right product at the right price at the right time*, the product supporting processes must be automated for the entire product life cycle. The need for business process automation has led to the development of several information systems. Today, many different information systems are available on the market that are specialized in separate areas of application, with frequent overlaps in system functionality. The most important systems are shown in Figure 3.1.

Although Figure 3.1 represents the systems as bars extending across several stages of the product life cycle, this does not necessarily mean that these systems are used in a number of different phases, but rather that certain functionalities are required in the various stages of the product life cycle. To explain this more clearly, let's take document management systems as an example. Documents are generated throughout the entire product life cycle. However, document management functionality is provided at different stages by different real systems [ZF00]. For example, during the product development the functionality is provided by the product data management system, whereas the needed electronic product catalogue during sales is supported by the sales supporting system. The following four information systems supports the main

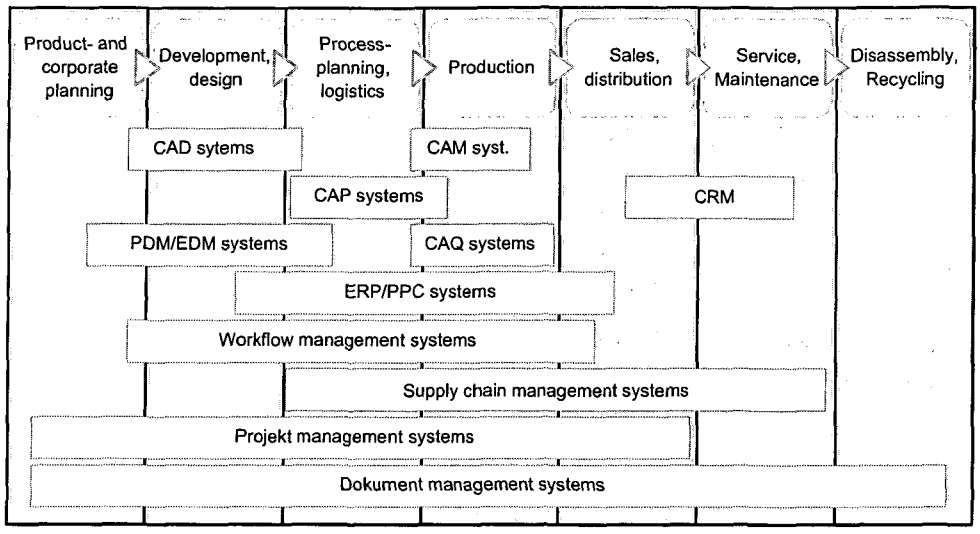


Figure 3.1: Information Systems in the Product Life Cycle [ZF00]

phases of the product life cycle:

- product data management (PDM) systems supporting product development,
- enterprise resource planning (ERP) systems supporting process- and production planning,
- supply chain management systems (SCM) supporting external logistics, and
- customer relationship management (CRM) supporting one-to-one customer communication.

3.1.1 Product Data Management (PDM)

Product data management (PDM) helps engineers and designers to manage the product development process and to access the data associated with it. PDM systems keep track of the vast amounts of data, and information, required to design, manufacture, and then support and maintain products [Sma03]. PDM systems are also referred to as EDM (engineering data management) systems. [ZF00] defines PDM systems as follows:

"The role of product data management systems is to manage all of the product data in the product creation process. They form a higher level platform for access to the product data and corresponding documents (drawings, parts lists, NC programs, etc.) as well as for backup, control, distribution, and archiving."

The aim of PDM systems is the communication with the different CAx-Systems of an enterprise such as computer aided design (CAD), computer aided manufacturing (CAM), or computer aided planning (CAP) to generate the so-called virtual product model [Kla00]. PDM systems manage not only direct data like drawings or documents, but also metadata. This metadata is the data about data, such as time of last change, version number, etc. The basic functions of PDM systems are as follows (Figure 3.2): [ZF00][Bra00]

- Product data management: The system manages the data and files in a database and/or in a protected area of the file system (vault). Users can therefore only access this data via the PDM system to create/modify objects and relationships, search for objects/data, and display the product structure.
- Workflow management functions: Workflows, which are used to control release processes, can be created for the objects in the PDM system.

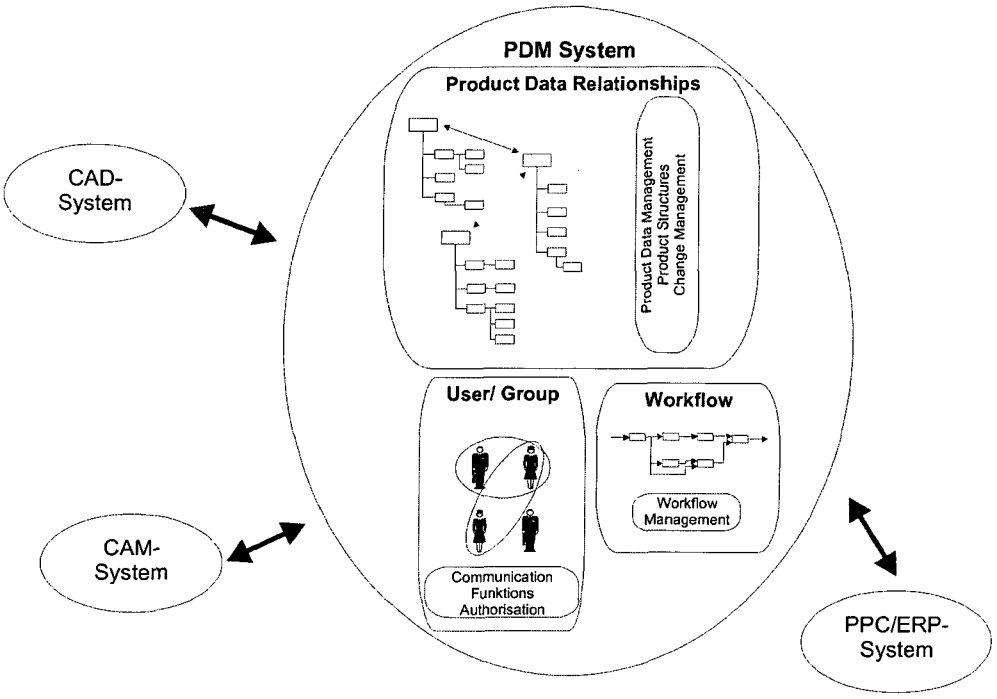


Figure 3.2: Product data management system [ZF00]

Activation of the workflow can be manual, time-controlled, or event-driven. A change in status can trigger actions like automatic notification.

- User and access administration: In order to control access to the data/-documents managed by the system, users are created in the system. These users can be assigned to roles and groups. The permissions for accessing data objects are managed via these users, groups, and roles.

- Change management: The system automatically generates a change history for documents and objects, which can traced back if necessary. The systems support version management.

- Communication functions: Users can send messages to one another using the integrated messaging functions. The communication functions are

also used e.g. for workflow-triggered notification.

- Additional functions: Some PDM systems also offer project management functions, archiving functions, visualization functions, or the option of integrating external viewers.

Today, the limitation on engineering of traditional PDM systems isn't still advisable because companies don't want to invest effort, time, and money in interfaces between development and construction on the one hand, and production, sales, and service on the other hand [EGH+01]. Instead of they want to expand the accessibility of product data beyond engineering to also include marketing and even customers [Int03]. This new requirement is supported by product lifecycle management (PLM) systems.

3.1.2 Enterprise Resource Planning (ERP)

Enterprise resource planning (ERP) is a structured approach to optimizing a company's internal value chain. The software, if fully installed across an entire enterprise, connects the components of the enterprise through a logical transmission and sharing of common data with an integrated ERP [NHH+00]. [ZF00] defines ERP systems as follows:

"ERP systems support and automate the organizational planning, control and monitoring of production and assembly sequences, from order receipt (by sales), through capacity scheduling, production scheduling, production data acquisition collection, right up to control of dispatch."

The task of ERP systems are to organize, codify, and standardize an enterprise's business processes and data. The software transforms transactional data into useful information and collates the data so that it can be analyzed. In this way, all of the collected transactional data becomes information that

companies can use to support business decisions [NHH+00].

ERP systems can include a wide range of functionality, using components that are often referred to as "modules". Examples for such modules are financial accounting, human resources, production planning, and sales [O'L00].

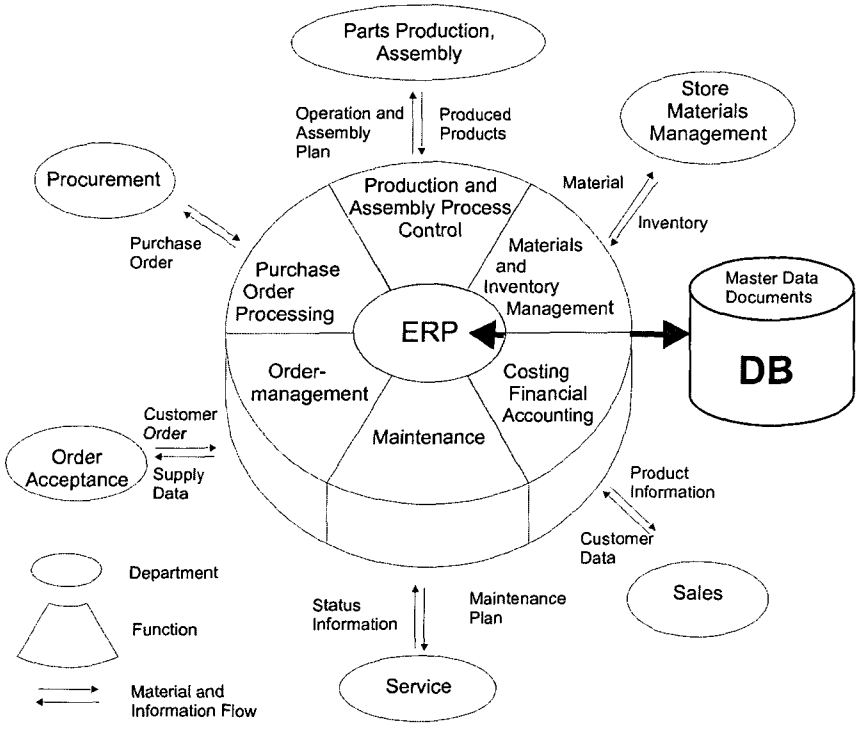


Figure 3.3: ERP system in an industrial enterprise [ZF00]

The most important areas of application of ERP systems are (Figure 3.3): [ZF00][Bra00]

- Production planning: sales, marketing and production/assembly rough-cut planning; material requirements plans; resource planning.
- Process planning: creation of work and assembly schedules.
- Materials and inventory management: inventory on stock; supplier order

processing.

- Production and assembly control.
- Costing, financial accounting, and sales.

3.1.3 Supply Chain Management (SCM)

In a typical supply chain, raw materials are procured, items are produced at one or more factories, shipped to warehouses for intermediate storage, and then shipped to retailers or customers. Consequently, to reduce cost and improve service levels, effective supply chain strategies must take into account the interactions at the various levels in the supply chain. Supply chain management is a set of approaches utilized to efficiently integrate suppliers, manufacturers, warehouses, and stores, so that merchandise is produced and distributed at the right quantities, to the right locations, and at the right time, in order to minimize systemwide costs while satisfying service level requirements [SLKSL00]. [ZF00] defines SCM systems as follows:

"SCM systems are IT tools that support effective management of the supply chain."

Figure 3.4 shows an element of a supply chain. Several suppliers deliver to an industrial enterprise via logistics service providers. The industrial enterprise basically consists of procurement, warehousing, production, assembly, and distribution. The products are, in turn, distributed to the customer via logistics service providers. These customers may also be industrial enterprises or even end customers. The joining together of these elements gives us the supply chain.

SCM systems offer the following functionalities: [ZF00][Bra00]

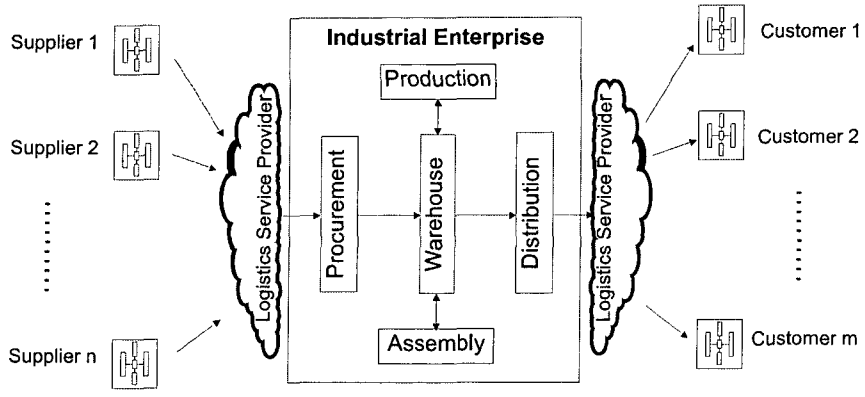


Figure 3.4: Element of a supply chain [ZF00]

- Supply chain monitoring: Visualization and monitoring of supply chains or of individual supply chain elements.
- Procurement management: Technical support for procurement transactions including parts analysis, make-or-buy decision support, or assistance for supplier selection.
- Third-party management: Technical support for outsourcing transactions including integration of third-party manufacturers, logistics service providers, or information service providers.
- Storage and transport management: Technical support for distribution operations including transport tracking or quality monitoring.

3.1.4 Customer Relationship Management (CRM)

Customer relationship management (CRM) is not only a software product or technology, it is a process that manages interactions between a company and its customers. For the Gartner Group, CRM is a business strategy designed to optimize profitability, revenue, and customer satisfaction. Therefore, the en-

terprises must put the customer in the center and integrate all processes around a single view of the customer. This is the fundamental concept of CRM, and one that makes it so difficult to do. In order to have a complete CRM system, a company must coordinate all customer contact points, which means capturing all company/customer interactions across all channels [AKY01]. [ZF00] defines CRM systems as follows:

"CRM systems are information, consulting, quotation and ordering systems for sales and marketing (Figure 3.5)."

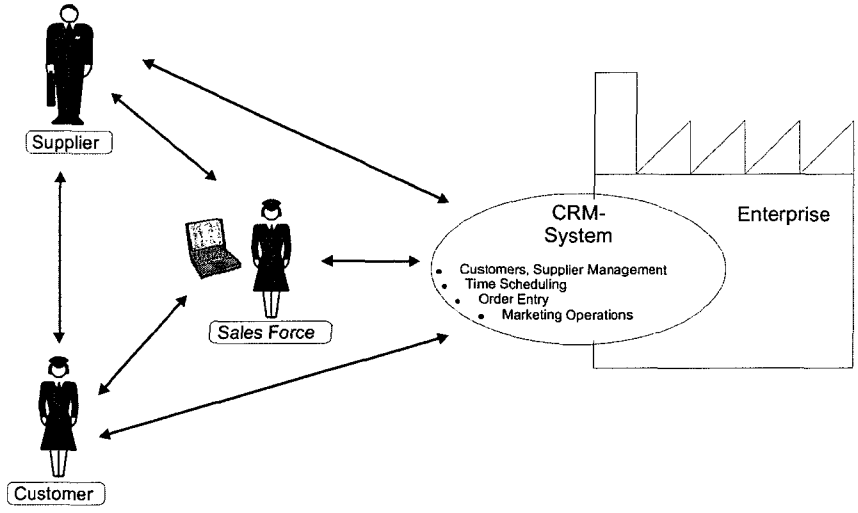


Figure 3.5: CRM system [ZF00]

Customer relationship management allows companies to be customer-focused, customer-driven, and customer-centric. Customer-driven competition is often called one-to-one marketing, a form of marketing that was prohibitively expensive to the traditional marketer just a few years ago. Today, one-to-one marketing is made possible by three important capabilities that information technology now provides: [PR97]

- Customer tracking: Computer databases can help businesses remember

and keep track of numerous complex, individual interactions with their customers. A business can now focus on one single customer from among the millions in its database.

- Interactive dialogue: The computer has also made an increasing array of interactive communications tools available.
- Mass customization: Applied to the assembly line and the logistical system, information technology now makes it possible for businesses to deliver mass customized products and services with enough variety and customization that nearly everyone finds exactly what they want [II93].

The main components of CRM systems are: [AKY01][Bra00]

- Sales management functionality
- Customer service and support functionality
- Marketing functionality
- Time management functionality

3.2 Theory of Enterprise Networking

Information technology (IT) enables and shapes the new economy the same way as steam, gas, and electricity have changed the agricultural economy into the industrial economy a hundred years ago. Business networking is not the only, but the most important feature of enterprises in the information age. Now, IT is leading to fundamentally new mechanisms in the collaboration between organizational units. In our time, IT is pushing the physical disintegration of markets and enterprises to its global limits and thus enables maximal

specialization of the enterprises. Business networking in the new economy can be seen as the coordination of processes within and across companies. More precisely, [ÖFA00] defines *"business networking as the management of IT enabled relationships between internal and external business partners"*.

3.2.1 Types of Business Collaboration

There are different approaches, different names that basically cover the same idea of networked enterprises: a flexible network of cooperating autonomous manufacturing units for the whole range of manufacturing activities from order booking through design, production and marketing to form an autonomous and decentralized manufacturing organization [KM98].

[CMA99b] defined the following terms (Figure 3.6):

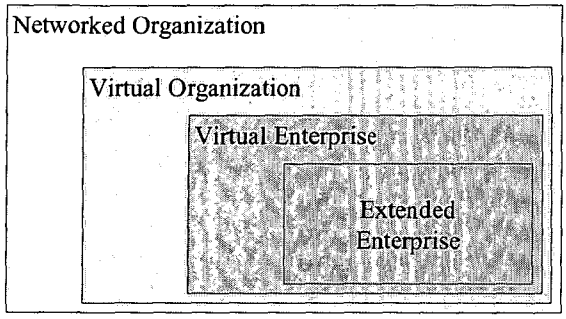


Figure 3.6: Different Types of Enterprise Networking [CMA99b]

- Virtual Enterprise (VE): A virtual enterprise is a temporary alliance of enterprises that come together to share skills or core competencies and resources in order to better respond to business opportunities, and whose cooperation is supported by computer networks.
- Extended Enterprise (EE): The concept of extended enterprise, the clos-

est "rival" term to virtual enterprise, is better applied to an organization in which a dominant enterprise "extends" its boundaries to all or some of its suppliers, whilst the VE can be seen as a more general concept including other types of organizations, namely a more democratic structure in which the cooperation is peer to peer. In this sense, an extended enterprise can be seen as a particular case of virtual enterprises.

- Virtual Organization: This is a concept similar to a virtual enterprise, comprising a network of organizations that share resources and skills to achieve its mission/goal, but not limited to an alliance of enterprises. An example of virtual organization could be a virtual municipality, associating via a computer network, all the organizations of a municipality (e.g. city hall, municipal water distribution services, internal revenue services, public leisure facilities, cadaster services, etc.). A virtual enterprise is, therefore, a particular case of virtual organization.

- Networked Organization: This is perhaps the most general term referring to any group of organizations inter-linked by a computer network, but without necessarily sharing skills or resources, or having a common goal. Typically, networked organizations correspond to a very loose type of organization.

- Supply Chain Management: This term refers to the policies and supporting mechanisms to manage the flow of materials in the value chain, from raw materials procurement to sales, including production, distribution and delivery, involving suppliers, manufacturers, distributors, wholesalers, retailers and customers. SCM is supported by a bi-directional flow of information amongst the supply chain participants, This concept is traditionally applied to organizations that are relatively stable, i.e. where the core partners remain the same for a large period of time, however

more dynamic supply chains are becoming current. The focus is on the logistics of the material/product flows and related business information.

- Cluster of enterprises: A group or pool of enterprises that have the potential, and the will, to cooperate and therefore may become the partners in a VE. These enterprises are normally "registered" in a directory, where their core competencies are "declared". Based on this information, a VE initiator/creator can select partners when a new business opportunity is detected.

[GNP95], [Gor99], [BMMF97], and [JBB94] have published further definitions for the different kinds of enterprise networking. Whereas the presented enterprise networking approaches are more focused on organizational issues, the focus of e-business and related approaches (e.g. business-to-business (B2B) or e-commerce) lies on technology such as the Internet enabling collaboration between companies.

3.2.2 The Extended Enterprise Definition

The multitude approaches and definitions for enterprise networking presented in Chapter 3.2.1 makes clear that there is the need for an exact definition. In the scope of this dissertation, the term "extended enterprise" is defined according to [Für02] as follows: *Extended Enterprises (EE) are*

1. *temporary or permanent networks of*
2. *independent enterprises*
3. *including a coordinator*
4. *cooperating with the aim to*

5. *design, manufacture, and sell a product or service*
6. *in a project-oriented way*
7. *independent of enterprise borderlines,*
8. *automated inter-enterprise communication through the usage of information technology, and*
9. *communicating via Internet-related technologies like XML-based Web Services.*

ad (1) Extended enterprises can be set up for one single product or service, or exist as frameworks for long-term business relationships. If you take a look e.g. at the automotive industry, you can find long-term relationships between a big car-manufacturer and its suppliers.

ad (2) The enterprises are independent from each other. Each enterprise contributes the specific core know-how to the extended enterprise.

ad (3) In the case of a very strong extended enterprise coordinator who dominates the whole extended enterprise, the coordinator really "extends" its business in order to integrate business processes with other enterprises more effectively. Other extended enterprises with a more balanced power distribution between the coordinator and the other partners are also included in this definition. The coordinator has to provide the services needed to run the extended enterprise.

ad (4) The enterprises can cooperate in one market area and be competitors in an other.

ad (5) Not only specific parts (like product development, production, etc.) but the whole product's life cycle is considered in this definition of extended enterprises. For a short explanation of this topic see Chapter 3.1.

ad (6) The term "project" refers to one or several business transactions which are conducted in the scope of an extended enterprise (see Chapter 3.2.3 for details).

ad (7) The non-existence of individual enterprise borderlines guarantees highly effective and automated communication between the enterprises.

ad (8) The use of information technologies to support and automate inter-enterprise and extended enterprise business processes is key to efficiency.

ad (9) Internet technology is especially important to automate enterprise-spanning business processes (see also Chapter 3.4.2).

3.2.3 Life Cycle of Extended Enterprises

The life cycle of networked enterprises defines how they are initiated, created, administrated, operated, and dissolved. The formation of such networks can be compared with the steps needed to build a snowman:

1. find the right snow/condition
2. build a small strong cluster
3. roll this one to make it larger
4. put different balls together to get a bigger snowman

[CMA97] has defined a very general life cycle for virtual enterprises (VE) (Figure 3.7) including:

- Creation: This is the initial phase when the VE is created/configured including the main functionalities: partner selection, contract negotiation, definition of access rights, etc.

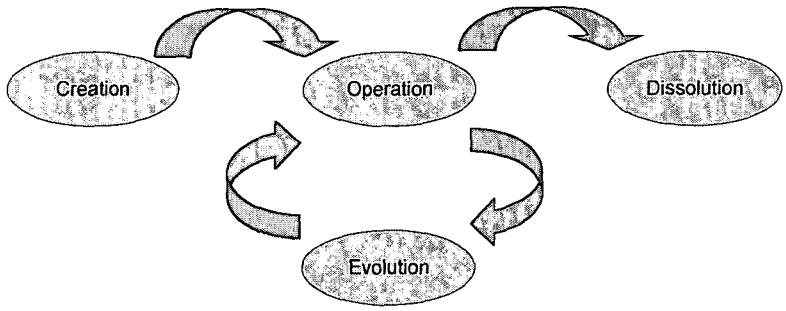


Figure 3.7: General Life cycle of Virtual Enterprises [CMA97]

- Operation: This is the phase when the VE is performing its business processes in order to achieve its common goals.
- Evolution: Evolutions might be necessary during the operation of a VE when it is necessary to add and/or replace a partner.
- Dissolution: This is the phase when the VE finishes its business processes and dismantles itself.

More suitable for this dissertation is the two-step approach for the extended enterprise life cycle defined by [Für02]:

1. The extended enterprise foundation (the left arm in Figure 3.8) comprises negotiating, agreeing upon, and setting up a legal and technical framework for doing business (projects) together.
2. The extended enterprise projects (the right arm in Figure 3.8) correspond to the actual business transactions which are conducted within the previously defined extended enterprise foundation.

Activities related to the extended enterprise foundation are:

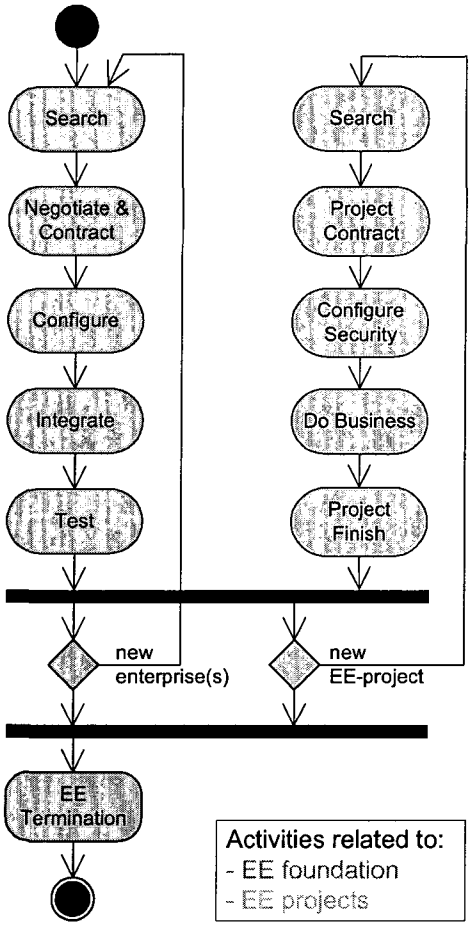


Figure 3.8: Extended Enterprise Life Cycle [Für02]

- Search: Members of the extended enterprise are searched and found.
- Negotiate and Contract: Negotiations include project-frameworks, security, and payment. This activity is concluded by the signing of the contract.
- Configure: Configuration of access control, authentication and authorization systems for providing extended enterprise members with secure business processes and services.

- Integrate: Integration of internal business processes and the external business processes to ensure autonomy, transparency, scalability and distribution.
- Test: Testing and performance analysis of the shared processes and removal of bottlenecks.
- EE Termination: In this activity the extended enterprise will be dissolved.

After the extended enterprise framework is established, specific projects can be realized. Activities related to the extended enterprise projects are:

- Search: The project group is established out of the set of extended enterprise members.
- Project Contract: Signing of the project contract according the framework contract.
- Configure Security: Configuration of project specific security on the basis of the general conventions.
- Do Business: This phase includes the following activities:
  1. invocation of business processes in the extended enterprise according to established extended enterprise contracts,
  2. management and control of invoked business processes and resources during real time operation,
  3. management and handling of business events,
  4. on-line performance evaluation and control of business processes.
- Project Finish: Finish project and reset security measures for the project.

The work of this dissertation considers only the activity "Do Business" of the extended enterprise projects. Therefore, the assumption is done that the extended enterprise is already created, configured, and in a steady state. Furthermore the project partners are defined, and the security adjustments are finished.

3.3 Workflow Management System Basics

Current and predicted market requirements, in consequence the general conditions to make money, force enterprises to continually adopt new technologies to handle their processes in a profitable manner. Successful enterprises have to satisfy a principally simply equation: enthusiastic customers, motivated employees, successful products, and efficient business processes. The concept of business processes is closely related to that of workflows. The WfMC (Workflow Management Coalition) defines a workflow as follows:

"The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules" [All00].

Workflow Management Systems (WfMS) are in general seen as a good and pragmatic facility to significantly improve efficiency and quality of business processes or parts of them. WfMS are process-oriented, holistic, and explicit [JBS97]. Holistic means that all aspects of the application's scope have to be modeled, including relationships and dependencies between entities of business processes. In the end every unit of an enterprise, regardless of its task, has to work together to guarantee success. Business process modeling (Chapter 3.4), the core of business engineering [Ham98], has to be done explicitly, explicitly in a way that no fact is expressed through others, and every identifiable thing

is modeled. The WfMC defines a WfMS as follows:

"A system that defines, creates and manages the execution of workflows through the use of software, running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participants and, where required, invoke the use of IT tools and applications" [Wor99].

When the Workflow Management Coalition began its work to define standards for the workflow industry sector in 1994, achieving workflow interoperability was seen as a key objective. Therfore, the WfMC elaborates a reference model, see Figure 3.9, for workflow management systems. In particular it defines mandatory functional components and their interfaces to the core component workflow enactment service, which consists of one or more workflow engines. A workflow engine is a software service or "engine" that provides the run time execution environment for a workflow instance [Wor99]. The WfMC defines five interfaces (Figure 3.9):

- Process Definitions (Interface 1) deals with getting the engineered process, respective workflow, from external tools in the workflow engine. It is far more than a read/write specification. Interface 1 covers the process definition language (see also Chapter 3.4). If one considers that the process language delimits the functional requirements of a workflow management system, the specification or choice is of broad influence to the performance of the overall system.

- Workflow APIs (Interface 2 & 3) have been combined. Interface 2 concerns notification services for clients. Interface 3 treats invocation of IT applications.

- Inter-Engine Workflow (Interface 4) opens local workflow management system boundaries for interaction with remote workflow management sys-

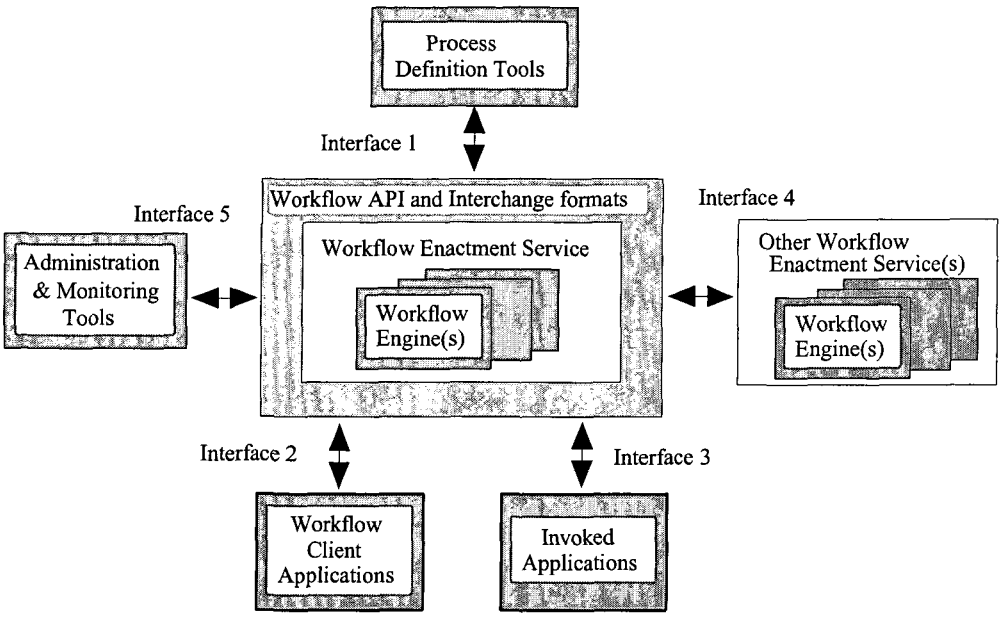


Figure 3.9: WfMC's Workflow Management System Reference Model [Wor99]

tems. One workflow engine may request another remote engine to enact a certain workflow from known definitions. Appropriate context and status information may be interchanged.

- Audit and Monitoring (Interface 5) serves tools with necessary information for logging the progress of actual instances and to administer running workflow instances. It delivers information for auditing and analyzing

As already mentioned, all workflow systems are process oriented. A process definition, a representation of what should happen, is created, and it typically comprises some sub-processes. Each process and sub-process comprises some activities. An activity is a single logical step in the process. Process definitions will describe all activities whether they are automatic or manual. After the process definition is uploaded into a workflow management system, instances of that process flow through the system. The instance of a process comprises of

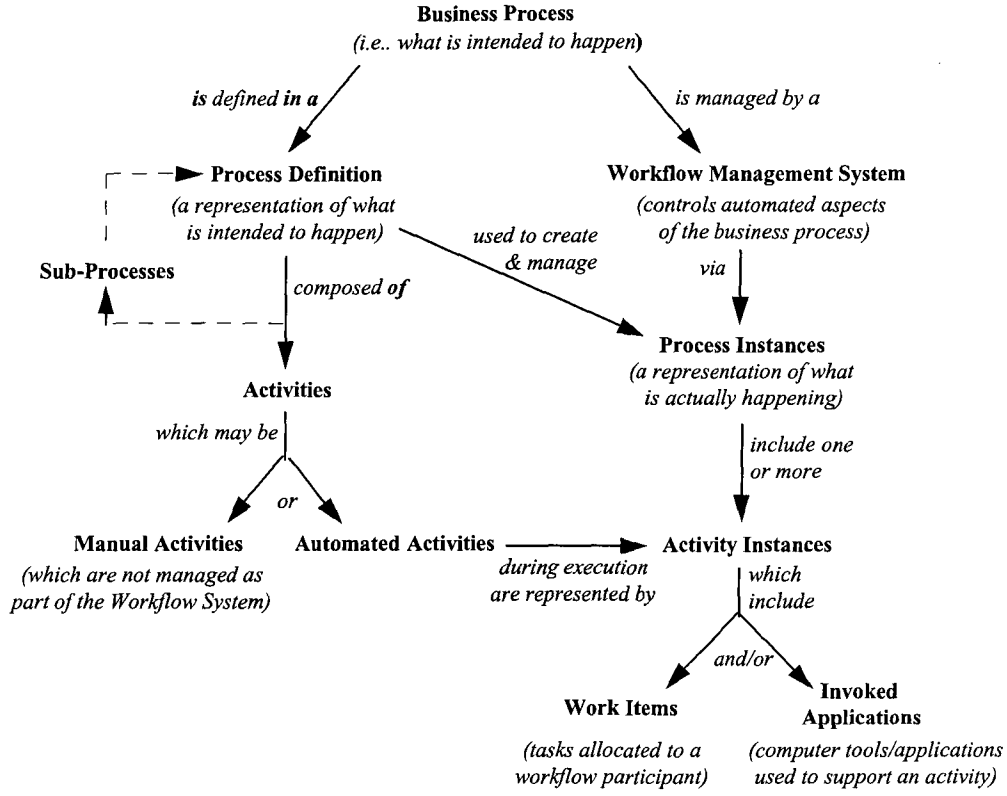


Figure 3.10: Relationships between Basic Terminology [Wor99]

activity instances that will include work items that are passed to an workflow participant for action, or to another application for action [All00]. Figure 3.10 shows the discussed relationships between the basic workflow terminologies.

3.4 Approved Standards for Business Process Modeling

The idea to orchestrate activities is not a new one and dates back to the definition of petri nets in the 1970ies. It has received major attention with the advent of modeling languages, workflow systems and most recently web services. Each of the distinct communities have proposed their own business

process description languages that can be defined as follows:

"The representation of a business process in a form which supports automated manipulation, such as modeling, or enactment by a workflow management system. The process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated IT applications and data, etc." [Wor99]

3.4.1 Unified Modeling Language (UML)

The Unified Modeling Language (UML) was created by Grady Booch, James Rumbauch, and Ivar Jacobson [BRJ99], and later standardized by the Object Management Group (OMG) in 1997. Since then, UML has quickly become the standard modeling language for software development. The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components.

The UML consists of nine different diagram types where each diagram is used for different purposes depending on the angle from which the system is viewed (Figure 3.11): [EP00]

- Class diagram: Describes the structure of a system. The structures are built from classes and relationships. The classes can represent and structure information, products, documents, or organizations.
- Object diagram: Expresses possible object combinations of a specific class diagram. It is typically used to exemplify a class diagram.

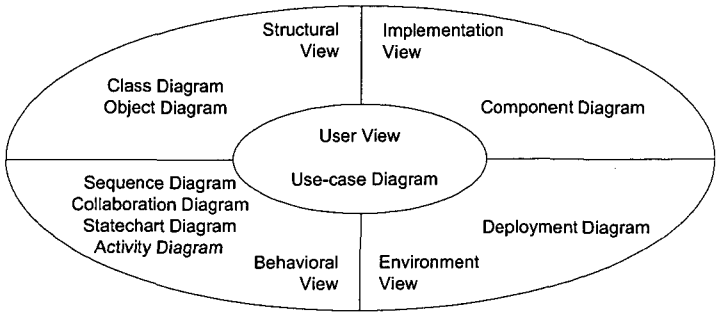


Figure 3.11: Model Views and UML Diagrams [Alh98]

- Statechart diagram: Expresses possible states of a class (or a system).
- Activity diagram: Describes activities and actions taking place in a system.
- Sequence diagram: Shows one or several sequences of messages sent among a set of objects.
- Collaboration diagram: Describes a complete collaboration among a set of objects.
- Use-case diagram: Illustrates the relationships between use cases. Each use case, typically defined in plain text, describes a part of the total system functionality.
- Component diagram: A special case of class diagram used to describe components within a software system.
- Deployment diagram: A special case of class diagram used to describe hardware within a software system.

Activity diagrams are very suitable to model business processes to a certain extent. Activity diagrams allow a good separation of control and data flow and

it is possible to model branching, concurrency, and loops. On the negative side is the absent of constructs for exceptions, compensation and aggregation. Roles can be modeled effectively with so called "swimlanes" [Obj01].

3.4.2 XML-based Standards describing Business Processes

The major candidates for business process modeling standards today have one thing in common: all are XML-based. Table 3.1 compares the most important XML-based business process definition languages with UML using the workflow patterns described in [vdAtHKB03]. If the standard directly supports the pattern through one of its constructs, it is rated "+". If the pattern is not directly supported, it is rated "+/-". Any solution which results in spaghetti diagrams or coding, is considered as giving no direct support and is rated "-". Additional note: MI means Multiple Instances.

3.4.2.1 XML-based Process Definition Language (XPDL)

The Workflow Management Coalition (WfMC) has defined a standard workflow reference model. The group has published an XML-based Process Definition Language (XPDL, [Wor02]) to facilitate process description interchange between different workflow system vendors. The language supports all important process modeling concepts like aggregation, loops, branching, concurrency, exceptions, and timing constraints. XPDL process definitions contain roles to model business partner. The language does not have a standard graphical representation.

Together with other WfMC standards, XPDL provides a framework for implementing business process management and workflow engines, and for designing, analyzing, and exchanging business processes. XPDL is the culmi-

Pattern	UML	XPDL	BPML	BPEL4WS
Sequence	+	+	+	+
Parallel Split	+	+	+	+
Synchronization	+	+	+	+
Exclusive Choice	+	+	+	+
Simple Merge	+	+	+	+
Multi Choice	-	+	-	+
Synchronizing Merge	-	-	-	+
Multi Merge	-	-	+/-	-
Discriminator	-	-	-	-
Arbitrary Cycles	-	+	-	-
Implicit Termination	-	+	+	+
MI without Synchronization	-	-	+	+
MI with a Priori Design Time Knowledge	+	+	+	+
MI with a Priori Runtime Knowledge	+	-	-	-
MI without a Priori Runtime Knowledge	-	-	-	-
Deferred Choice	+	-	+	+
Interleaved Parallel Routing	-	-	-	+/-
Milestone	-	-	-	-
Cancel Activity	+	-	+	+
Cancel Case	+	-	+	+

Table 3.1: Comparison of Modeling Standards [Tec03]

nation of a fifteen-month effort by multiple vendors and users to provide a standard that satisfies the needs of diverse organizations. The specification is intended for use by software vendors, system integrators, consultants and any other individual or organization concerned with the design, implementation, and analysis of business process management systems as well as with

interoperability among workflow systems.

3.4.2.2 Business Process Modeling Language (BPML)

The BPML (Business Process Modeling Language) is a meta-language for the modeling of business processes, just as XML is a meta-language for the modeling of business data [Bus01]. BPML provides an abstracted execution model for collaborative and transactional business processes based on the concept of a transactional finite-state machine.

The BPML specification is provided by the business process management initiative (BPMI) organization. BPMI.org is a non-profit corporation that empowers companies of all sizes, across all industries, to develop and operate business processes that span multiple applications and business partners, behind the firewall and over the Internet. The initiative's mission is to promote and develop the use of Business Process Management (BPM) through the establishment of standards for process design, deployment, execution, maintenance, and optimization.

In much the same way XML documents are usually described in a specific XML Schema layered on top of XML, BPML processes can be described in a specific business process modeling language layered on top of the extensible BPML XML Schema. BPML represents business processes as the interleaving of control flow, data flow, and event flow, while adding orthogonal design capabilities for business rules, security roles, and transaction contexts.

3.4.2.3 Business Process Execution Language for Web Services (BPEL4WS)

A joint effort of IBM, BEA and Microsoft produced the Business Process Execution Language for Web Services (BPEL4WS, [IBM02]). The language is designed to combine the strength of its predecessors, although some compromise was inevitable resulting in a mixture of block oriented (inherited from Microsoft's XLANG [Tha01]) and graph oriented (inherited from IBM's WSFL [IBM01]) notation. Nevertheless BPEL4WS is an expressive business process modeling language containing XLANGs powerful modeling concepts (aggregation, branching, concurrency, loops, exceptions, compensation, and time constraints) and proposes a reasonable life cycle and message correlation model.

The language allows specifying business processes and how they relate to Web services. This includes specifying how a business process makes use of Web services to achieve its goal, as well as specifying Web services that are provided by a business process. Business processes specified in BPEL4WS are fully executable and portable between BPEL4WS-conformant environments. A BPEL4WS business process inter-operates with the Web services of its partners, whether or not these Web services are implemented based on BPEL4WS. Finally, BPEL4WS supports the specification of business protocols between partners and views on complex internal business processes.

With the backing of IBM and Microsoft, BPEL4WS is likely to become the de facto standard for webflow description.

3.5 Concepts and Standards behind Web Services

For the extended enterprise, the Internet is generally accepted as the most adequate medium of communication. Evidence for this can be found in most e-business-related definitions and virtual enterprise approaches. In spite of the Internet-hype is over there is no doubt that automated communication using the Internet enables significant efficiency increases in enterprise-spanning business process automation and integration.

Web services are the fundamental building block for the next-generation communication and collaboration among distributed applications over the Internet. A Web service represents a unit of business, application, or system functionality that can be accessed over the Web. Web services are applicable to any type of Web environment, whether Internet, Intranet, or Extranet, whether with a focus on business-to-consumer, business-to-business, department-to-department, or peer-to-peer communication. A Web service consumer could be a human user accessing the service through a desktop or wireless browser; it could also be an application program or another Web service [Sun03a]. A Web service exhibits the following basic characteristics:

- A Web service is accessible over the Web.
- Web services communicate using XML messages over standard Web protocols.
- A Web service exposes an XML interface description.
- A Web service is registered and can be located through a Web service registry.

Figure 3.12 shows the Web Service Model where three kinds of roles and the performed operations can be identified. The roles are as follows [Ira02]:

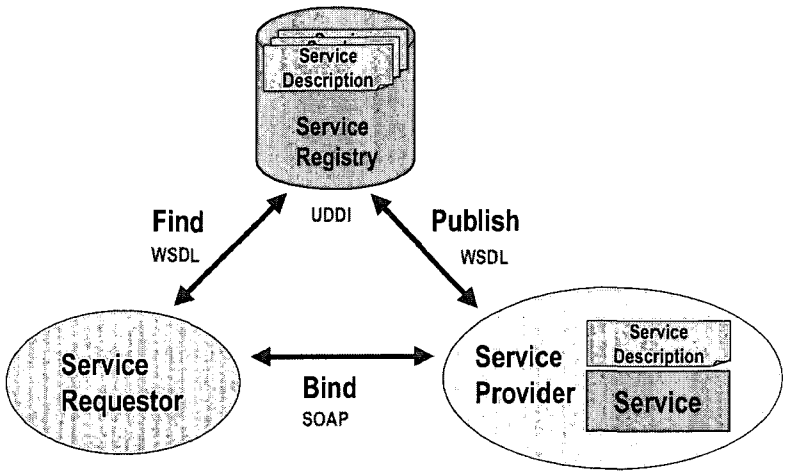


Figure 3.12: Web Service Model [SFW02b]

- Service Provider: A service provider is the entity that creates the Web Service. Typically, the service provider exposes certain functionality in their organization as a Web Service for any organization to invoke. The service provider needs to do two things to reach the full potential of a Web Service. First, it needs to describe the Web Service in a standard format, which is understandable by all organizations that will be using that Web Service. Secondly, to reach a wider audience, the service provider needs to publish the details about its Web Service in a central registry that is publicly available to everyone.

- Service Requestor: Any organization, any entity using the Web Service created by a service provider is called a service requestor. The service requestor can know the functionality of a Web Service from the description made available by the service provider. To retrieve these details, the service requestor does a find in the registry to which the service provider had published its Web Service description. More importantly, the service requestor is able to get from the service description, the mechanism to

bind to the service provider's Web Service and in turn to invoke that Web Service.

- Service Registry: A service registry is a central location where the service provider can list its Web Services, and where a service requestor can search for Web Services. Service providers normally publish their Web Service capabilities in the service registry for service requestors to find and then bind to their Web Service. Typically, information like company details, the Web Service that it provides, and the details about each Web Service including technical details is stored in the service registry.

In the Web Service model there are three operations that are fundamental to make Web Services work - "find", "bind", and "publish". To achieve inter-application communication irrespective of the kind of language the application is written in or the platform the application is running on, the following standards for each of these three operations are defined: SOAP, WSDL, and UDDI (Table 3.2). Web services are built on existing communication standards like HTTP, SMTP, or FTP, which make them transport-independent. In the present, HTTP is the most ubiquitous transport protocol.

Service Publication/Discovery	UDDI
Service Description	WSDL
XML Messaging	SOAP
Transport Network	HTTP(s),SMTP,FTP,...

Table 3.2: Web Service Stack [Ira02]

3.5.1 Simple Object Access Protocol (SOAP)

SOAP (Simple Object Access Protocol) is a lightweight, message oriented, XML based RPC (Remote Procedure Call) protocol for exchange of information in a decentralized environment. SOAP is completely independent of any platform, operating system or programming language and can easily be used over the Internet with the ubiquitous HTTP protocol [Wor00].

The motivation for development of SOAP was, that existing RPC protocols as DCOM, CORBA-IIOP or RMI either was designed on using in LAN and eventually are blocked by the firewall and proxy servers or are depending on platform respectively program language. Distributed application, which are constipated to communicate with many different partners over Internet cannot grab to such protocols. SOAP uses on the whole the HTTP protocol as carrier protocol. But it will be planed that other carrier protocol can be used. SOAP defines three structure elements (Figure 3.13):

- the envelope (must exist)
- the header (may exist) and
- the body (must exist)

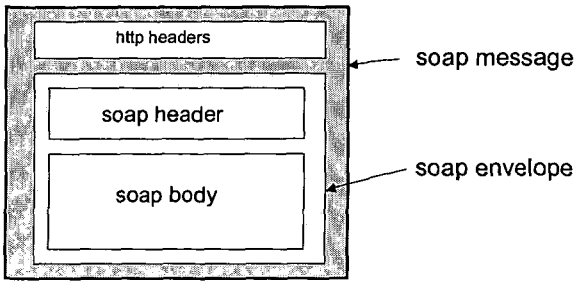


Figure 3.13: The Structure of SOAP [SFW02a]

The SOAP envelope construct defines an overall framework for expressing what is in a message, which should deal with it and whether it is optional or mandatory. The envelope is the top element of the XML document representing the message and has the header and body elements as sub-elements. The header is generic mechanism for adding features to a SOAP message without prior agreement between the communication parties. The body is a container for mandatory information intended for the ultimate recipient of the message. It defines, which object and method is addressed and which parameters are transferred.

3.5.2 Web Services Description Language (WSDL)

As communications protocols and message formats are standardized in the Web community, it becomes increasingly possible and important to be able to describe the communications in some structured way. WSDL (Web Services Description Language) addresses this need by defining an XML grammar for describing network services as collections of communication endpoints capable of exchanging messages. WSDL service definitions provide documentation for distributed systems and serve as a recipe for automating the details involved in applications communication [Wor01].

A WSDL document defines services as collections of network endpoints, or ports. In WSDL, the abstract definition of endpoints and messages is separated from their concrete network deployment or data format bindings. This allows the reuse of abstract definitions: messages, which are abstract descriptions of the data being exchanged, and port types, which are abstract collections of operations. The concrete protocol and data format specifications for a particular port type constitute a reusable binding. A port is defined by associating a network address with a reusable binding, and a collection of ports defines a

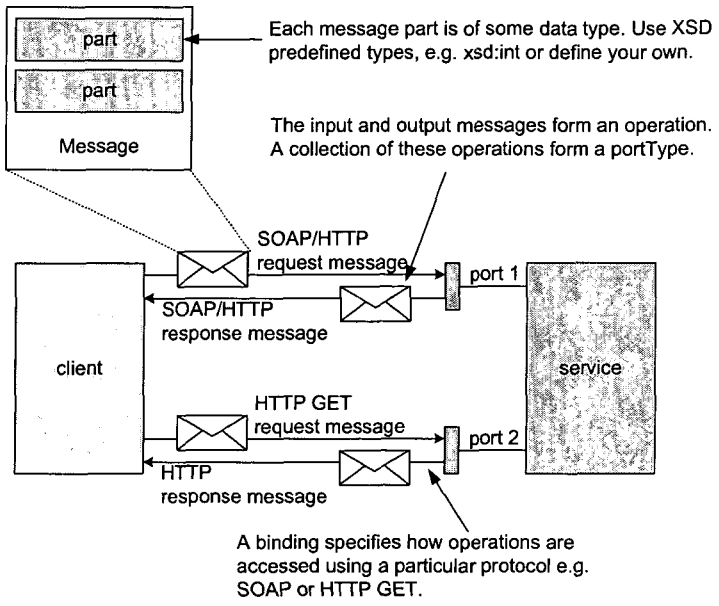


Figure 3.14: Description of WSDL Elements [SFW02a]

service (Figure 3.14).

3.5.3 Universal Description, Discovery, and Integration (UDDI)

The UDDI (Universal Description, Discovery, and Integration) initiative is an industry consortium including IBM, Intel, Microsoft, Oracle, SAP, and Sun Microsystems. UDDI creates a global, platform-independent, open framework to enable businesses to (1) discover each other, (2) define how they interact over the Internet, and (3) share information in a universal, Web-based business directory called the UDDI Business Registry that will more rapidly accelerate the global adoption of B2B eCommerce [Org03].

The UDDI specifications consist of an XML schema for SOAP messages, and a description of the UDDI API specification. Together, these form a base information model and interaction framework that provides the ability to pub-

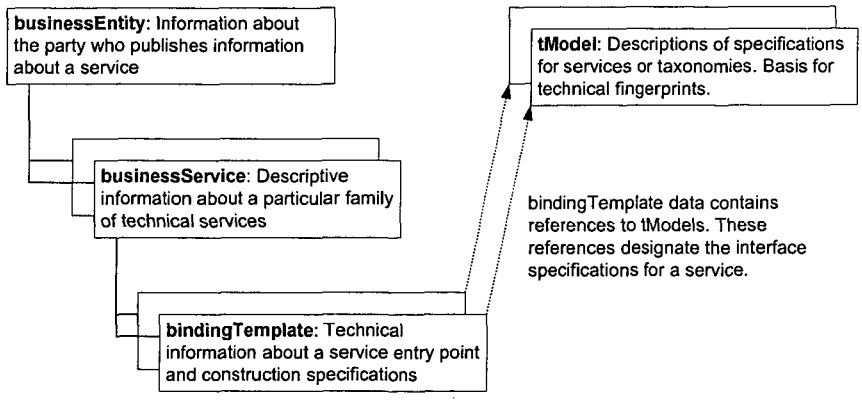


Figure 3.15: Core Information Types of UDDI [FSW02a]

lish information about a broad array of Web Services. The core component of the UDDI project is the UDDI business registration, an XML file used to describe a business entity and its Web Services. Conceptually, the information provided in a UDDI business registration consists of three components: "white pages" including address, contact, and known identifiers; "yellow pages" including industrial categorizations based on standard taxonomies; and "green pages", the technical information about services that are exposed by the business. The core information model used by the UDDI registries is defined in an XML schema. This UDDI XML schema defines four core types of information that provide the kinds of information that a technical person would need to know in order to use a partners Web Services. These are (see Figure 7): business information; service information, binding information; and information about specifications for services.

Chapter 4

State of the Art

In this chapter the international state of the art in enterprise-spanning business process integration and automation inside an extended enterprise is discussed.

Due to the enterprise-centric view in the 80s and 90s, the introduced systems have generated isolated automation islands. The enterprise application integration (EAI) technologies have been developed to connect them. The focus of these technologies was the avoidance of many, expensive point-to-point interfaces between the different enterprise systems inside one enterprise.

To point out the usage of XML technology for the automation in the field level, the emerging conception of PROFInet is mentioned. Such developments are very important to establish a continuous vertical integration through all levels of an enterprise.

Since the Internet boom in the 90s, nearly all software companies (have to) present in their marketing slides the latest technology buzzwords to bring out their innovation strategy. Currently this development results in an equalization of product presentations, because every vendor has to provide "e-"products. Therefore, in this chapter only an assortment of extended enterprise projects

and systems, which are highly relevant to this work, are discussed.

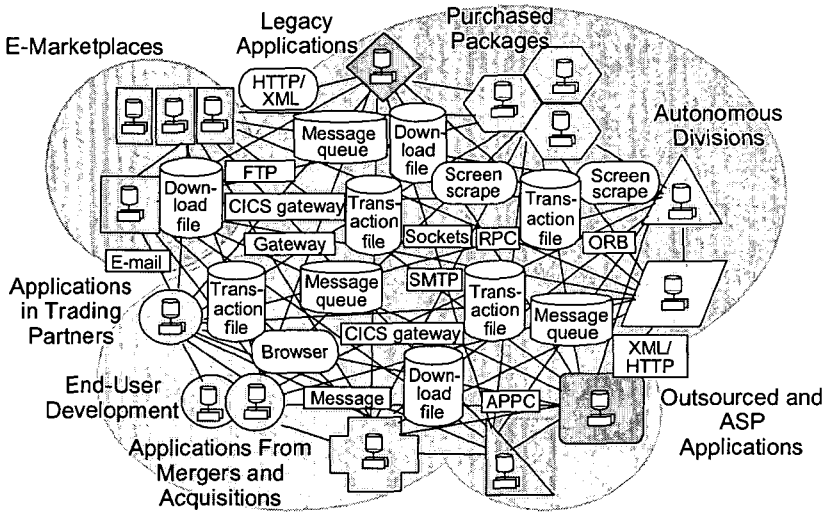
Finally, this dissertation is positioned in opposite to the current state of the art, particularly highlighting the newness of this research work.

4.1 Methodology for Enterprise Application Integration

The trend toward enterprise application integration (EAI) is a logical progression. After implementing different isolated information systems which are often designed as autonomous, stand-alone applications, companies have discovered that they need to link to other, partly legacy applications and/or data to complete the big picture. According [Lin00], *"EAI is the unrestricted sharing of data and business processes among any connected applications and data sources in the enterprise"*. And for the Gartner Group, *"enterprise application integration means making independently designed systems work together"* [Uni03].

As proof for the enormous demand for suitable EAI solutions, some facts are given by market research companies. The firm Meta Group estimates that the average Global 2000 firm has roughly 49 systems applications [Ran03]. Forrester Research Inc. and Gartner Group Inc. both estimate that companies spend up to 35% of its IT budget to realize point-to-point interfaces between the applications [Uni03]. The Fortune 1000 companies spend 100 billion euro per year to integrate their IT systems [Mic03].

To save money and improve business responsiveness, companies have to avoid the enterprise reality of the "network spaghetti" of software systems and point-to-point connections between them (Figure 4.1). Using EAI enterprises can reduce the possible amount of interfaces from n*(n-1)/2 down to only n, which will enhance the extensibility and maintainability of the network.



APPC	Advanced Program-to-Program Communications	**HTTP**	Hypertext Transport Protocol
ASP	Application service provider	**ORB**	Object request broker
CICS	Customer Information Control System	**RPC**	Remote procedure call
HTML	Hypertext Markup Language	**SMTP**	Simple Message Transfer Protocol
		XML	Extensible Markup Language

Source: Gartner Research

Figure 4.1: The Enterprise Reality: Network Spaghetti [Pez02]

Before organizations can introduce EAI, they must select which processes and data elements require integration. This process can take on several dimensions: [Lin00]

- Data-level EAI is the process of moving data between data stores. This can be described as extracting information from one database, perhaps processing that information as needed, and updating it in another database. The advantage here is the cost of using this approach because it is not necessary to change the application.

- Application interface-level EAI refers to the leveraging of interfaces exposed by custom or packaged applications. The only limitations are the specific features and functions of the application interfaces. Because of

the packaged applications realize the interfaces in very different ways, it is necessary to extract the information from one application and place it in a format understandable by the target application.

- Method-level EAI is the sharing of the business logic that may exist within the enterprise. For example, the method for updating a customer record may be accessed from any number of applications, and applications may access each other's methods without having to rewrite each method within the respective application.

- User interface-level EAI is a more primitive approach. Using this scenario, developers are able to bundle applications by using their user interfaces as a common point of integration.

The enabling technology for EAI is middleware. Middleware is software that hides the complexities of the underlying operating system and network in order to facilitate the easy communication and integration of various systems in the enterprise. The first middleware solutions were remote procedure calls, message-oriented middleware, and transactional middleware (Figure 4.2). The next generation of middleware is here with new categories such as message brokers, distributed objects, and application servers. In the following the most important types of middleware are discussed in brief.

4.1.1 Remote Procedure Call (RPC)

Remote procedure calls (RPCs) are the oldest and easiest to understand type of middleware. RPC enables developers to write a program that runs on one machine, and makes function calls to another machine, thus utilizing the resources of two machines. Due to RPCs are synchronous (stop the execution of the program in order to carry out the call) they are known as blocking

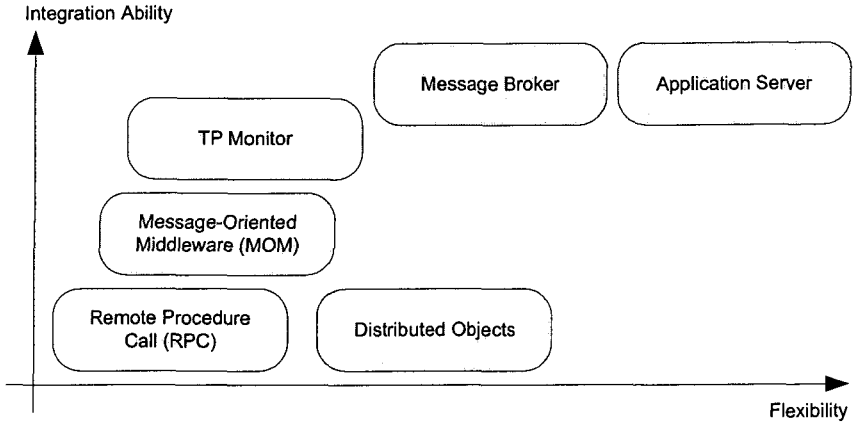


Figure 4.2: Types of Middleware [Uni03]

middleware. In fact, most RPCs are not well-performing middleware products. Their advantage is their simplicity.

4.1.2 Message-Oriented Middleware (MOM)

Message-oriented middleware (MOM) was created to address some shortcomings of RPCs though the use of messaging. In general, MOM is characterized by a peer-to-peer distributed computing model supporting both synchronous and asynchronous interaction between distributed computing processes [Ste03]. The asynchronous model allows the application to continue processing during the middleware manages the message transport. MOM supports two different models: point-to-point and message queuing.

4.1.3 Transaction Processing Monitor

Transaction processing (TP) Monitor is a software that monitors a transaction as it passes from one stage in a process to another. A transaction is a unit of work with a defined start- and end-point. If an application logic is encapsulated

within a transaction, then the transaction either completes, or is rolled back completely including undo of all made changes. Because transactions are small units of work, they are easy to manage and process within the TP monitor environment.

4.1.4 Message Broker

Message brokers are servers that broker messages between two or more applications. In addition to brokering messages, they provide several additional services: message translation, rules processing, intelligent routing, message warehousing, flow control, repository services, directory services, and adapters. An advantage to message brokers is that they leave systems "where they are", minimizing change while still allowing data to be shared.

4.1.5 Distributed Object

Distributed objects facilitate inter-application communications. Distributed objects are really small application programs that use standard interfaces and protocols to communicate with one another. For example, two distributed objects run on different operating systems. Because both objects are created using the same standard, and both objects use the same standard communication protocol, then the objects should be able to exchange information and carry out application functions by invoking each other's methods. Today, there are different types of distributed objects on the market: CORBA, COM+, and EJB.

CORBA (Common Object Request Broker Architecture) is a very general and open industry standard for working with distributed objects. CORBA was developed by the Object Management Group (OMG), a large non-profit

consortium that includes the major software vendors (Sun, DEC, IBM, Apple, Hewlett-Packard, etc.) as well as end users. CORBA allows the interconnection of objects and applications, regardless of the computer language of the applications that provide or use the objects, the machine architecture of the computers involved, and the geographical location of the computer. The Object Request Broker (ORB), which is a part of CORBA, is the central part of the Object Management Architecture (OMA) and provides an infrastructure, which enables communication to the objects independently of the specific platforms and implementations [Ros98].

COM+ is a standard promoted by Microsoft. COM+ is an extension to COM that builds on COM's integrated services and features, making it easier for developers to create and use software components in any programming language. COM (Component Object Model) specifies how to build components that can be dynamically interchanged. The specification include interface standards and communication protocols [Rog97].

EJB (Enterprise JavaBeans) is a specification of Sun Microsystems for an industry standard that simplifies the process of building enterprise-class distributed component applications in Java. It defines both the component model used by the developers of application solutions and the infrastructure implemented by vendors of application servers. This component architecture allows writing scalable, reliable, and secure middleware applications without writing your own complex distributed component framework. Enterprise JavaBeans servers provide automatic support for middleware services such as transactions, state management, multi-threading, resource pooling, security, database connectivity, and more. The EJB specification 2.0 defines three different types of Enterprise Beans: session (stateful or stateless), entity (container-managed or bean-managed persistence), and message-driven [DP02][Rom99].

EJBs are the cornerstone of Sun's J2EE (Java 2 Platform, Enterprise Edition). The mission of J2EE is to provide a platform-independent, portable, multiuser, secure, and standard enterprise-class platform for server-side deployments written in the Java language [MH00]. The J2EE platform provides a multitier distributed application model. This means application logic is divided into components according to function, and the various application components can be installed on different machines. The J2EE architecture defines a client tier, a middle tier (consisting of one or more sub-tiers), and a backend tier providing services of existing information systems (Figure 4.3). The client tier supports a variety of client types, both outside and inside of corporate firewalls. The middle tier supports client services through Web containers in the Web tier and supports business logic component services through EJB containers in the EJB tier. The backend tier supports access to existing information systems by means of standard APIs.

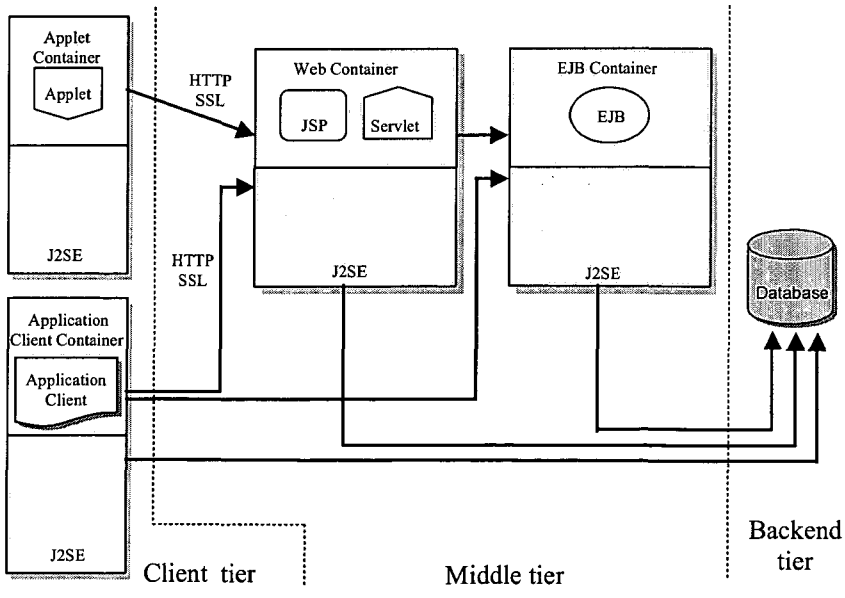


Figure 4.3: J2EE Multitiered Application [Uni03]

All J2EE components are supported by a system-level entity called container, in which they are installed during deployment. In form of this container a J2EE server provides underlying services such as life cycle management, security, deployment, and runtime services for each component. There are containers for Applets, application clients, Web container and EJB container. Containers provide all application components with the Java 2 Standard Extension (J2SE) platform APIs.

4.1.6 Application Server

An application server is a software platform designed to simplify building, deploying, running, and maintaining large-scale, multiuser applications. The basic functions of an application server fall into three groups:

- Hosting components: The role of the middle tier in the three-tier model is to encapsulate the business logic of the application. Application servers accomplish this by hosting components in way that enhances their scalability, reliability, and manageability, adding such capabilities as queuing (the ability to guarantee that a message from one component to another will be delivered) and pooling (the ability of components to be kept in a "ready" state, awaiting a request).

- Connecting to data: The ultimate function of most corporate applications is to retrieve or update information that is stored in databases. Therefore, application servers must allow middle-tier components to connect to a variety of databases and coordinate the activities of the components in a way that ensures the integrity of the underlying data.

- Supporting UIs (User Interface): Users can interact with applications through both browser-based and custom application UIs (often referred

to as thin and thick clients, respectively). Application servers typically support both types of UIs by including class libraries or APIs (Application Programming Interface) for creating custom client applications and for integrating with Web servers to enable browser-based UIs.

4.2 PROFInet: Emerging Conception for Automation

Automation technology has been in a process of continuous change since it was founded. A major innovation in this connection is the use of fieldbus technology (market leader: PROFIBUS). It enables the changeover from central to decentralized automation systems.

Today's face of automation technology is being shaped increasingly by another element - information technology and its principles and standards. That integration opens up possibilities of global data communication between automation systems, which is the aim of the Ethernet-based communication standard PROFInet [Pro03a].

To become the totally integrated automation concept of the future, PROFInet provides the following features:

- Defines a standard for multi-vendor, plant-wide integration of distributed automation systems
- Implements an object-based component model based on Microsoft's Component Object Model (COM, see chapter 4.1.5)
- Uses established IT standards such as Ethernet and XML
- Integrates existing PROFIBUS systems without adaption

Through the use of established information technology, PROFInet wants to

connect the field level with the top management level ("vertical integration"). Due to PROFInet realizes the communication with COM objects, there is one main restriction of this approach: communication is not possible if there are firewalls in use to secure the Internet connections of enterprises.

4.3 Extended Enterprise Projects and Systems

4.3.1 US Project NIIIP

NIIIP (National Industrial Information Infrastructure Protocols and Related Projects) was an US initiative with the intention to support the formation of industrial virtual enterprises (VEs) and to provide technologies that allow virtual enterprise participants to collaborate within a heterogeneous computing environment. The project ran in two phases: first phase from September 1994 through December 1997; second phase from January 1997 through December 2000.

The NIIIP consortium set oneself the following targets [NII03]:

- Establish standard-based software infrastructure protocols, that integrates a variety of distributed business systems, business processes, data and computing environments.
- Implement NIIIP from emerging, existing, and de-facto standards and system technologies and accelerate consensus on standards that promote the deployment of virtual enterprises.
- Provide the technical foundation for implementing virtual enterprises by developing software, toolkits, and a resource file, documenting the experiences of virtual enterprises.

The NIIIP project was structured into four parts [NII03]:

- VE Requirements deal with the formation of VE collaborations, their operations, and termination after attaining the desired purposes.
- NIIIP Protocols are responsible for the whole communication inside the VE collaboration. The protocols are divided into layers, which controls authorization, access rights, and validation in all type of data exchange.
- NIIIP Components are different types of workflow and management software modules, which are highly integrated.
- VE Member Resources contain information about the assigned members of VEs.

In its general scope, NIIIP addresses the complete virtual enterprise "life cycle": identifying market needs, looking for partners, assisting the negotiation process between partners, and supporting virtual enterprise instantiation, operation and dissolution. However NIIIP is based on a too "harmonized" view of the business world, and the focus of the project was only the initialisation of industrial virtual enterprises. But the project was a very important impulse for the developments in the area of enterprise networking.

4.3.2 European Research Project PRODNET II

The European Esprit project PRODNET II (October 1996 until September 1999) aims to design and develop a reference architecture and an open platform to support industrial virtual enterprises with special focus on the needs of SMEs. PRODNET II separates the internal functionalities from the network-related ones and develops the necessary mappings to the legacy systems. To

support this environment a basic infrastructure could consider two main modules as shown in Figure 4.4 [Pro03b]:

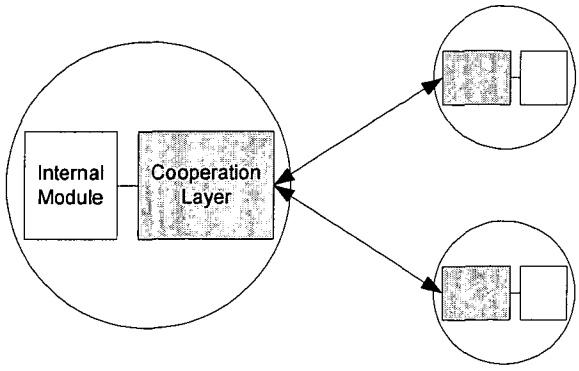


Figure 4.4: General Approach for a VE environment by PRODNET II [Pro03b]

- The internal module represents the autonomous unit of a particular company. It includes the complete structure of the company's information (databases, information systems, etc.) and all the internal decision making processes or enterprise activities.
- The cooperation layer contains all the functionalities for the interconnection between the company and the network. It represents the communication and coordination role and works as the interlocutor of the company within the net.

This research project has separated the internal functionalities from the network-related ones and has considered the internal systems as a "black box". Therefore, the focus was the development of mappings to legacy systems and not the realization of new business functionality.

4.3.3 IMS Research Project GLOBEMEN

IMS (Intelligent Manufacturing Systems) is an industry-led, international research and development program established to develop the next generation of manufacturing and processing technologies. "GLOBEMEN" (Global Engineering and Manufacturing in Enterprise Networks) is one of such a project. The duration of GLOBEMEN was three years (January 2000 until January 2003). The estimated effort was 1000 person-months. Built on the cognitions from the predecessor IMS project "GLOBEMAN21" (March 1996 until March 1999), this project had the aim to define and develop the architecture for globally distributed product life cycle management, project and manufacturing management [Glo03].

The main goal of this project was the development of a reference architecture for virtual enterprises called VERAM (Virtual Enterprise Reference Architecture and Methodology), which is a virtual enterprise specialization of GERAM (Generalized Enterprise Reference Architecture and Methodology, ISO15704:2000). The architectural framework positions elements that support modeling, formation/set up, management and ICT support of VEs, such as reference models, and supporting tools and infrastructures. VERAM consists of three layers (see Figure 4.5): [Glo03]

- Virtual Enterprise Concept
- VERA - Virtual Enterprise Reference Architecture
- VERAM Components

The Virtual Enterprise Concept introduces the concepts of the virtual enterprise and the enterprise network. The main purpose of the network is to prepare the life cycle of VEs and the products created by the VEs as well as to

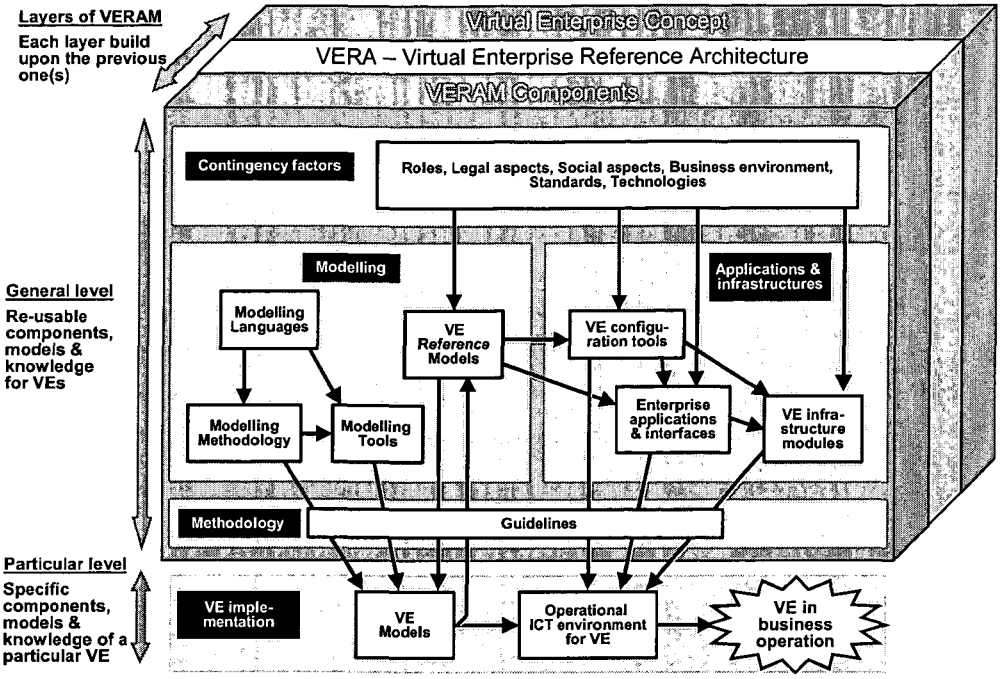


Figure 4.5: VERAM - Virtual Enterprise Reference Architecture and Methodology [Glo03]

manage the life cycle of VEs. The network should be seen as a potential from which different VEs can be established in order to satisfy diverse customer demands. [Glo03]

The layer VERA organizes the virtual enterprise related generic concepts recommended for use in virtual enterprise engineering and integration projects. VERA illustrates the logical, recursive relationships between the network entity, the VE entity, and the product entity. Each of these three entities is represented by a life cycle describing possible phases an entity can be in throughout its life span from identification to decommission. VERA illustrates that the network can create VEs in its operational phase and, correspondingly, that a VE can create products and/or services in its operational phase (see Fig-

ure 4.6). [Glo03]

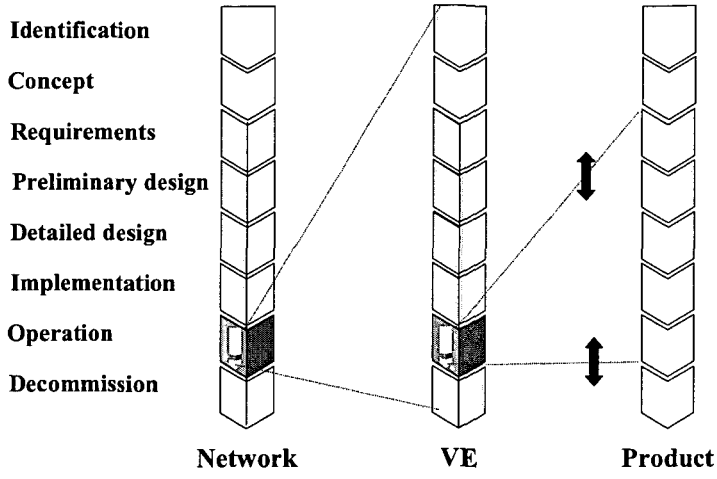


Figure 4.6: VERA - Virtual Enterprise Reference Architecture [PvdB00]

The VERAM components are used during the application of the architectural framework in practice. It consists of those tools, applications, models, and so on, that can be used during the formation and operation of VEs and enterprise networks. Perhaps the most important component is the guidelines that indicate how the tools, applications, and models should be used in practice. [Glo03]

Three main business processes were addressed during the project: sales and service, inter-enterprise project management, and distributed engineering. The results of the project were applied as prototype solutions in the business processes of the involved industrial companies. [Glo03]

Regarding the status of VERAM at the end of the GLOBEMEN project, some parts of the framework are more mature than others. The foundations are relatively stable and mature, but the top layer might be elaborated upon as new insight becomes available. This is an ongoing process, which should be continued beyond the GLOBEMEN project. [Glo03]

4.3.4 SAP R/3 System

After the Internet, SAP R/3 (SAP stands for Systems, Applications and Products in Data Processing) is one of the most important topics in the computer industry, and the company that developed it, SAP AG, has become one of the most successful in the software market. SAP AG was founded in 1972 by four former IBM employees. After the introduction of SAP R/3 in 1992, SAP AG has become the world's leading vendor of standard application software in the area of enterprise resource planning (ERP).

From an architectural and functional point of view, SAP R/3 is an open client/server back-office system, designed to manage business information needs of an enterprise. That means, among other things, that it is a fully integrated and standard business application. For the real business world, it means SAP R/3 is a standard software system - as opposed to a custom-developed one - that can model the business practices of an enterprise in its own data model. To "customize" the SAP system to the needs of one organization, the modeling tool ARIS (= Architecture of Integrated Information Systems, [Sch01]) will be used. The whole data flow of SAP R/3 works in an integrated way, which means that data needs to be entered just once, and the system automatically triggers or updates other logically related functions or data. For example, booking a sales order can trigger logically related functions in the production plan and shipping information systems, which, when appropriate, start the billing process and end up automatically updating the company's financial statements. [Her97]

In more detail, SAP R/3 is based on a three-tier client/server architecture shown in Figure 4.7:

- Presentation Server: It provides a graphical user interface usually running

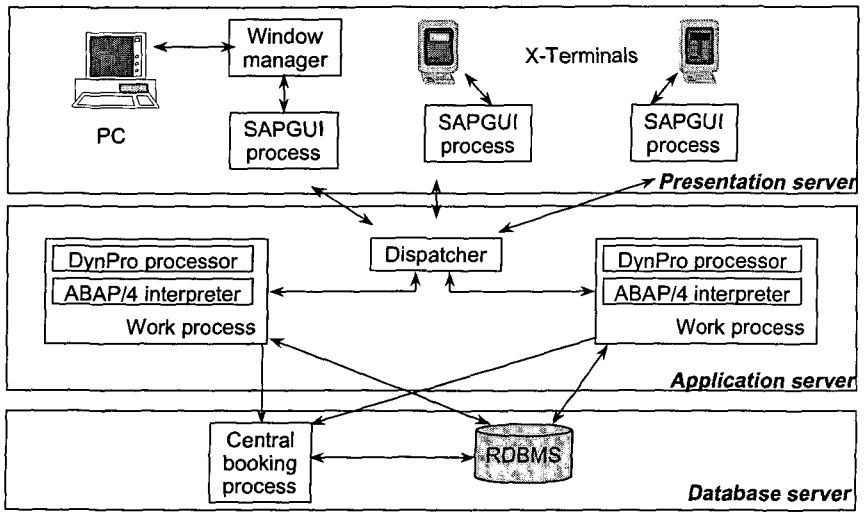


Figure 4.7: Three-Tier Client/Server Architecture of SAP R/3 [MZ98]

on PCs that are connected with the application servers via a local or a wide area network.

- Application Server: It comprises the business administration "know-how" of the system. It processes pre-defined and user-defined application programs. Application servers are usually connected via a local area network with the database server.

- Database Server: It is implemented on top of a (second party) commercial database product that stores all data of the system (e.g. business data of a company; all of SAP R/3-internal control data; code of all application programs).

A very common way for SAP to illustrate the R/3 system is the one shown in Figure 4.8, with the R/3 kernel system providing the necessary integration and infrastructure for the R/3 applications.

The R/3 kernel makes use of standard communications and application pro-

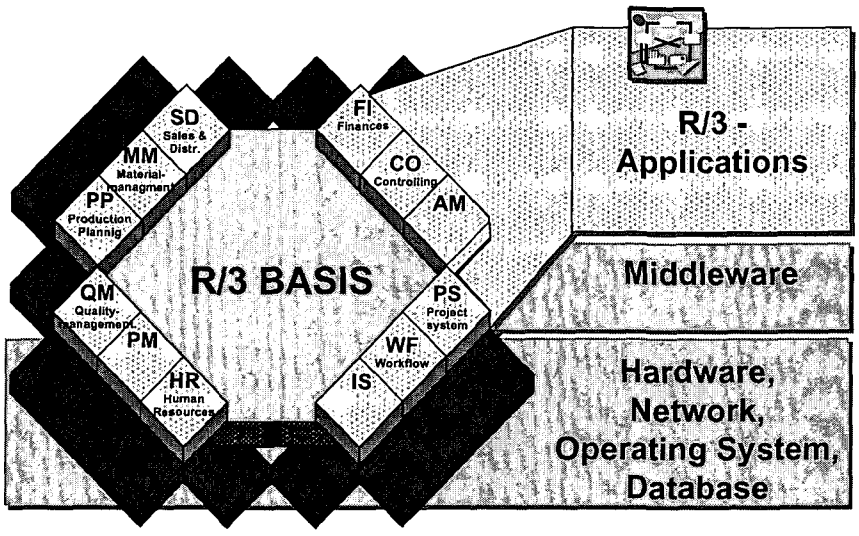


Figure 4.8: SAP R/3 Applications and their Technological Basis [MZ98]

gram interfaces to access the operating system, the database, and the network. This kernel layer is located below the application logic and data layers of the system and operates independently from the applications. This architecture allows users to change system configuration and install new systems without interrupting or altering the applications themselves. [Her97]

Although the R/3 system has the lion's share of the ERP market, it has received criticism in certain areas - a major one for not being responsive enough to its customers' need to interface data into and out of R/3. With the advent of more standard interfaces such as XML, SAP will gradually open up the system. But at this point, it still uses predominantly proprietary interfaces such as IDOC (Intermediate Document). This approach will be only advantageous in those instances where customers have followed SAP's strategies implicitly, which requires a rethink of the Ptolemy vs. Copernicus debate. Placing your integration server at the center of your solar system gives you wide control over many different applications, of which SAP may be one. Due to it's hard

to interface the SAP system leading to the assumption that SAP R/3 is the center of the enterprise solar system. [DLT00]

4.3.5 e-Business Systems

Nowadays, e-commerce is entering the third phase, called e-business, with a focus on how the Internet can impact profitability of firms. In its first phase (1994 - 1997), e-commerce was about presence by designing the first Web pages. The second phase (1997 - 2000) was about transactions between buyers and sellers who never would have found each other in the past.

In addition to encompassing e-commerce, e-business includes both front- and back-office applications that form the core engine for modern business. Thus, e-business is not just about e-commerce transactions or about buying and selling over the Web. It's the overall strategy of redefining old business models, with the aid of technology, to maximize customer value and profits. [KR99]

The e-business design built on an application architecture has become the main topic as more companies than ever integrate applications to streamline operations and compete in the e-commerce arena. But disparate applications are like modular building blocks - they have to be put together systematically to create an e-business enterprise. In most cases, the current e-business architecture is a combination of front-office applications with the back-office application - mostly the SAP R/3 system (see chapter 4.3.4) - as shown in Figure 4.9.

Most e-business application vendors were founded as independent companies. SAP is the exception: this ERP vendor set out to extend its business into new application areas. Other ERP-vendors though have decided that their

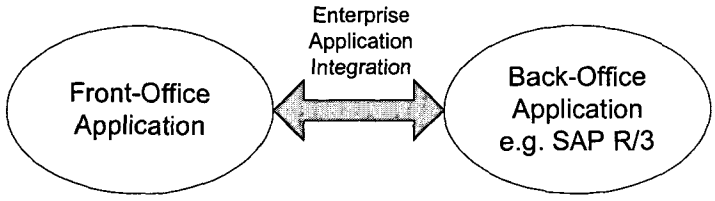


Figure 4.9: Interaction between Front- and Back-Office Applications

own initiatives would not address the market fast enough, and have therefore acquired e-business application vendors. Only one of these vendors, the market leader Siebel Systems, remains independent. Due to the rapid changing software market, only solutions from the key players SAP and Siebel are discussed in this chapter.

4.3.5.1 mySAP.com

The mySAP.com e-business platform is a family of solutions and services that empowers organizations to collaborate successfully. SAP's integration strategy is based on the assumption that the customer possesses the R/3 system. No other e-business vendor demands this prerequisite step. But the SAP R/3 component is an important building block of the mySAP.com e-business platform. SAP R/3 provides the ERP functionality that is a prerequisite for any successful business solution. The next version of SAP R/3 - SAP R/3 Enterprise - not only continues SAP's flagship ERP software, but also extends its functionality using a newly designed open technical platform that ensures an easy transition to e-business.

Along with SAP R/3 Enterprise, other software components from SAP, such as mySAP SCM or mySAP CRM, provides the technical building blocks of the mySAP.com e-business platform (Figure 4.10). According SAP, the

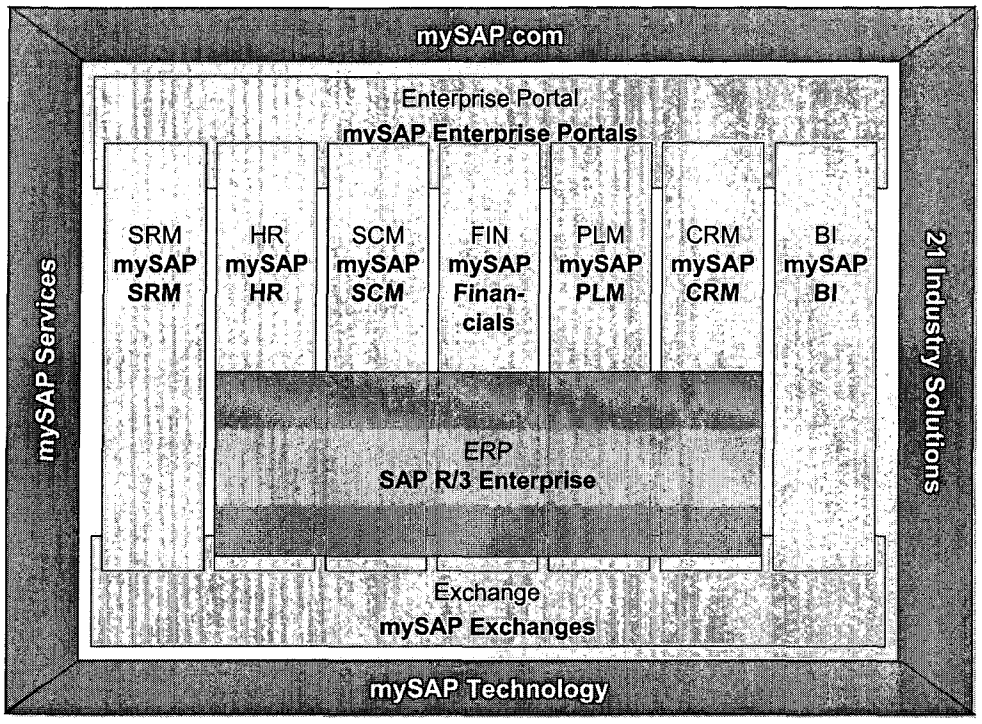


Figure 4.10: Architecture of mySAP.com [SAP02]

mySAP.com e-business platform provides the open infrastructure necessary for a seamlessly integrated suite of applications to perform in complex networked environments. [SAP02]

4.3.5.2 Siebel CRM Solutions

Siebel Systems is CRM (customer relationship management) market leader, and therefore a leading provider of e-business applications software, enabling corporations to sell to, market to, and serve customers across multiple channels and lines of business. With more than 3,500 customers worldwide, Siebel Systems provides organizations with a proven set of industry-specific CRM applications and business processes, empowering them to consistently deliver

superior customer experiences and establish more profitable customer relationships.

Siebel's integration solution is called Siebel eBusiness Application Integration (Siebel eAI), which provides interfacing for a number of other applications, including back-office applications such as SAP R/3 (Figure 4.11).

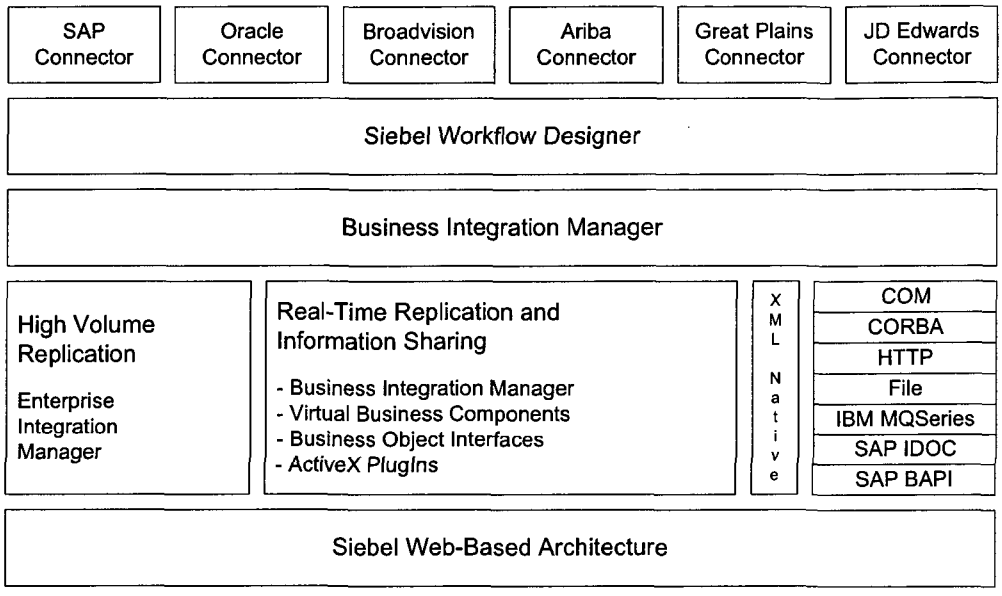


Figure 4.11: Siebel eBusiness Application Integration [Güm00]

The differences between Siebel's solution and the approach of SAP are: [Güm00]

- Like SAP, Siebel delivers the integration framework. Customers do not require a license for EAI-software, as they do with other vendors. Siebel charges a modest license fee for Siebel eBusiness Connector for SAP R/3 (approximately $10,000 per server).
- Unlike SAP, Siebel explicitly addresses the use of EAI-middleware. Users who have already installed EAI-software, or who are planning to use EAI,

can extend their integration strategy.

- Unlike SAP and other e-business applications vendors, Siebel prepackages the eAI for a number of target systems to achieve higher installation productivity. The number of target systems addressed is greater than that provided by other vendors.

4.3.5.3 Microsoft Business Solutions

Presently Microsoft is the world's largest software company (2002 turnover: 25,6 billion $) - followed by IBM (13,1 billion $), Oracle (6,9 billion $), and SAP (6,8 billion $) [iX 03b] - controlling more than 90 percent of the desktop operating system and business application markets. Because of both the consumer and business markets for desktop applications are saturated, one way for Microsoft to grow is to get into markets where it hasn't been competing, such as enterprise applications.

The acquisition of Great Plains Software and Navision was the first step of Microsoft's long-planned move into the terrain enterprise applications dominated by such industry giants as Oracle, SAP, Siebel Systems, and PeopleSoft.

Microsoft intention is to lead and dominate the midmarket for business applications by selling anything an enterprise would need to run their business. According to Microsoft CEO Steve Ballmer, "the biggest part of the computer market is not the enterprise or the consumer market. It's the small and medium size businesses." [Alo03]

As Oracle, Microsoft's root business is IT infrastructure, such as databases and development tools. Selling on top running applications fuels new sales of infrastructure products. That means every time Microsoft makes an enterprise application sale, it sells more copies of products like Windows operating sys-

tems and the SQL Server database, which is used by all of Microsoft's Business Solutions. Therefore, Microsoft is in a position to offer their business solutions very low priced.

4.4 Classification of the Dissertation

The aim of this chapter is to position the dissertation opposite to the current state of the art - particularly to **highlight the newness of this work**.

With the triumphal procession of the Internet during the 1990s, a veritable hype about terms like e-business, e-commerce, or business-to-business has been started. Software vendors have promised ultimate solutions to customers solving their problems. Companies such as Forrester Research forecast exorbitant optimistic figures: electronic commerce will reach as high as $3.2 trillion in 2003 [Sha00].

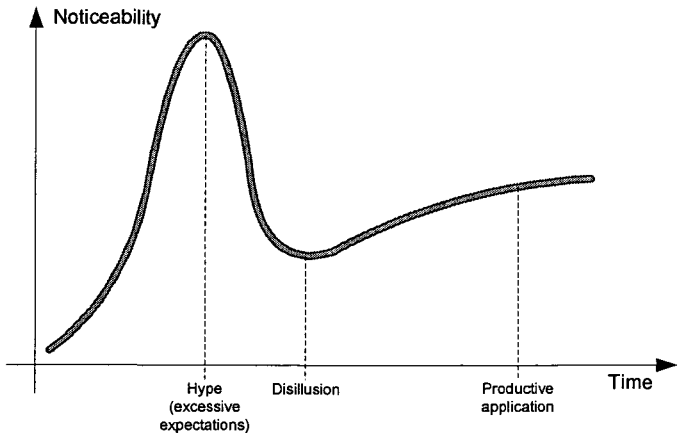


Figure 4.12: Hype Cycle by Gartner [DS03]

Few years ago, the hype has gone because of too high visions could not be realized as fast as was promised by the vendors. Additional reasons for this

development were inexperienced enterpriser or even frauds (".com crash"), and immature technology. Nowadays, the end of the collaboration hype is identifiable, pictured as point "disillusion" in Figure 4.12 of the hype cycle originated by the Gartner Group. This appraisement is confirmed by the business process report 2003 rendered by IDS, where the preponderant majority of the 150 interviewed companies want to focus more on internal business processes opposite to the external processes including integration with suppliers, customers, and partners [IX-03a].

Regardless of the suffered setback, Internet-related technologies and systems will initiate a revolution for businesses, comparable with the consequences of the letterpress for the book industry. With the knowledge of that enormous economic potential for the software industry, all big IT players, such as Microsoft, IBM, or Sun, invest a lot of money and huge effort in developing more mature technologies and applications. On account of the turbulent and rapid developments, today designed newly concepts or solutions can be become state-of-the-art very quickly.

In contrast to software vendors, which tend toward developing vendor-specific solutions for their customers, **the scientific community is able to develop open, well thought out concepts without direct market pressure by customers and competitors**. A very demonstrative example for the use of specifically solutions is the automobile industry where numberless suppliers have to invest in expensive point-to-point integration applications to their big customers because of there is no common collaboration concept available up to now.

The presented approach in this dissertation puts the presently isolated integration islands of companies behind one, with the aim to enable the totally integrated enterprise.

In the literature, there are approaches and technologies either for vertical integration inside organizations, or for horizontal integration between enterprises. First usable technologies enabling vertical integration are gathered under the term enterprise application integration (see chapter 4.1). The aspects of horizontal integration are dealt within many different theoretical concepts named enterprise networking, business-to-business, or extended enterprise (discussed in chapter 3.2). The most important and well-known research project in the field of virtual enterprises was the project called PRODNET II. In this project, a supplementary cooperation layer was introduced which is responsible for supporting all interactions between a company and its virtual enterprise partners [CMA99a]. Therefore, during this research project only horizontal integration between companies has been considered.

For the first time, the developed holistic approach handles both integration components - vertical and horizontal integration. The key enabler for such an approach is a business process engine processing the intra-/inter-organizational business processes inside extended enterprises. The business processes take center stage and are taken out of the several business systems. The new control engineering perspective, which sees the business process engine controlling all operations inside the extended enterprise, brings out the importance of this component (chapter 5.3).

The main part of the business process engine takes on by a workflow engine. Workflow management systems are a key technology for improving the efficiency of business processes. But this automation is limited to only one organization, because nowadays workflow management systems focus on processes which are circumscribed by the bounds of an organization [vdA98]. **Therefore, it is necessary to enhance an existing workflow engine in such a way that it is possible to handle complex, enterprise-spanning**

business processes. One additional success factor for an applicable business process engine is the use of an open, standardized business process description language. Compared to other initiatives such as [jBp03], this work does not use a proprietary developed language.

Over the last years, XML-based technologies have been established as de facto standards enabling open, vendor- and platform-independent solutions. Especially Web services are very promising to solve still existing integration problems. Due to the short lifetime of this new technology (Figure 4.13), **the developed system in this dissertation is the first implemented, integrated Web service-based software architecture to automate and integrate business processes inside extended enterprises.** The Web service stack shown in Figure 4.13 includes - besides the basic technologies SOAP, WSDL, and UDDI - additional standards for security and business process description.

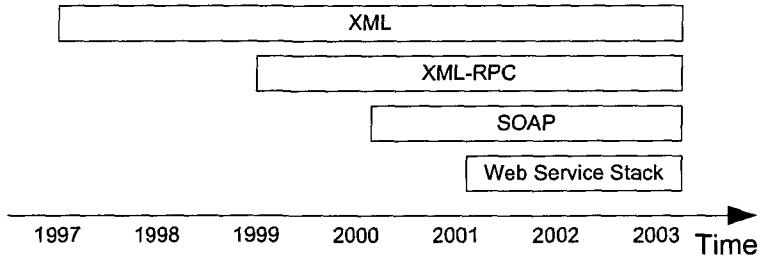


Figure 4.13: History of Web Services

The support of the whole product life cycle right from the start is a very important, innovative facet of this work - compared to existing business systems that automate only parts of the life cycle. Of course, software vendors try to enrich their existing systems with additionally functionality. Therefore, a rapid convergence between different systems is happened. For example, the ERP vendors are rushing to add more sophisti-

cated SCM functionality to their ERP products. And the SCM vendors are also expanding their functionality, further encroaching on the area inhabited by the ERP vendors. It seems that all the leading SCM vendors are partnered with the all the leading ERP vendors [Tex00]. But in most cases, this enlargement is done by integrating purchased systems as "plug-in" with the consequence of emerged "dinosaur-systems", which become too complex for most of organizations, especially for small and medium sized enterprises.

The counteragent is an open, component-based, service-oriented system architecture as designed in this work. This approach enables very simple "plug-and-play" of additional, needed business functionality into the software system, that is available within one enterprise and inside extended enterprises at the same time. That easy way of extensibility ensures the flexible, adaptable support of the product life cycle.

As a result of the nowadays scattered business system landscape inside enterprises, human users have to work with many separated systems to do their daily work. This cause in inefficient flow of work and inhomogeneous working environments for the users. **In this work, this drawback is removed by the concept of "single entry-point", where human users are supported with all needed information and business processes that are provided by both role-based presentation logic and business process engine.**

Chapter 5

New Approach enabling Extended Enterprise Collaboration

Due to fast growing market and technology development, enterprise internal information technology support, business process automation and integration, and intensive collaboration with other enterprises over Internet are the key success factors to be competitive in today's turbulent markets. To facilitate that there is an enormous need for an open, flexible, and holistic approach for the next generation business information system. In order to guarantee a widespread acceptance of the developed approach, and to avoid "a sea with vendor-specific islands" that are unable to interact among each other, the software system has to build up on international established standards.

In this chapter, after the extended enterprise requirements for an information technology system are discussed, the vision for the next generation extended enterprise enabling business system is presented. This vision consid-

ers the support of the product life cycle, integration of business information systems and business processes, and security. Because of this dissertation can't deal with all points of this extended enterprise vision, one focus of this work lies on the integration and automation of business processes inside the extended enterprise.

The main parts of this chapter are the presentation of the new and innovative approach enabling most flexible integration and automation of enterprise-spanning business processes in extended enterprises, and the discussion of the developed information technology architecture to implement the new, component-based software system. The core component of this system is the so-called business process engine to enable the execution and management of the enterprise-spanning business processes.

5.1 Extended Enterprise Requirements

Rapid technology (r)evolution and increasing competition in global markets put high pressure on companies to reduce costs, reduce time to market, improve quality and to speed up innovation for complex products and services. Nowadays, most companies have to act in a customer-driven, turbulent market which makes fast reactions a must to survive.

These fast reactions can be achieved by using information technology to support and connect specific tasks and processes, like product development, logistics and production, and networking with other companies in an extended enterprise. Extended enterprises are *"temporary or permanent networks of independent enterprises including a coordinator cooperating with the aim to design, manufacture, and sell a product or service in a project-oriented way independent of enterprise borderlines, automated inter-enterprise communi-*

cation through the usage of information technology, and communicating via Internet-related technologies like XML-based Web services" [Für02].

An example of an extended enterprise with the goal to deliver a mechanical component to its customer is shown in Figure 5.1. The extended enterprise consists of the enterprises which take on the following roles according their core-competencies: management, design and engineering, production, and sales. Due to this aggregation of different core-competencies, the whole product life cycle can be better supported by such an extended enterprise compared to the individual companies. Members of the extended enterprise are also the suppliers with the core-competence to deliver specific raw materials and semi-finished goods. A tight integration of the customer is needed, if the product or service has to be customized to the customers needs.

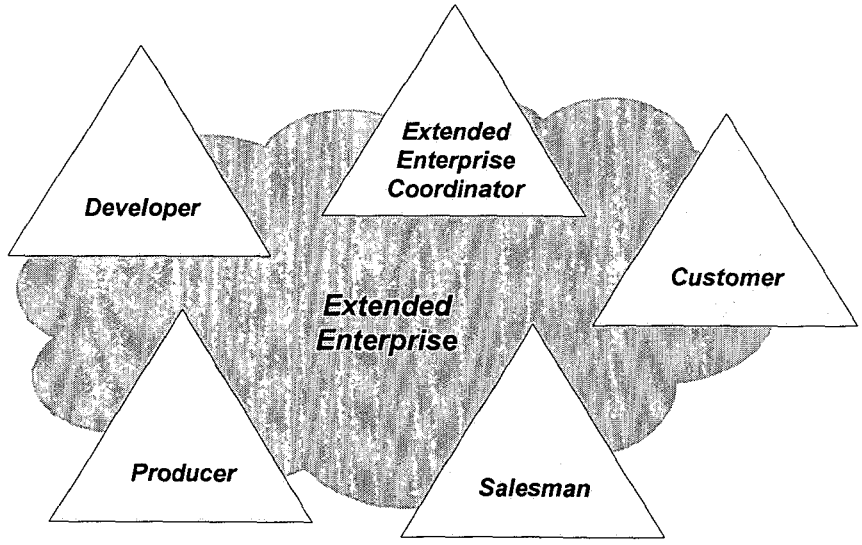


Figure 5.1: Principle of Extended Enterprises

The figure shows the high integration between the participating enterprises, including the customer. All processes are networked to achieve the common goal to deliver a specific product or service to the customer(s). The extended

enterprise coordinator is the responsible person for the whole project.

To work together in a consortium to develop a product is not a new concept. It is reality in nowadays business as Table 5.1 shows the estimated figures for important development attributes of five products. The figures for the size of external and internal development teams show, that the external development teams are noticeable bigger than the internal teams for some products.

	Stanley Tools Jobmaster Screwdriver	Roller-blade In-Line Skate	Hewlett-Packard DeskJet Printer	Volkswagen New Beetle Automobile	Boing 777 Airplane
Dev. Time	1 year	2 years	1.5 years	3.5 years	4.5 years
Int. team	3 people	5 people	100 people	800 people	6,800 people
Ext. team	3 people	10 people	75 people	800 people	10,000 people
Dev. cost	$150,000	$750,000	$50 million	$400 million	$3 billion

Table 5.1: Estimated Figures for Attributes of five Products [UE00]

Whereas the need to work together for developing, producing, and selling new products is growing, also the problems of doing this lead to major problems. Some of them are:

- High efforts to coordinate the common product development process
- Distributed design artifacts
- Redundant information storage
- Manual interaction between the manufacturing partners

Derived from the problems and needs of extended enterprises, there are five basic requirements for information technology systems:

5.1.1 Functionality

Currently, many different information systems are used to automate and support the information flow during the whole product life cycle inside one enterprise. The following four are the main ones:

- Product data management (PDM, see chapter 3.1.1) systems for product development,
- Enterprise resource planning (ERP, see chapter 3.1.2) systems for production and assembly,
- Supply chain management (SCM, see chapter 3.1.3) systems for external logistics, and
- Customer relationship management (CRM, see chapter 3.1.4) systems for product and service marketing.

All these systems had been implemented to organize, support, and automate internal business processes of one enterprise. Only within the supply chain management systems, logistics focused automation of enterprise-spanning business processes are realized. But to enable highly flexible extended enterprises, the distributed business processes for the product/service life cycle has to be supported. The need for very high flexibility stems from the volatility of the extended enterprise business process itself. Where the internal structures of an enterprise evolve normally very slowly, participants of extended enterprises can change very quickly. These quick changes in participants, process-steps, and organization structure lead to a very high volatility of distributed business processes inside an extended enterprise.

5.1.2 Integration

Efficiency gains have been achieved by the successful implementation of the systems mentioned above. However, companies of today strive to streamline their business processes (intra- as well as inter-organizational). This gives rise to a strong need for much better integration of systems - existing as well as new in-house developed systems or packaged ERP solutions - across functional boundaries and in line with the business processes. There are two different forms of integration (Figure 5.2):

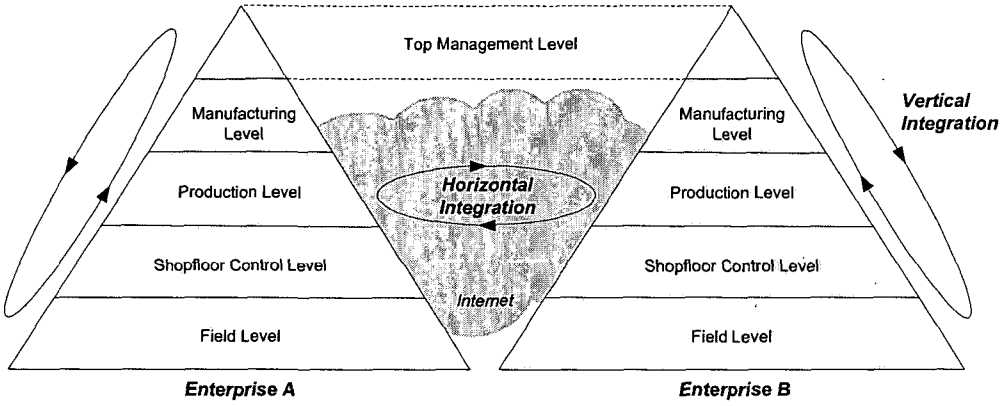


Figure 5.2: Horizontal and Vertical Integration (according to [ZF00])

- between systems on different control and management levels of an organization (vertical integration).
- between systems of different organizations using the Internet (horizontal integration).

The main problem of vertical integration is to create a kind of system integration capable of handling internal business processes which are spanning more than one enterprise information system, running on different enterprise

levels. *"Vertical information chains and the totality of all contributions from different levels are the core competences for successful dynamic industrial processes."* [ZF00] Moreover for the enterprise success factors - the right product at the right time and the right price - the inter-level control loops are absolutely essential.

Especially for all kinds of e-business applications (B2C, B2B), the horizontal integration of internal systems and the outer world (customer, business partner) is a must in the today turbulent market environment to reach a common, complex goal. Inside extended enterprises there has to be - beside internal enterprise business processes - extensive enterprise-spanning business processes each spanning a number of systems or business processes. During the process execution, the specific process activities inside the extended enterprise interact with the enterprise business processes of each enterprise (Figure 5.3).

As a matter of course, any enterprise can be a member in multiple extended enterprises - two enterprises might simultaneously cooperate in one business segment and compete in another. For example, in Figure 5.3 enterprise A participates the extended enterprises 1 and 2. Therefore, enterprise A has to integrate their internal business processes with both extended enterprise business processes to enable a barrier-free horizontal integration with the participating business partners.

5.1.3 Usability

Usability means easy and effortless access to services of an enterprise for internal and external human users. Many users have to access more than one system to do their work. They have to learn the usage of all systems they need, which can sum up to enormous overheads for even simple tasks [FRZ01]. Figure 5.4 shows three different users which collect manually the needed in-

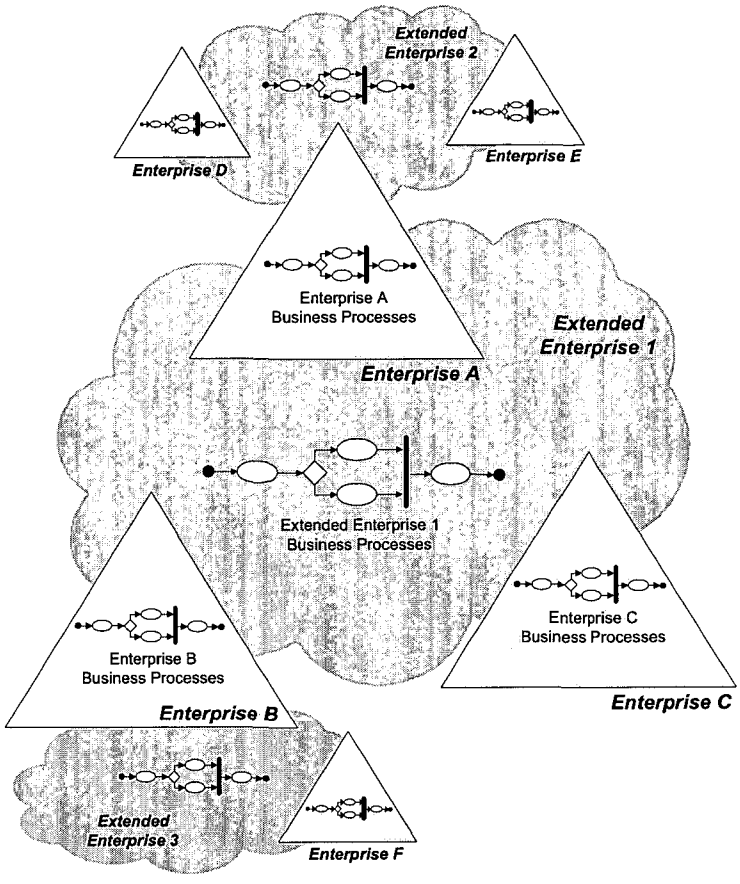


Figure 5.3: Horizontal Integration of Intra-/Inter-Business Processes

formation from the existing business information systems. For example, the salesman needs detailed information about the customers (CRM-system), features of the offered products (PDM-system), and the current status of the production capacity and possible delays of delivery dates (ERP-system) to satisfy the customer requests.

One answer to the latter problem is the simplification and customization of user-interfaces, but this will not satisfy the need to access many systems. In this case, users are trained to support systems processes. In a more efficient and

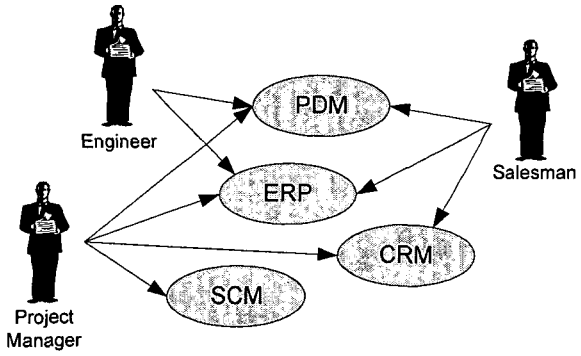


Figure 5.4: Manual Access to different Business Information Systems

user-friendly environment, systems should be trained to support user processes [FRZ01].

5.1.4 Security

Security is a very important and necessary assurance requirement to consider when deploying a system inside the Internet environment. Security mechanisms provide the following services:

- Integrity Verification: The beneficiary has some verification that the data or application is accurate.
- Confidentiality Preservation: The beneficiary has some assurance that confidentiality of the data or application has been preserved.
- Authorization/Access Permission: Properly authorized permission to access security- critical data, application, or other resource has been provided.
- Identity Verification: The beneficiary has some verification that the identity associated with the data or application source is who they say they

are.

- Authenticity Permission: Properly authenticated permission to access some data, application, or other resource has been provided.
- Availability of Service: The beneficiary has access to the data, application, or other resource when it expects to have access.
- Non-repudiation Evidence: The beneficiary has some evidence regarding from whom the data or application comes or has some evidence that an intended receiver actually received data sent to it.
- Auditing Evidence: The beneficiary has some evidence of security-critical operations that take place in the system.

5.1.5 Flexibility

In a stable environment the above-mentioned requirements could be fulfilled by ongoing enhancing the information technology systems in order to reach an optimal solution. But in reality, this is not possible due to additional system requirements, which are caused by an evolvement of basic technologies and changes in business processes. This is especially true with extended enterprises. Change is inherent within these enterprise networks, which lead to the demand of highly flexible, performant and easy-to-manage systems. It is very important to be able to fit the system easily into new requirements and processes in order to keep it interesting for both parties (buying and selling) to do business together [FRZ01].

5.2 Integration Concept for Knowledge-driven Industry

Taking into account the discussed requirements in chapter 5.1, a holistic vision for extended enterprises has been developed (shown in Figure 5.5). The common goal of the extended enterprise members is the support of the whole product life cycle composed of product development, logistics, process planning, production, marketing and sales, etc.

As already mentioned in chapter 3.1, there is the need for several information systems to automate the particular life cycle steps. Different from isolated enterprises, the supporting information system functionalities are widespread in all participating enterprises. Therefore, a very extensive horizontal integration between these enterprises is necessary so that each enterprise is able to contribute their part to the information systems inside the extended enterprise.

All business processes that supports the product life cycle - summarized into the term "extended enterprise business processes" - interacts with the enterprise business processes of the respective company. One precondition for such a closely horizontal integration is the existence of a vertical integration inside the company to provide all internal business processes and business system functionalities to the business partners.

Another very important ingredient for a successful software system enabling extended enterprises is security, particularly if the Internet is used as the communication technology. For that purpose, in Figure 5.5 each participating company has been enveloped with a security layer to guarantee a secure information exchange, authorization, authentication, and other security mechanisms. One enterprise, most the extended enterprise coordinator, has to do the extended enterprise administration including choosing of the business partners and assigning partners to projects and roles.

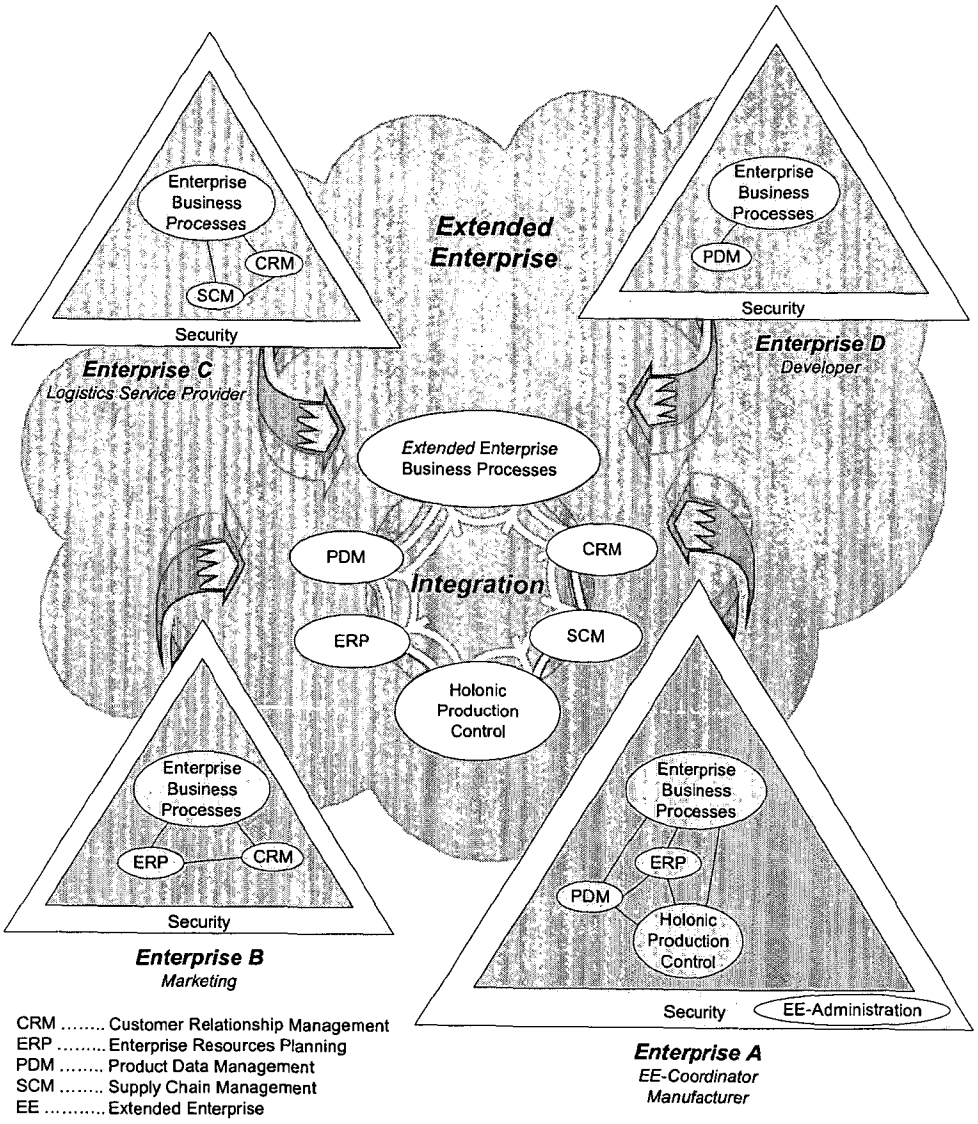


Figure 5.5: Extended Enterprise Integration Concept

From outside, the extended enterprise appears like one enterprise which is able to develop, produce, and sell complex products and services to the market. But, each enterprise inside the extended enterprise fulfills specific roles (e.g. project manager, engineer, salesman, etc.) that usually represents

their core-competencies. In many cases, one enterprise takes different roles in one or more extended enterprises. This is one reason why the automation of the business processes is essential for a successful, flexible extended enterprise.

The exemplary extended enterprise (Figure 5.5) consists of four independent enterprises, which are - beside extended enterprise coordination - responsible for development, production, logistics, and marketing of products and services. Each enterprise has its own internal business processes and information systems to support these processes. The internal business processes in conjunction with their customized IT-support represent the specific core know-how of the different enterprises.

Based on the presented vision, three new building blocks for extended enterprises can be detected:

1. Business process automation and integration supporting the whole product life cycle inside the extended enterprise.
2. Integrated, consistent, and standardized information flow between the business information systems and the holonic production control level.
3. Security concept for extended enterprises to enable the secure exchange and usage of confidential business information.

The aim of this dissertation is the development of a method of resolution for point (1), which will be discussed in chapter 5.3 in detail. The points (2) and (3) are discussed sketchily in the following chapters to give a compact overview about the additional research areas regarding extended enterprise at the Automation Control Institute of the Vienna University of Technology.

ad (2) Customers' desires of customized products force enterprises to strive economic lot size one production. This ambitious objective requires high flex-

ible production systems as well as tight integration with business information systems. Shop floor services, as shown in Figure 5.6 and presented in [GSWF03], marries both concepts that of holons and that of Web services (chapter 3.5).

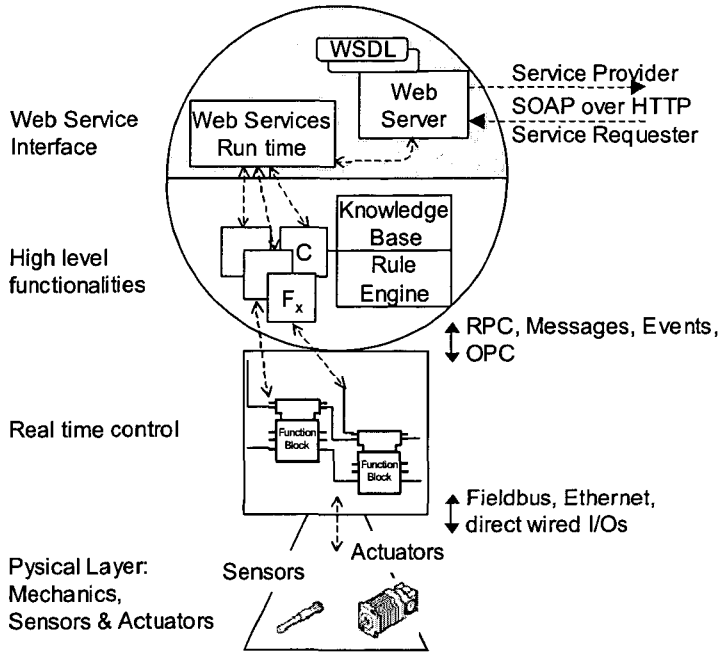


Figure 5.6: Shop Floor Service Architecture [GSWF03]

A holonic manufacturing system (HMS) is a production system having cooperative and autonomous characteristics. These characteristics are a consequence of the system's building components called holons. Holons are self-assertive and integrative. The introduced shop floor services should keep the autonomy and smartness of holons and adds value by interfacing these with a new consolidating standard for distributed applications based on Internet technologies. By providing this services to the upper levels of an enterprise, it will be possible to integrate the distributed, holonic automation devices into the extended enterprise network. At the Automation Control Institute,

there is a real testbed for holonic distributed control systems ("Odo Struger Lab") available, which is included in the proof-of-concept prototype described in chapter 7.

ad (3) To foster collaboration between business partners, secure and flexible Internet-based communication is essential. In [FSW02b], the developed approach, called distributed role-based access control for extended enterprises (DRBAC-EE), focuses on authentication and authorization in extended enterprises, which have been identified as very important issues to be solved, as it is still missing in the research community. DRBAC-EE is based on the concept of RBAC (role-based access control) introducing a supplementary abstraction layer, the so-called extended enterprise roles (Figure 5.7).

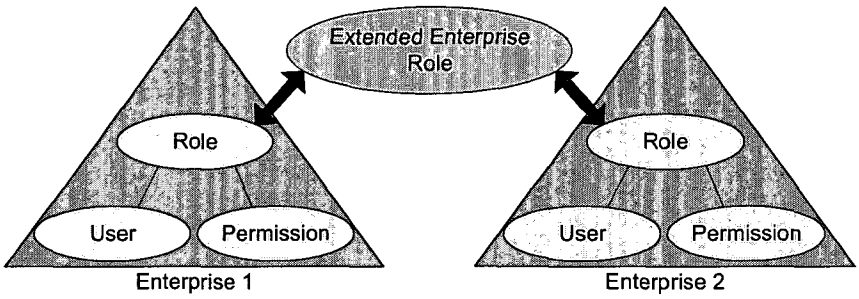


Figure 5.7: Role Mapping inside Extended Enterprise

Therefore, a role mapping to translate enterprise-internal into extended enterprise roles becomes necessary. Each internal role is assigned to user(s) and permission(s). The central part of this approach is the "access management" component. The main responsibilities are privilege verification for internal and extended enterprise roles, role-mapping, and logging for documentation, data mining, and legal actions. Additionally, an EE-administration component is essential for easy administration of the business network that could be assembled by hundreds of independent companies.

5.3 Fundamental Conception for Problem Solving

In consideration of the extended enterprise requirements discussed in chapter 5.1 and the developed vision presented in chapter 5.2, the innovative, holistic conception for business process automation and integration inside extended enterprise has been designed (Figure 5.8).

As Figure 5.8 shows, the conceived approach is based on services, which means that each functionality inside the (extended) enterprise is available as a service. The main components of this concept are the so-called business process engines which have to execute either the enterprise business processes inside one enterprise (enterprise business process engine), or the extended enterprise business processes inside the extended enterprise (extended enterprise business process engine). Due to the business processes itself are provided as services, the essential horizontal integration can be achieved. To fulfill the activities of a process, services provided by enterprise information systems or other business processes are used. The end users, which take in several, project-depending roles, work with a role-based presentation that uses also services provided by the business process engines.

To systematize the discussion about the presented approach, the features of this new concept are confronted with the extended enterprise requirements discussed in chapter 5.1:

- Functionality: The demand for this requirement is the automation and support of the information flow during the whole product life cycle. As a result of the service orientation, where enterprise information system has to provide their business functionalities as standardized services to the environment, this need can be satisfied. Thereby, all kind of software systems such as in-house solutions, packaged software like ERP or PDM,

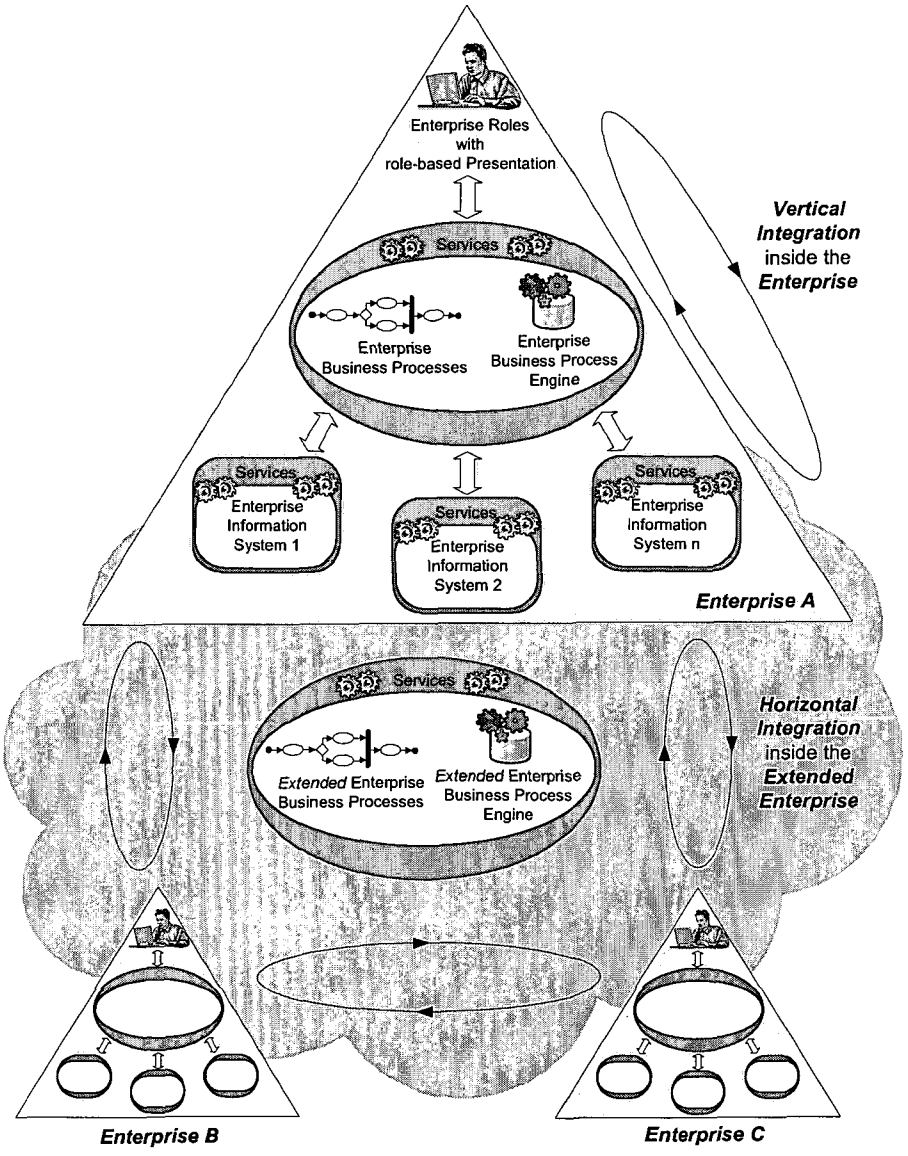


Figure 5.8: Basically Approach for Extended Enterprise Collaboration

or new developed components can be used by the business processes to support the product life cycle.

- Integration: Companies of today have a strong need for a barrier-free

integration of systems across functional boundaries and in line with the intra- and inter-organizational business processes. The basis for the vertical integration inside one enterprise are the enterprise business processes executed by the enterprise business process engine. Each activity of a process calls one or more services provided by the existing enterprise information systems. Because of the business processes are also offer their functionality as services, a universal cooperation methodology can be established - starting at the end user via the business processes down to the business systems, independent the enterprise level. The horizontal integration is done by the extended enterprise business process engine processing the extended enterprise business processes, which interact with the intra-organizational services from the processes and enterprise information systems.

- Usability: That means easy and effortless access to services of an enterprise for internal and external human users. In the approach shown in Figure 5.8, the different roles - finally the human users - are supported in doing their daily work with a role-based presentation of the necessary information and business processes. Realizing only one working environment for the human user, the exhausting and time-consuming data-collection from independent software systems has an end. In addition to improved working conditions, the job efficiency will be enhanced and the school enrolment for the employees will be less expensive.

- Security: This requirement is out of scope of the work described in this dissertation, and therefore not considered in the developed approach.

- Flexibility: Especially in extended enterprises, the demand for flexibility is very high. Reasons for that are e.g. changing of business partners, redefining of business processes, etc. Because of the service-oriented con-

cept, it is possible to realize a "plug-and-play" collaboration between the business partners. The redefinition of the business processes can be facilitated in a very easy, flexible way, if inside the extended enterprise services will be added or removed.

Figure 5.9 emphasizes the meaningfulness of the business process engine for next generation extended enterprises. In the shown closed loop, the business process engine is the controller for the extended enterprise processes, which are influenced by different kind of disturbances such as competitors or economic conditions. The plant of this control loop is the market, in that the extended enterprise delivers services and products to customers and business partners.

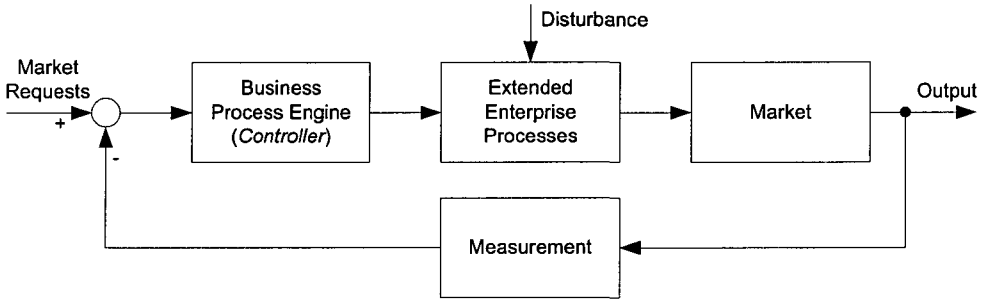


Figure 5.9: Qualitative Illustration of Controlled Extended Enterprise

The set-point of the closed loop are the market requests such as customer satisfaction, time-to-market, time-to-money, enterprise value, etc. These requests have to be compared with the measured output of the control loop. The measurement has to be done inside the feedback loop.

Chapter 6

Technical Specification of the Software System

To cast the new and innovative approach (see chapter 5), which enables most flexible integration and automation of enterprise-spanning business processes in extended enterprises, into software, in this chapter the technical specification for the software system is discussed.

At the beginning, the developed system architecture is described tier by tier to illustrate the fulfillment of system requirements. The design and tasks of each tier, and the cooperation between them are presented in detail.

The most important component of the system architecture - the business process engine - is discussed more precisely. The basis for such an engine has to be an open, expandable workflow engine. The main points argued for an open source engine are mentioned. Due to many different open source workflow engines are available today, the most apposite engine has to be selected based on decision criteria. The selection process and the characteristics of the selected engine are presented in this chapter. Finally, the deployment of the

business process engine is described.

6.1 System Architecture

Due to the characteristic of extended enterprises, the system is going to be a distributed software system. To facilitate that, the designed system architecture shown in Figure 6.1 is multi-tiered, service-based, role-based, and modular. Each tier can only make use of the functionality of the tier below it.

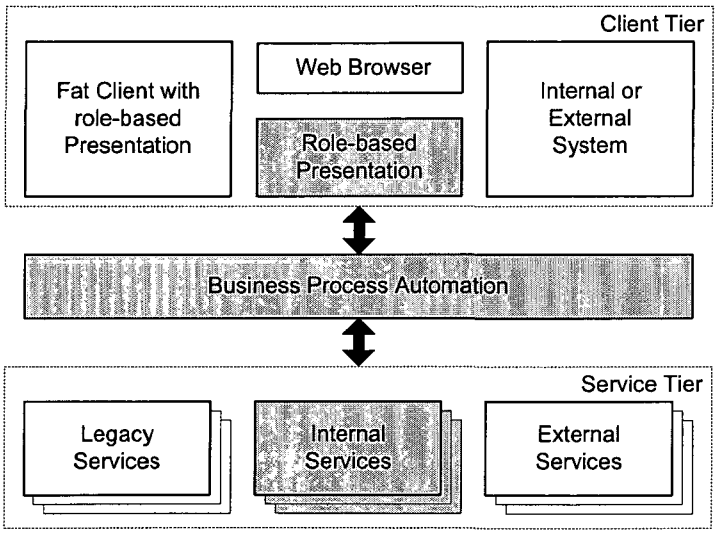


Figure 6.1: IT Architecture

The gray-marked boxes in Figure 6.1 use J2EE (Java 2 Platform, Enterprise Edition), which is a platform supporting the development and deployment of distributed, multi-tiered enterprise applications. This architecture can be subdivided into four tiers: client tier, role-based presentation tier, business process automation tier, and service tier.

6.1.1 Client Tier

The client tier is responsible for presenting data and interacting with human users or other systems. Both could be enterprise internal or external clients. Possible client types are:

- Human user using a Web browser to interact with the system.
- Human user using a client application. This fat client application is implemented with Java or other programming language, and includes its own role-based presentation tier to format the received information in a role-specific way.
- Internal or external systems that use the functionality of the system. Internal systems are other in-house business systems. External systems are systems from business partners inside the extended enterprise.

6.1.2 Role-based Presentation Tier

In the case of a fat client, the role-based presentation tier is included in the client application. If the human user works with a standard Web browser, this tier is responsible for creating a role-specific Web presentation including session handling, personalization, etc. The communication technology between the role-based presentation tier and the Web browser is HTTPs. The implementation of this tier is based on J2EE using the technologies Java Server Pages (JSP) and Java Servlets. In this way J2EE components are realized that are running inside the Web container, which provides underlying services for such components.

The design pattern for this tier is the model-view-control (MVC) paradigm.The intention of the MVC architecture is to separate the business

logic and data of the application from the presentation of data to the user. Following is the small description of each of the components in the MVC architecture (Figure 6.2):

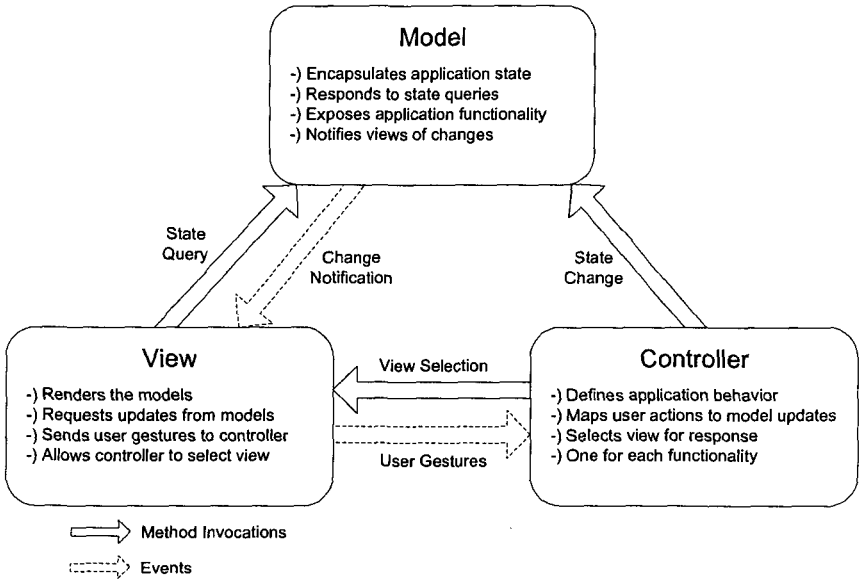


Figure 6.2: MVC Architecture [Sun03b]

- Model: The model represents the data of an application. Anything that an application will persist becomes a part of model. The model also defines the way of accessing this data (the business logic of application) for manipulation. It knows nothing about the way the data will be displayed by the application. It just provides service to access the data and modify it.

- View: The view represents the presentation of the application. The view queries the model for its content and renders it. The way the model will be rendered is defined by the view. The view is not dependent on data or application logic changes and remains same even if the business logic undergoes modification.

- Controller: All the user requests to the application go through the controller. The controller intercepts the requests from view and passes it to the model for appropriate action. Based on the result of the action on data, the controller directs the user to the subsequent view.

The usage of MVC has many benefits. Separating model from view (that is, separating data representation from presentation) makes it easy to add multiple data presentations for the same data, and facilitates adding new types of data presentation as technology develops. Model and view components can vary independently (except in their interface), enhancing maintainability, extensibility, and testability. Separating controller from view (application behavior from presentation) permits run-time selection of appropriate views based on workflow, user preferences, or model state. Separating controller from model (application behavior from data representation) allows configurable mapping of user actions on the controller to application functions on the model.

6.1.3 Business Process Automation Tier

This tier is responsible for the automation of both intra-organizational and enterprise-spanning business processes. Processes are realized as composition of feature rich business Web services, either provided by the service tier or by another enterprise's business process automation tier. This interaction is possible because of business processes inside the extended enterprise can be accesses via a Web service interface, too. As Figure 6.3 shows, the business process automation tier is the key enabler for the horizontal integration between business partners.

The main component of the business process automation tier is the business process engine. Consuming a standardized, XML-based business process defi-

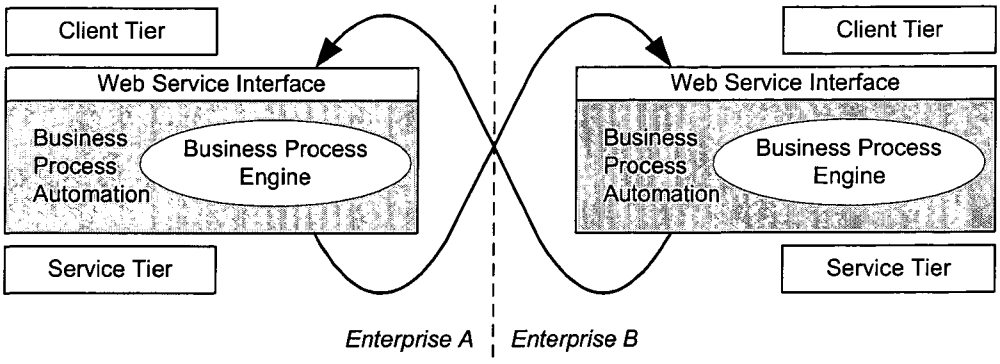


Figure 6.3: Business Process Automation Tier enables Horizontal Integration

nition the engine can walk through the business processes activity by activity utilizing the needed business services. Due to this process engine is the key building block of the system to enable business process automation inside the extended enterprise, a detailed discussion is necessary (done in chapter 6.2).

6.1.4 Service Tier

This tier provides all necessary, feature rich business services for the business process automation tier. The constituents for the service body are the standardized Web service interface representing the service access point, business logic realizing the service functionality, and persistence using in most cases a database (Figure 6.4).

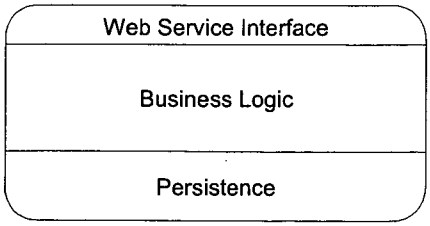


Figure 6.4: Service Body

The architecture differentiates three kinds of services:

- Legacy Services: Nowadays, the deployment of a software system without considering existing business systems is impossible, because there is no company in the world exclusive of information technology systems. Hence, it is a must to "plug-in" these external systems into this new architecture to make their business functionalities available in the (extended) enterprise and to facilitate vertical integration. This can be done by building a wrapper which has to make the business services available as platform-independent Web services, because they are supported by all well-known programming languages.

- Internal Services: These services are implemented directly as part of the architecture. They should provide additional business functionality that is not already available inside the (extended) enterprise. Examples for such services are document management service, production planning service, or procurement service. The building blocks for the realization are EJBs (Enterprise JavaBeans) which are running inside an EJB container. The communication between internal services and the business process automation tier (deployed in the same J2EE container) must not use only Web service technology because of performance reasons. If services are used within the same J2EE container they have to be handled more intelligently and optimally. Compared to Web service based communication, J2EE based interaction is much more efficiently. Therefore, such services have to provide both Web service and J2EE interfaces.

- External Services: From the very first, the developed architecture considers the collaboration feasibility between business partners. External services are the enablers because additionally to intra-organizational services (internal and legacy) all available services inside the extended en-

terprise can be utilized by the business process automation tier. External services can be accesses via a Web service interface.

6.2 The Key Component: Business Process Engine

The foundation for the required business process engine has to be an open, expandable, J2EE compatible workflow engine. Such an engine is a software service providing the runtime execution environment for workflows, which are *"the automation of business processes"* [All00]. To get a workflow engine achieved the mentioned requirements, an open source engine is the rights choice. Reasons for that are discussed in chaper 6.2.1. Since successful open source projects like Linux and Apache the demand for such products and the trustfulness of customers is increasing dramatically. As a consequence, there are a multiplicity of practical, open source workflow engines available nowadays. The selection process is discussed in chapter 6.2.2. Due to the automation of local, easy manageable processes is originally the application for conventional workflow engines, there is the need to enhance the selected workflow engine to an open, applicable business process engine for extended enterprises. This step is discussed in chapter 6.2.3.

6.2.1 Motivation for Open Source Engine

Open source software is an idea whose time has finally come. For twenty years it has been building momentum in the technical cultures that built the Internet and the World Wide Web. Now it's breaking out into the commercial world, and that's changing all the rules, e.g. have in mind the success of Linux or the Apache project.

The term "free software" has a dangerous ambiguity, due to "free" meaning

both "freedom" and "gratis". Open source software does not have to be gratis. The main features that characterize open source software is the freedom that users have to: [Wor03]

- Use the software as they wish, for whatever they wish, on as many computers as they wish, in any technically appropriate situation.
- Have the software at their disposal to fit it to their needs. Of course, this includes improving it, fixing its bugs, augmenting its functionality, and studying its operation.
- Redistribute the software to other users, who could themselves use it according to their own needs. This redistribution can be done for free, or at a charge, not fixed beforehand.

To satisfy those previous conditions, there is a fourth one which is basic, and is necessarily derived from them: Users of a piece of software must have access to its source code.

In the majority of cases, the first contact with open source software makes nearly every developer or IT manager skeptic. The main doubts are:

- How is the software quality ensured given that developers of unknown skills are contributing to the project?
- The source code is open to everyone couldn't that yield weakness in security?
- Are there projects that are stable and mature for application in productive systems?

Experiences in the open source community have shown that the rapid evolutionary process produces better software than the traditional closed model,

in which only a very few programmers can see the source. This is also valid regarding security, because security through obscurity just does not work. The reason is that security-breakers are lot more motivated and persistent than good guys. Supplementary there are a good deal more reasons for open source software: [Wir03]

- Contains exact the needed functionality because open source software is developed by end user for end users.
- Full control of the system functionality because the source code is visible.
- Utilizing open standards and platforms to avoid vendor-specific solutions.
- Rapid elimination of software bugs and security holes.
- Cost efficient acquirement due to very equatable licenses.

6.2.2 Workflow Engine as the Principal Component

6.2.2.1 Selection of appositely Workflow Engine

Before a workflow engine survey can be done, the question of how to find open source projects has to be tackled. The starting address to get information and different considerations around benefits and risks of open source developments is the website of opensource.org. It provides links to established open source development platforms like freshmeat.net or sourceforge.net. Of course, common Web search engines deliver also useful results.

The survey introduces seven open source projects around the topic of workflow management systems. A few of them have started recently and are therefore far away from being applied in productive systems. Table 6.2 gives a

compact overview of ongoing workflow engine projects. It allows a coarse evaluation in a few seconds. Note: "n/s" means "not specified", and ".../" in the row homepage stands for "http://sourceforge.net/projects/".

Table 6.1 shows the most important decision criteria to choose the the most qualified workflow engine. Considering these criteria, the open source engine OFBiz (Open For Business) is the right choice to be the foundation for the business process engine.

Criteria	**Target**
Maturity	High
Development Activities	High
Programming Language	Java
Application Server	J2EE
Process Definition Language	Standardized
Web Service Integration	Yes

Table 6.1: Most Important Decision Criteria

	Compiere	OBE	OpenFlow	OFBiz	WFToolkit	OpenSymphony	Powerfolder
Homepage	.../compiere	.../obe	.../openflow	.../ofbiz	.../wftk	.../opensymphony	.../powerfolder
License	Mozilla Public Lic. 1.1	Apache SW Lic.	GNU GPL	MIT Lic.	GNU GPL	Apache SW Lic.	Lesser GNU Lic.
Maturity	stable/running	mid	stable/running	stable/running	mid	stable/running	mid
Dev. Activities	high	permanent	permanent	high	low	high	mid
Programming Lang.	Java,PL/SQL	Java	Python,C	Java (Prolog)	C,Python	Java	Java
J2EE/.NET	yes	no	no	yes	no	yes	yes
Application Server	JBoss	no	Zope	any J2EE conform	Zope	any J2EE conform	any J2EE conform
EJB Integration	yes	no	no	yes	no	yes	yes
Web Container	Jetty	no	Zope	Tomcat,Jetty,...	Zope	Tomcat,Jetty,...	Tomcat
Process Def. Lang.	no	XPDL	WfMC-inspiered	XPDL	n/s	own	own
Web Service Integr.	need to be developed	no	n/s	yes	n/s	yes	n/s
Platform	Win2k,Linux,Solaris	independent	Linux,Win,MacOS	independent	Win,POSIX	independent	independent
Graph. Proc. Def.	process-def GUI	yes	yes	no	no	no	yes
Administrator UI	yes	API	yes	Web/API	API	Web/API	yes
User Interface	JavaSwing,Servlets,JSP	n/s	Web oriented	Servlets,JSP	CGI,Python	XSLT,JSP,...	Servlets,JSP
Invoking IT-Apps.	J2EE connectors	n/s	yes	yes	via adapters	n/s	SOAP
Logging/History	n/s	yes	yes	yes	n/s	yes	yes
Transactions	JBoss	no	yes	yes	n/s	yes	yes
Persistence	Oracle,PostgreSQL	n/s	Oracle,MySQL,...	Oracle,MySQL,...	ODBC,Oracle	any RDB	JDBC

Table 6.2: Comparison of Open Source Workflow Engines

6.2.2.2 Characteristics of selected Workflow Engine

OFBiz is a very mature implementation, which is compatible with J2EE based application servers. The objective of OFBiz is to make an environment available for realizing Web-based business applications. A fundamental requirement is to be as flexible as possible. It covers a workflow engine capable of reading and executing XML Process Definitions Language (XPDL) workflow definition files. XPDL allows features like invocation of subflows, structuring with inline blocks, AND- or XOR-splits, joins, loops and so on.

The OFBiz implementation is not EJB-based but it can use the concept of container managed persistent entity beans of several application servers and according transaction mechanisms. To understand OFBiz the breakdown into two topics is helpful: entity representation and service invocation. There is a so-called entity engine responsible for managing the access of data objects independent of the underlying method (EJB, JDBC) enabling a high variety of databases. The second part is the service engine that handles the invocation of services. Services could be Java methods, beanshell scripts, remote SOAP service calls or even workflows.

The workflow engine is built upon both the entity engine and the service engine. Workflow activities can call services, but also the workflow itself is treated as a service. Workflow definitions as well as data of running workflow instances are stored in the applied database. Figure 6.5 depicts the components and structure of the service engine. The central unit is the service dispatcher responsible for handling incoming service calls of different types and invoking the appropriate service. That can be simply invoking Java methods, beanshell scripts, activating workflows, or making SOAP calls to external services. The invocation of workflows is treated as OFBiz service calls and also if workflow activities require any kind of service it is done the same way using the service

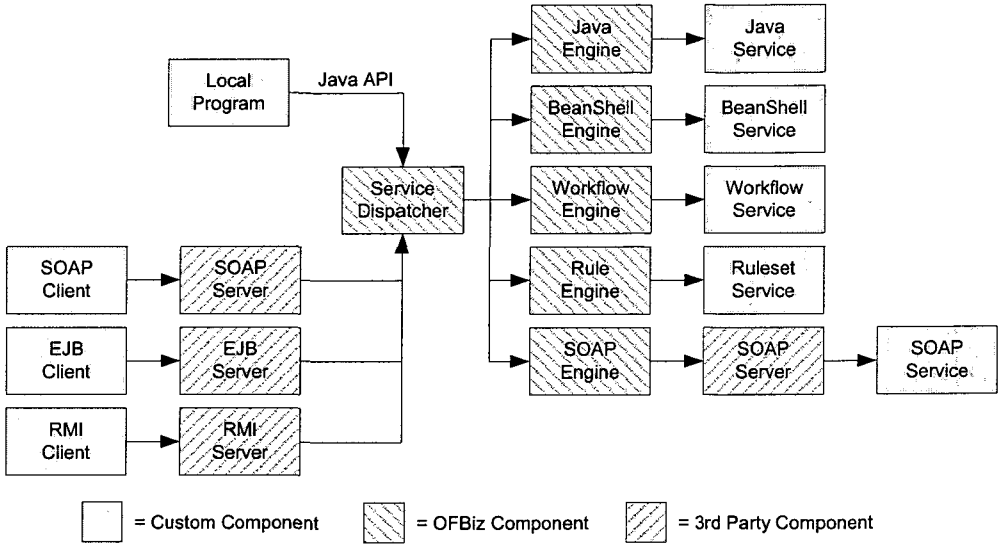


Figure 6.5: Service Engine Components of OFBiz [OFB03]

dispatcher.

6.2.3 Deployment of the Business Process Engine

As mentioned in chapter 6.2.2.2 the open source workflow engine OFBiz is not J2EE-based. Therefore, it is necessary to enhance the engine to provide both a Web service interface and a J2EE interface to different requestors, either inside or outside of the J2EE server (Figure 6.6). This is essential to fulfill the requirements for an extended enterprise system.

The implemented business process engine interface calls - according the requests - either the service dispatcher or the delegator of the OFBiz engine. The service dispatcher is used if a defined OFBiz service has to be invoked. The delegator, which can be also used by the OFBiz services, handles the database access inside OFBiz.

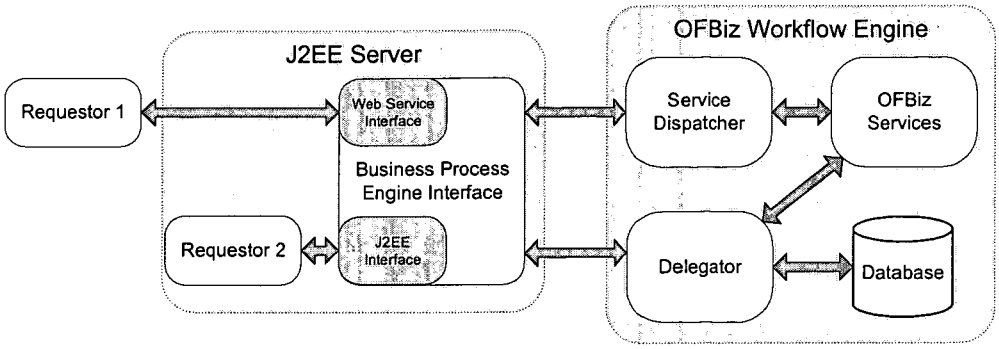


Figure 6.6: Business Process Engine Interface

The underlying standard for the business process engine interface is the Workflow Application Programming Interfaces (WAPI) by the Workflow Management Coalition (WfMC) [Wor98]. The purpose of this specification is to specify standard WAPIs to provide a consistent method of access to workflow management functions in cross-product workflow management engines. The WAPI standard covers the interfaces 2 and 3 of the WfMC's Workflow Management System Reference Model (see Figure 3.9).

By the realization of the business process engine interface, the system is independent from the used workflow engine and their special features. Therefore, other workflow engines - either open source or commercial ones - can be integrated into this flexible extended enterprise system to be the foundation for the essential business process engine.

Now back to the features of OFBiz, and how the business process engine interface collaborates with them. The following UML collaboration diagrams show two different scenarios: in the first case the requestor asks for the worklist (Figure 6.7), and in the second case the requestor completes a work item (Figure 6.8).

UML collaboration diagrams describe interactions among objects. These

interactions are modeled as exchanges of messages between objects through their associations. The messages are numbered, indicating the order in which they are sent (e.g. 1 followed by 1.1, 1.1.1, 1.2, and so on). The synchronous messages must be fulfilled before anything else can be accomplished (e.g. message 1.1.1 must be sent and accomplished before message 1.2 can be sent). Comprising, collaboration diagrams make it easy to express complex interactions and to show the relationships between the collaborating objects. [Alh98][EP00]

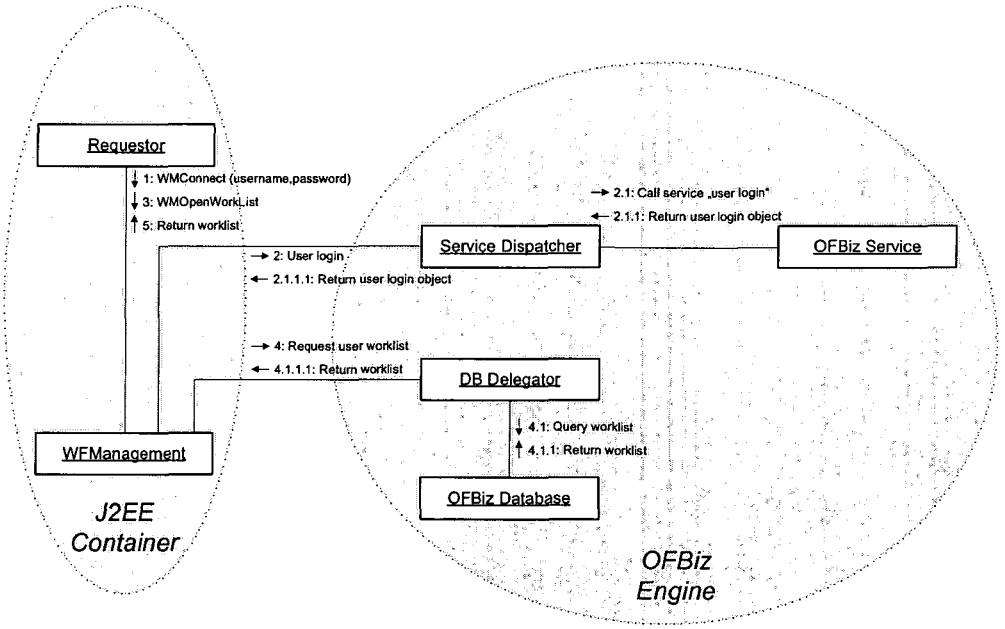


Figure 6.7: Collaboration Diagram to Get the Worklist

In the scenario in Figure 6.7, the requestor has to build up the connection to the workflow engine at first. This is done by calling the method "WMConnect" of the business process engine interface (labeled with "WFManagement") to submit the username and password (1). WFManagement calls the service dispatcher, which uses the service "user login", to login the user (2 and 2.1).

As a result, the user login object is returned (2.1.1 and 2.1.1.1). After the requestor has invoked the method "WMOpenWorkList" (3), the WFManagement requests the user worklist via the database delegator from the OFBiz database (4 and 4.1). Finally, the worklist is returned to the requestor (4.1.1, 4.1.1.1 and 5).

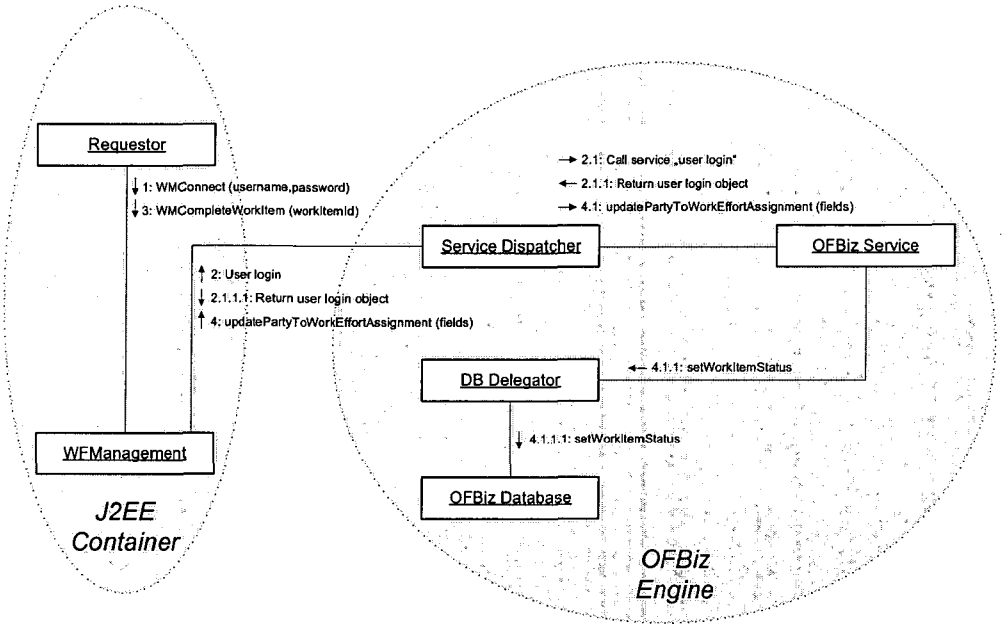


Figure 6.8: Collaboration Diagram to Complete the Work Item

In the second scenario (Figure 6.8), the requestor wants to set the status of a work item to complete. Therefore, he has to login to the workflow engine in the same way described in scenario 1. After invoking the method "WMCompleteWorkItem" (3), the WFManagement calls the predefined OFBiz service "updatePartyToWorkEffortAssignment" to change the status (4 and 4.1). The OFBiz service uses the database delegator to access the OFBiz database, and sets the status of the work item to complete (4.1.1 and 4.1.1.1).

To describe the structure of the system, the UML class diagram can be

used. In UML, classes are defined as a set of objects with a name, attributes, and operations. A class is drawn with a rectangle that is divided horizontally into three compartments. The top compartment contains the name of the class; the middle compartment contains the attributes of the class; and the bottom compartment contains the operations of the class. Attributes/operations with a plus sign (+) in front of them are public, meaning that other classes can access them. Attributes/operations preceded by the minus sign (-) are private, indicating that only the class itself and its objects can access them. The classes can be associated to each other via different kind of associations. [Alh98][EP00]

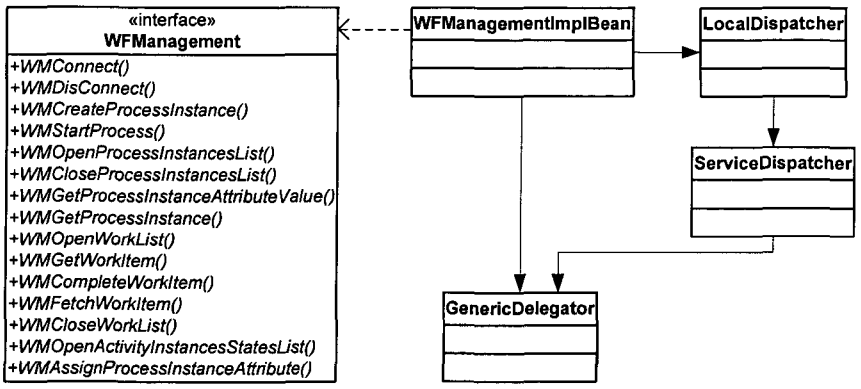


Figure 6.9: Class Diagram of Business Process Interface

The class diagram in Figure 6.9 shows the composition and cooperation between the used software classes to provide the business process interface. The interface "WFManagement" is implemented by the class "WFManagementImplBean" and provides all needed operations of the business process engine. The classes "LocalDispatcher", "ServiceDispatcher", and "GenericDelegator" are OFBiz internal classes (therefore the attributes and operations are not shown), that are associated by the class "WFManagementImplBean".

Chapter 7

Proof-of-Concept Prototype Realization

After detailed discussion about the approach enabling extended enterprise collaboration (chapter 5), and the presentation of the system architecture (chapter 6), it's time to cast the theoretical work into real software code - to prove the developed concept. The realization of this prototype has been done within the European research and development project "FLoCI-EE" (Flexible Low-Cost Internet Extended Enterprise; Growth: G1RD-CT-2000-00324). The duration of the project was from January 2001 until December 2003. The consortium was composed of seven partners (one university, four software developers, and three end users) form five countries (Austria, Germany, Greece, Netherlands, Spain). The entire project costs were more than 2.85 million euros, and the total work effort was nearly 31 man-years. The project initiator and coordinator was the Automation Control Institute.

7.1 Exemplary Business Case

7.1.1 Goal and Background Information

The goal of the business case is to prove the developed concept for enterprise-spanning business process automation and integration inside extended enterprises. It demonstrates sharing enterprise capabilities, services, and business processes among the participating partners.

The realization of an enterprise-spanning business process starting at the award of contract to the extended enterprise by the customer, and ending with the recognition of money to the bank account after satisfying delivery shall suffice for proving. For processing the needed business processes, the business process engine based on OFBiz is applied.

To understand the motivation for the chosen business case described in chapter 7.1.2, some background information is essential. One central research activity at the Automation Control Institute (ACIN) is the development on holonic, distributed control systems for shop floor automation. To animate that research work, at ACIN there is a testbed for such holonic controls available - the so-called Odo Struger Lab.

In coarse the testbed consists of a storage handling system, a multipoint pallet transfer system, a robot measurement system, and a robot assembly system. For simple demonstration of the networked control systems a flexible assembly process of a configurable numeric PC keypad was designed and deployed. The assembly process input is the customer's description of the arrangement of keys on the pad.

Therefore, the testbed offers the possibility to integrate a "real" production - in a closer context an assembly - process into the extended enterprise business

case scenario for demonstration purpose.

7.1.2 Scenario Description

A customer approaches the extended enterprise for producing a certain amount of customized keypads. Within the central enterprise of the extended enterprise, which is the extended enterprise coordinator, is the contact person for the customer. The central company itself is only able to develop and produce the printed boards for the keypads in-house as well as the final assembly of the keypads.

For the other necessary production tasks, this enterprise has to build up an extended enterprise, where the keys, the keypad body, and the electronic components for the printed board are delivered by the business partners. A logistics service provider takes care of delivery of parts and shipping the final product to the customer.

To exploit the advantage of extended enterprises, all sub-processes should be transparent on a certain level within the collaboration network. Some services have to provide progress status information, e.g. about the production process at the suppliers or the progress of the shipment. These information is callable either by the roles of the business partners or by the customer itself.

The extended enterprise constitutes out of:

- EE Coordinator: Is responsible to acquire orders, schedule production and to assemble the final product.
- Customer: Places the order for keypads by the EE Coordinator.
- Supplier 1: Produces the plastic injection-molded parts, which are the keys and the keypad body.

- Supplier 2: Delivers the electronic components for the printed board of the keypad.
- Logistics Service Provider: The tasks are to convey parts from the suppliers to the coordinator and to ship the final product to the customer.

7.2 Modeling the Business Case

The business case has been modeled according the recommended procedure if using UML diagrams. The first step is the definition of required roles participating the enterprise-spanning business process inside the extended enterprise (chapter 7.2.1). UML use case diagrams model the system functionality which is needed by the defined roles (chapter 7.2.2). Finally, the business process itself is modeled in two ways: using an UML activity diagram for descriptive modeling (chapter 7.2.3.1), and using the XML-based business process definition language XPDL to generate the input for the business process engine (chapter 7.2.3.2).

7.2.1 Definition of Required Roles

The defined naming convention for extended enterprise roles are: the roles start with EE followed by the role name with an optional appendix denoting a particular specialization of the role. The involved extended enterprise roles are as described below:

- EECustomer (@ customer): The customer is modeled inside the extended enterprise for the time span of the contracted order or project.
- EESalesman_EE (@ coordinator): Interacts tightly with the customer in supporting, negotiating and contracting orders.

- EEProductionManager_Assembly (@ coordinator): Schedules the final assembly of the product therefore fixes timing for procurement and parts suppliers.
- EEProductionManager_PrintedBoard (@ coordinator): Does production planning for printed board production.
- EEProcurement (@ coordinator): Serves for placing part orders.
- EELogisticsManager (@ logistics service provider): Concerns information exchange on transport orders and shipping status.
- EESalesman_Supplier (@ supplier 1): Represents the salesman on supplier 1 side for receiving and negotiating part orders and providing production progress information.
- EESalesman_Supplier (@ supplier 2): Same as for supplier 1.

7.2.2 Specifying the Prototype Functionality

Upon defining the participating roles, the system functionality has to be modeled from the perspective of those roles. The designed model has to present their requirements of the software solution.

The standardized technique for such models is the UML use case diagram, which describes the functionality of a system and users of the system. These diagrams contain the following elements: [EP00][Alh98]

- Actors are roles that users or other systems have in relationship to the system. An actor always has some interest in using the functionality the system provides.

- Use cases, which represent functionality or services provided by a system to actors. They are defined through plain text.

The use case diagram summarizes which use cases are available and their relationships to each other. Use cases can be linked with three types of relationships: [EP00]

- Include: An include relationship between use cases means that one use case can include and use other use cases in an explicitly defined place in its description.
- Extend: An extend relationship between use cases means that the base use case (the use case that is extended) is extended with additional behavior by the extending use case. The extension can be seen as optional functionality added to the base use case, and the base use case does not need to know of the extending use case.
- Generalization (not used in the following diagrams): One use case can be specialized to one or several use cases that inherit and add features to it.

Each role participating the exemplary business case has their own specific requirements that are designed with UML use case diagrams discussed in the following. To describe the diagrams, the main use cases are explained in plain text.

Figure 7.1 shows the use case diagram for the role EECustomer (@ customer). The main use cases are:

- Get capabilities of EE: Displays the extended enterprise capabilities to the actor. The actor can choose between ordering a standard product, a configurable product, or a new product. Further general information about the extended enterprise are shown to the actor.

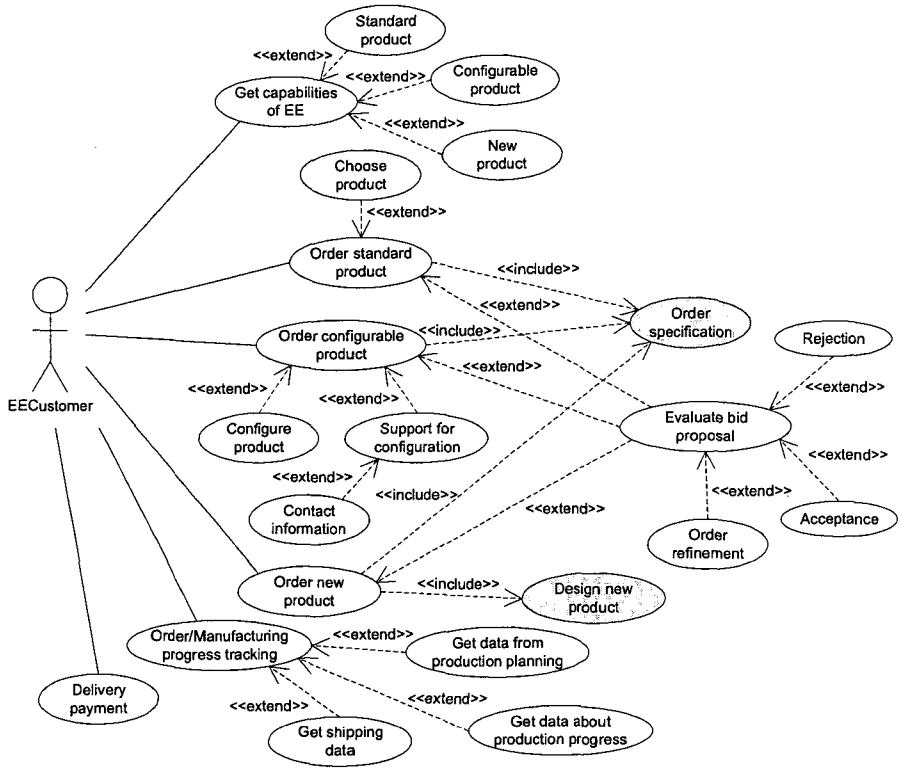


Figure 7.1: Use Case Diagram for the Role EECustomer (@ customer)

- Order standard product: After the actor has chosen the product, the order specification has to be done (see Figure 7.2). After that, the actor has to evaluate the offered bid proposal. The actor has three options: acceptance or rejection of the bid proposal, or order refinement.

- Order configurable product: If the actor needs assistance, support for product configuration is given - including contact information of the salesman. Having done the detailed order specification (see Figure 7.2), the actor has to evaluate the offered bid proposal in the same way as for standard products.

- Order new product: In addition to the use cases order specification and

evaluation of the bid proposal, the design of the new product is essential (details see Figure 7.3).

- Order/manufacturing progress tracking: As an additional service for the customer, the tracking of the order/manufacturing progress is offered. For it, data from the production planning system, data about the production progress, and shipping data is needed.
- Delivery payment: Payment of the delivered goods.

Figure 7.2 and Figure 7.3 show the use case diagrams for the use cases "order specification" and "design new product".

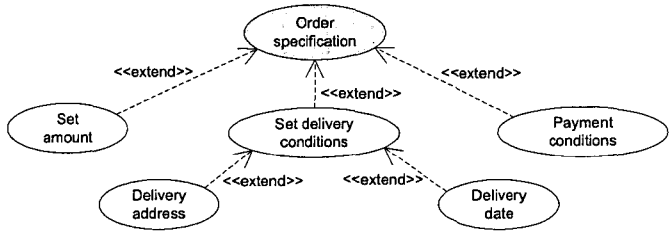


Figure 7.2: Use Case Diagram for the Use Case "Order Specification"

- Set amount: Definition of number of pieces.
- Set delivery conditions: The delivery conditions, including the delivery address and the delivery date, have to be defined.
- Payment conditions: Definition of payment conditions.

- Define product requirements: The detailed product requirements for the new product have to be specified as exact as possible.
- Negotiation about product specification: The offered product specification has to be negotiated. After that the functional specification can be

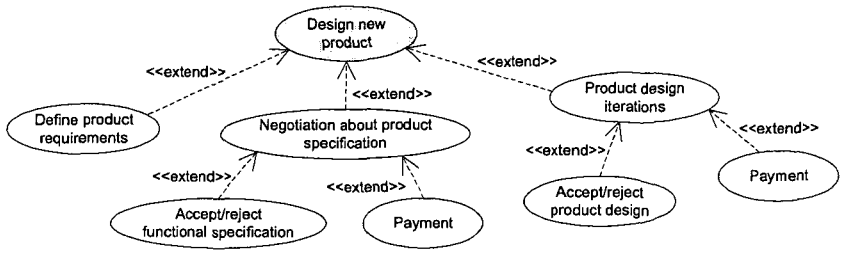


Figure 7.3: Use Case Diagram for the Use Case "Design New Product"

accepted or rejected. Independent from this decision, in most cases the creation of the product specification has to be paid.

- Product design iterations: The product is designed in one or more iterations. After each iteration, the design can be accepted or rejected. In both cases, a payment is required.

Figure 7.4 in combination with Figure 7.5 design the use cases for the role EESalesman_EE (@ coordinator).

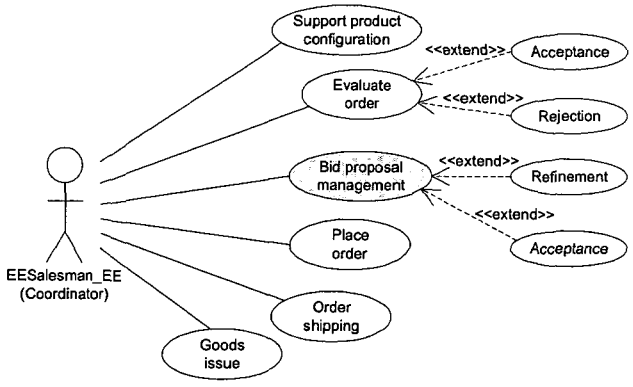


Figure 7.4: Use Case Diagram for the Role EESalesman_EE (@ coordinator)

- Support product configuration: Support the customer configuring the product (via e-mail, phone, etc.).

- Evaluate order: The incoming orders have to be evaluated and either accepted or rejected.
- Bid proposal management: The bid proposal generated by the system (see Figure 7.5) is either accepted or has to refine manually.
- Place order: If the incoming order is accepted, it has to placed into the system.
- Order shipping: For the final product, the needed shipping has to be ordered by the logistics service provider.
- Goods issue: The actor hands over the final goods to the logistics service provider.

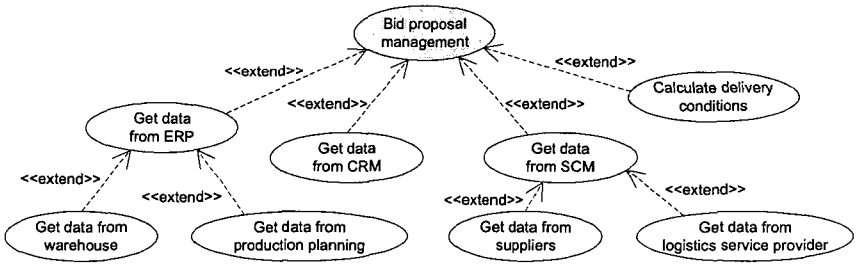


Figure 7.5: Use Case Diagram for the Use Case "Bid Proposal Management"

- Get data from ERP: Data from the warehouse (e.g. stock information) and production planning (e.g. capacities, production progress, etc.) is needed.
- Get data from CRM: Preferred customers get e.g. a discount, special delivery conditions, etc.
- Get data from SCM: Data from the suppliers (e.g. stocks, production progress, etc.) and logistics service providers (e.g. capacities) is needed.

- Calculate delivery conditions: Including delivery date, price, payment conditions, etc.

The last use case diagram (Figure 7.6) aggregates the requirements of the outstanding roles.

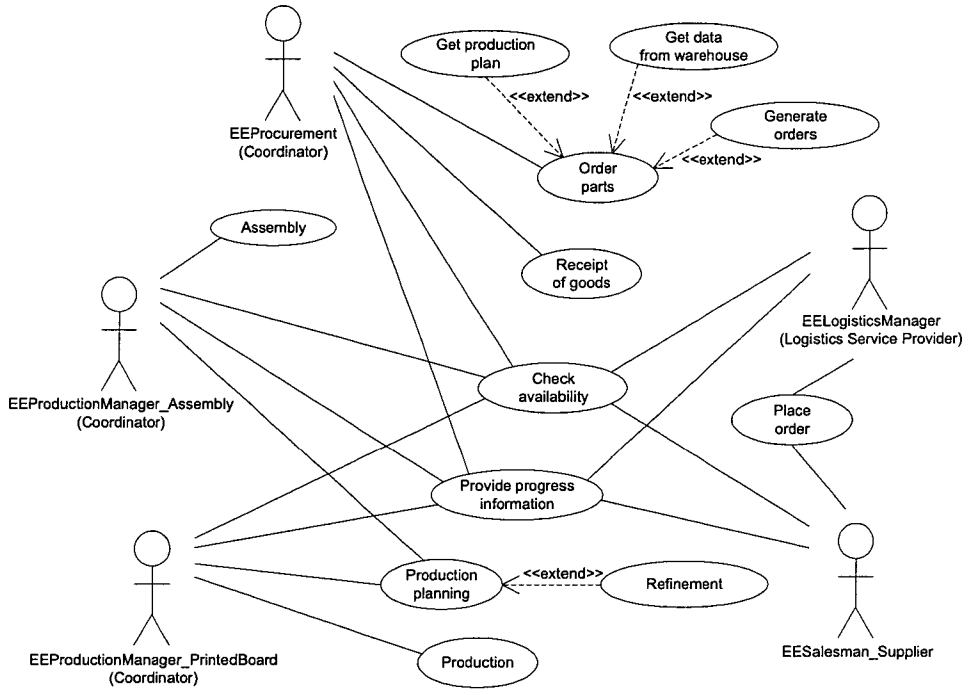


Figure 7.6: Aggregated Use Case Diagram

- Check availability: As a reaction of incoming orders by the customer, all participating roles have to check (e.g. if there are enough parts on stock) if the order can be fulfilled with the wanted specification.
- Order parts: The needed parts have to be ordered from the suppliers. The basis for the generated orders are the production plan to get the time schedule, and the warehouse data to calculate the amount of parts.

- Production planning: The production process has to be planned, e.g. the time schedule.
- Place order: The incoming orders have to be placed into the internal system to start internal processes.
- Production and Assembly: Internal processes to produce and assemble products.
- Provide progress information: The basis for decisions and calculations is the progress information provided by the business partners (e.g. according to schedule delivery by suppliers).
- Receipt of goods: The receipt of goods delivered by the logistics service provider.

7.2.3 Modeling the main Business Process

7.2.3.1 Descriptive Model

To explore and describe business processes in an descriptive model, the standardized UML activity diagram is used. This diagram contains the following elements: [Alh98][EP00]

- Activity States: The "states" in an activity diagram are activities to be performed, and the transitions are fired by ending activities. Activity states, or short activities, can be divided into subactivities. The symbol for activities is a rectangle with rounded corners.
- Control Flow: Activities are connected via transitions, which make up the control flow in an activity diagram. Each activity diagram has one or more start (solid circle) and stop (small solid circle surrounded by a

larger empty circle) symbols for the control flow. Guards can be applied directly to the control flow or in combination with a decision symbol (hollowed diamond). A control flow can be forked and joined. Forking means that the flow of control is split into several flows, each with its own activities. This technique is used when activities are performed in parallel with some form of synchronization. When flows are joined, if one flow reaches the synchronization bar first, it will wait for the other flows before continuing. The symbol for the synchronization bar is a bold line.

- Swimlanes: Can be used to explicitly show where activities are performed. Swimlanes are drawn as vertical rectangles in the activity diagram, and the activities that belong to a swimlane are placed within its rectangle. The swimlane is given a name that is placed at the top of the rectangle.

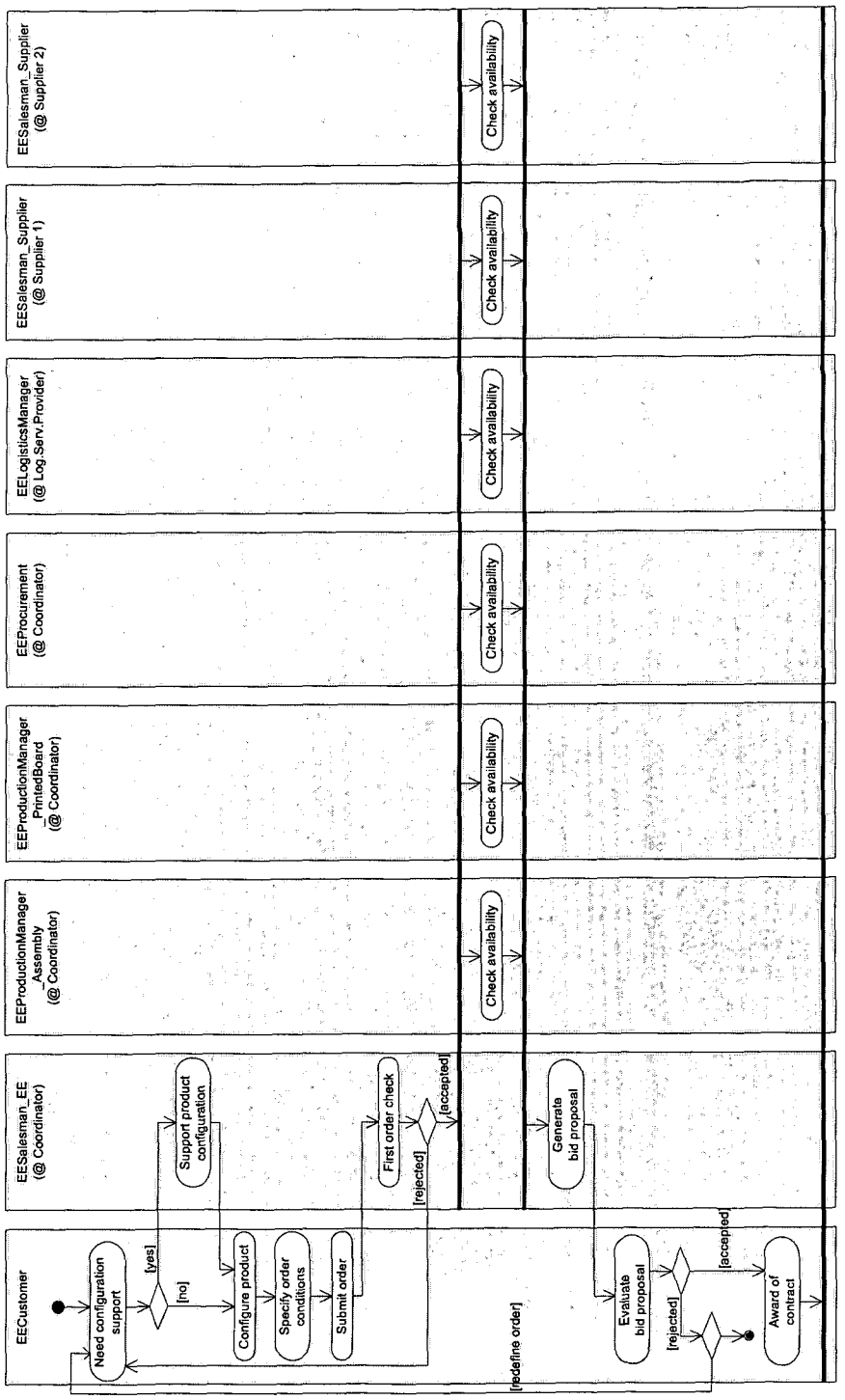


Figure 7.7: Activity Diagram of Main Business Process (Part 1)

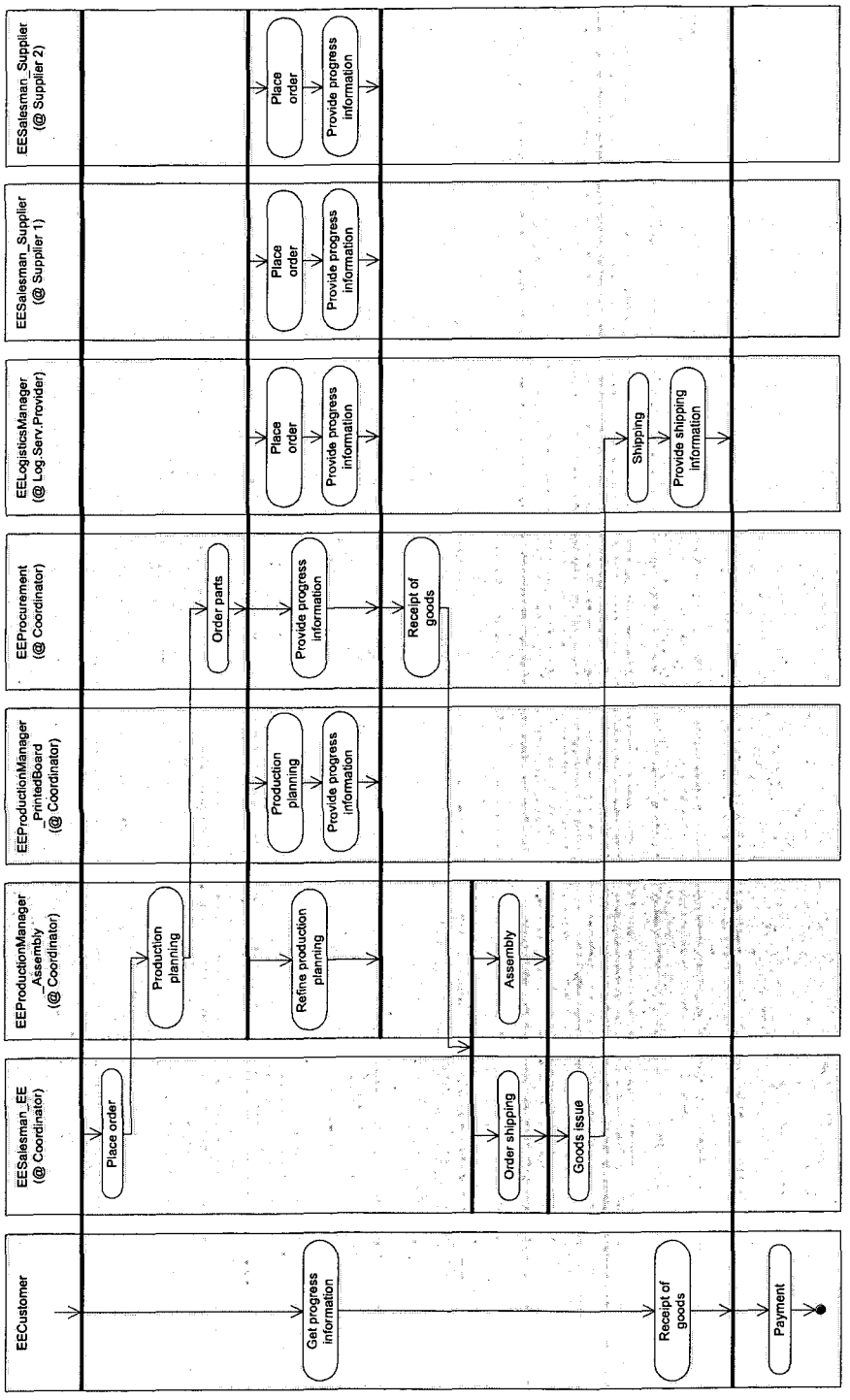


Figure 7.8: Activity Diagram of Main Business Process (Part 2)

Figure 7.7 and Figure 7.8 show the descriptive model of one existing enterprise-spanning business process of the exemplary business case implemented in the proof-of-concept prototype. This business process is started if the customer wants to order a configurable product - in this business case a customized keypad. The whole process can be structured into three emphases:

1. negotiating the order to get an award of contract from the customer,
2. controlling and executing the production and assembly process, and
3. delivery and payment of the product.

In the following paragraphs the three blocks of the process are discussed more precisely. As a precondition, the customer is convinced that the exemplary extended enterprise has the right capabilities to satisfy the requirements.

The process starts with the customer's decision whether or not to request support for using the online product configurator from the role EESalesman_EE (@ coordinator). He proceeds with configuring the product at the extended enterprises Web site supported with or without a salesman's assistance. The next step covers determining order conditions like delivery date, accounting address, shipping address, kind of payment, etc. After the order is fully specified, the order is ready to submit it to the extended enterprise, in particular to EESalesman_EE (@ coordinator). The job of the salesman is to first check the order on reasonability - if rejected, the customer has to restart the product configuration. Secondly to check for feasibility according the customers constraints. This involves inquiring production planning at assembly and printed board production, verifying suppliers availability through procurement and to contact the logistics service provider asking for free capacity. Basing on the gathered information the salesman is able to calculate a bid proposal. The customer then evaluates the bid and either allocates the

award of contract, or requires another iteration for refining the order, or in worst case canceling the order negotiations.

The second part of the process concerns production/assembly coordination and execution. The salesman places the order to the extended enterprise system. This act triggers the production planning at the EEProductionManager_Assembly (@ coordinator). In particular the production planner schedules the final assembly in respect to the order conditions. This data then goes to EEProcurement (@ coordinator) and EEProductionManager_PrintedBoard (@ coordinator). EEProcurement (@ coordinator) places suborders to both EESalesman_Suppliers dependent to the particular enterprise. Progress/status information from the suppliers, the logistics service provider and the printed board production is concurrently used for two purposes. One for eventually refine the assembly production schedule and second to provide information to the customer if requested. When all parts are received in time the final assembly is launched according the work schedule. In parallel the EESalesman_EE (@ coordinator) orders the shipping at the logistics service provider. In parallel to all activities the customer can request progress information about the order.

The last step covers delivery and payment. The goods are picked up by the logistics service provider and shipped to the customer. The logistics service provider provides status data during shipment to the customer. Reception of goods triggers payment and terminates the business process.

7.2.3.2 Utilizable for Business Process Engine

A standardized business process description language enables a process definition, generated by one modeling editor (e.g. open source editor JaWE [Enh03]), to be consumed as input to a workflow engine (Figure 7.9). The

meta-model framework identifies commonly used entities within a process definition, their relationships and attributes. As already mentioned, the realized business process engine uses the standard XPDL (XML Process Definition Language, published by the Workflow Management Coalition [Wor02]).

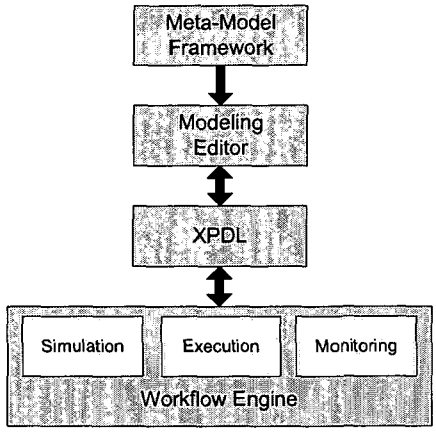


Figure 7.9: Business Process Modeling using XPDL

Because it makes no sense to describe the whole designed XML-based business process definition in detail, only the most important parts of such a XPDL file is discussed to give the reader a feeling how business processes can be described using XPDL.

In XPDL the concept of a package is introduced (Figure 7.10), which acts as a container to hold a number of common attributes from the process definition that are automatically inherited by each process definition contained within the package. It is possible to define several processes within one package, which may share the same tools and participants.

As Figure 7.10 shows, the element "Package" has the attributes "Id" (required) and "Name" to identify the package, and sequenced sub-elements explained in Table 7.1.

```
<xsd:element name="Package">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:PackageHeader"/>
            <xsd:element ref="xpdl:RedefinableHeader" minOccurs="0"/>
            <xsd:element ref="xpdl:ConformanceClass" minOccurs="0"/>
            <xsd:element ref="xpdl:Script" minOccurs="0"/>
            <xsd:element ref="xpdl:ExternalPackages" minOccurs="0"/>
            <xsd:element ref="xpdl:TypeDeclarations" minOccurs="0"/>
            <xsd:element ref="xpdl:Participants" minOccurs="0"/>
            <xsd:element ref="xpdl:Applications" minOccurs="0"/>
            <xsd:element ref="xpdl:DataFields" minOccurs="0"/>
            <xsd:element ref="xpdl:WorkflowProcesses" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="Name" type="xsd:string"/>
    </xsd:complexType>
</xsd:element>
```

Figure 7.10: XML Schema for XPDL Element "Package" [Wor02]

PackageHeader	Set of elements specifying package characteristics.
RedefinableHeader	Set of elements/attributes used by both the package and process definitions.
ConformanceClass	Structural restriction on process definitions in this package.
Script	Identifies the scripting language used in expressions.
ExternalPackages	Reference to another package definition defined in a separate document.
TypeDeclarations	List of data types used in the package.
Participants	List of resources used in implementing processes in the package.
Applications	List of workflow application declarations.
DataFields	List of workflow relevant data defined for the package.
WorkflowProcesses	List of the workflow processes that comprise this package.
ExtendedAttributes	List of vendor-defined extensions that may be added to the package.

Table 7.1: Sub-Elements of Element "Package" [Wor02]

To define the main business process for the prototype discussed in chapter 7.2.3.1, the sub-elements "Participants" and "WorkflowProcesses" are essential, additionally to "PackageHeader" (required) and "RedefinableHeader". The sub-element "Participants" (Figure 7.11 and Table 7.2) defines the performer(s) of the process activities. The following types are possible: resource set, resource, organizational unit, role, human, or system. A role and a resource are used in the sense of abstract actors.

```
<xsd:element name="Participants">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:Participant" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="Participant">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:ParticipantType"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:ExternalReference" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="Name" type="xsd:string"/>
    </xsd:complexType>
</xsd:element>
```

Figure 7.11: XML Schema for XPDL Element "Participants" [Wor02]

ParticipantType	Definition of the type of workflow participant entity.
Description	Short textual description of a workflow participant.
ExternalReference	Reference to an external specification of a participant.
ExtendedAttributes	Optional extensions to meet individual implementation needs.

Table 7.2: Sub-Elements of Element "Participant" [Wor02]

The sub-element "WorkflowProcesses" (Figure 7.12 and Table 7.3) defines the elements that make up a workflow. It contains definitions or declarations for activities, transitions, applications, participants, and process relevant data.

ProcessHeader	Set of elements specifying process characteristics.
RedefinableHeader	Set of elements/attributes used by both the package and process definitions.
FormalParameters	List of parameters that may be passed to the process.
DataFields	List of workflow relevant data defined for the process.
Participants	List of resources used in implementing the process.
Applications	List of workflow application declarations.
ActivitySets	List of self contained sets of activities and transitions.
Activities	List of activities that comprise the process.
Transitions	List of transitions that connect the process activities.
ExtendedAttributes	Optional vendor-defined extensions to meet implementation needs.

Table 7.3: Sub-Elements of Element "WorkflowProcess" [Wor02]

```
<xsd:element name="WorkflowProcesses">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:WorkflowProcess"
                    minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="WorkflowProcess">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:ProcessHeader"/>
            <xsd:element ref="xpdl:RedefinableHeader" minOccurs="0"/>
            <xsd:element ref="xpdl:FormalParameters" minOccurs="0"/>
            <xsd:group ref="xpdl:DataTypes"/>
            <xsd:element ref="xpdl:DataFields" minOccurs="0"/>
            <xsd:element ref="xpdl:Participants" minOccurs="0"/>
            <xsd:element ref="xpdl:Applications" minOccurs="0"/>
            <xsd:element ref="xpdl:ActivitySets" minOccurs="0"/>
            <xsd:element ref="xpdl:Activities" minOccurs="0"/>
            <xsd:element ref="xpdl:Transitions" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="Name" type="xsd:string"/>
        <xsd:attribute name="AccessLevel">
            <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                    <xsd:enumeration value="PUBLIC"/>
                    <xsd:enumeration value="PRIVATE"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:complexType>
</xsd:element>
```

Figure 7.12: XML Schema for XPDL Element "WorkflowProcesses" [Wor02]

After this short, pointwise report about the features of XPDL, the possibilities of XML-based business process description languages are well imaginable. The most important advantages are standardized, machine-processable definitions of business processes, and enabling the interchangeability between different software systems.

7.3 Implementation Environment

To realize the aim of a cost-effective software system, additionally to the already discussed open source business processs engine based on OFBiz (see chapter 6.2) supplementary open source software packages are used to implement the proof-of-concept prototype. Figure 7.13 shows the interaction of the software tools to build up an optimal implementation environment:

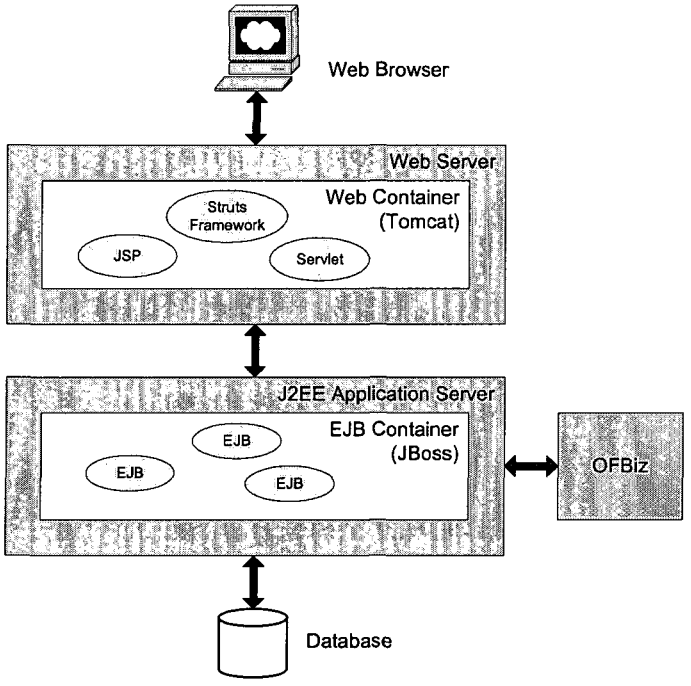


Figure 7.13: Implementation Environment for the Prototype

- Web browser: For an Internet-based application, a standard Web browser such as Microsoft Internet Explorer is the client to present information to the human user. Web browsers are available for free.
- Web container: Tomcat is a free, open-source implementation of Java Servlet and JavaServer Pages technologies developed under the Jakarta

project at the Apache Software Foundation. Tomcat is available for commercial use under the ASF license from the Apache Web site [Apa03b]. Tomcat is itself a Web server to reply the HTTP-requests of the users. But it is also possible to combine Tomcat with another Web servers, if necessary. Benchmarking Tomcat with a commercial servlet engine, both performed in a consistent and similar fashion, and Tomcat can even handle more concurrent users without generating as many errors as the commercial one [Gui03]. To realize the MVC design pattern, the user interface framework Struts is used (see chapter 7.3.1), which is running inside this container.

- EJB container: JBoss ([JBo03]) was originally designed as EJB container hosting EJBs, today JBoss is a mature application server (see chapter 7.3.2).

- OFBiz: see chapter 6.2.

- Database: Because of using the platform independent standard JDBC (Java Database Connectivity), which is a standard library for accessing relational databases, every available relational database can be used for the prototype. Most of the companies already have an existing database including valid licenses.

7.3.1 User Interface Framework

Just like a building must have a solid foundation from which the rest of the structure can grow, Web applications should be built with the same principle in mind. The Struts open source framework, which was originally created by Craig R. McClanahan, provides developers a unified framework from which Internet applications can be based upon. The Struts framework is one of

many well-known and successful Apache Jakarta projects, and is based on Java Servlet and JavaServer Pages (JSP) technologies [Apa03a].

Java Servlets are the front line in Java Web application development because they provide an easy way for server side programs to communicate with Web-based clients. Due to Java Servlets are not executed directly by the Web server, they require a servlet container, sometimes referred to as a servlet engine, to host the servlet. Among other things, the container is responsible for managing the life cycle of the servlet. [Pat99]

Based on servlet technology, JavaServer Pages is set to be one of the most important elements of Java server programming. They combine markup (whether HTML or XML) with nuggets of Java code to produce a dynamic Web page. Each JSP page is automatically compiled to a servlet by the JSP engine, the first time it is requested, and then executed (Figure 7.14). [PB99]

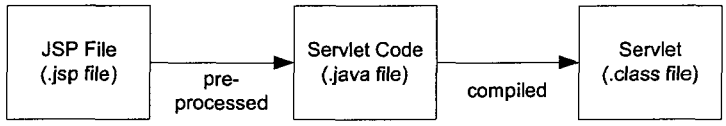


Figure 7.14: JSP-To-Servlet Flow [IBM03]

Both technologies - servlets and JSP - have their strengths and weaknesses. In fact, where servlets is strong, JSP is weak, and vice versa [Pan01]. Therefore, for the perfect solution a combination of the two technologies is inevitable. Additionally to that, Struts brings the advantages of MVC to J2EE Web application development by using the so-called Model 2 architecture as the underlying concept (Figure 7.15): [IBM03]

- Client browser: An HTTP request from the client browser creates an event. The Web container will respond with an HTTP response.

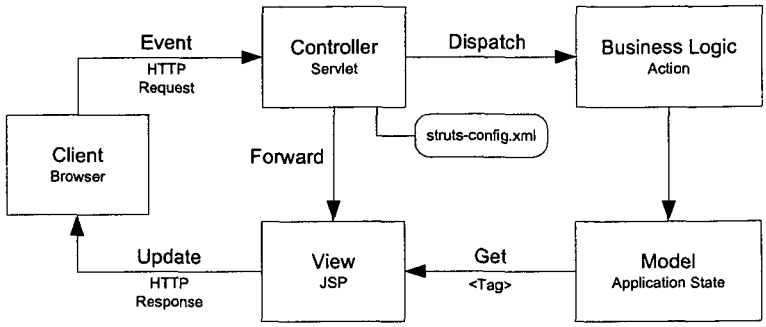


Figure 7.15: Struts Overview [Pan01]

- Controller: The controller receives the request from the browser, and makes the decision where to send the request. With Struts, the controller is a command design pattern implemented as a servlet. The struts-config.xml file configures the controller.

- Business logic: The business logic updates the state of the model and helps control the flow of the application. With Struts this is done with an action class as a thin wrapper to the actual business logic.

- Model state: The model represents the state of the application. The business objects update the application state. ActionForm bean represents the model state at a session or request level, and not at a persistent level. The JSP file reads information from the ActionForm bean using JSP tags.

- View: The view is simply a JSP file. There is no flow logic, no business logic, and no model information.

7.3.2 Enterprise JavaBeans Container

The J2EE components Enterprise JavaBeans (EJB) run in the EJB container, a runtime environment within the J2EE application server. This container provides system-level services such as transactions to its EJBs. These services enable a quick deployment of EJBs, which form the core of transactional J2EE applications.

The EJB specification defines three different types of Enterprise JavaBeans (Figure 7.16):

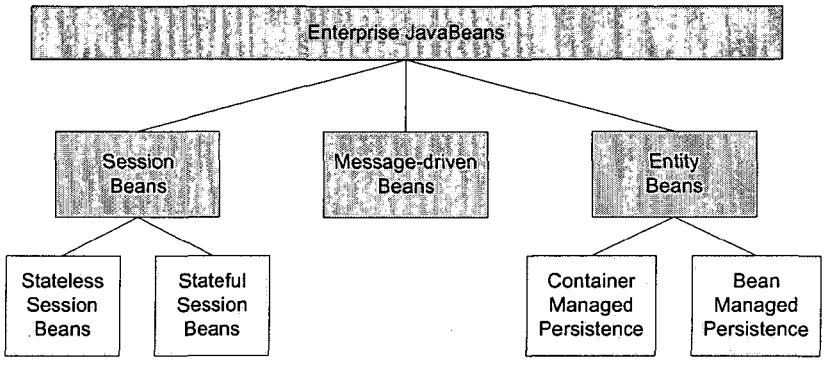


Figure 7.16: Types of Enterprise JavaBeans

- Entity beans: An entity bean represents a permanent or inventory item used by the business (e.g.: bank account). A container-managed persistence (CMP) entity bean leaves management of its persistent state and relationships to an EJB container. There is no need to write database-access code because it is generated automatically at deployment time. A bean-managed persistence (BMP) entity bean is responsible for managing its own relationships and persistence state in the database. The database-access logic is written directly into the bean class.

- Session beans: A session bean represents services, processes and client-

server sessions, and they are non-persistent, transaction-aware components. A stateless session bean is shared by many clients and so they are not dedicated to a single client. A stateless session instance cannot hold session data from one method invocation to the next. A stateful session bean is dedicated to a client that created it. A stateful session instance can hold session data from one method invocation to the next.

- Message-driven beans: Message-driven beans are stateless, server-side components that are used to receive and process messages that Java clients send to them. With message-driven beans, asynchronous communication can be implemented.

The open source project JBoss was started in March 1999. While JBoss began as, and still is, an EJB container, today JBoss is a mature J2EE application server containing many additional functionalities. JBoss is downloaded more than 150,000 times a month, and competes head on with proprietary offerings such as BEA WebLogic and IBM WebSphere.

7.4 Realization Fruits

This chapter want to spotlight the realization fruits achieved during the prototype implementation. The aim is not to present the whole software development, but rather to take out the most important results that can help on the reader of this work.

7.4.1 Performance Measurement of Web Services

To familiarize with the brand-new Web service technology and to reduce development risks as early as possible, the measurement of the Web service perfor-

mance using different levels of security was implemented. The test environment was built up on the server side with a SOAP servlet from Apache running in the Web container, and an EJB container for session and entity beans (Figure 7.17). On the client side, there was a Java application using SOAP over HTTP(s) to communicate with the server.

For the evaluation of the performance measurement of Web services the Apache SOAP servlet was modified: the servlet measured the time of certain processing steps and inserted the data into the database at the end of processing.

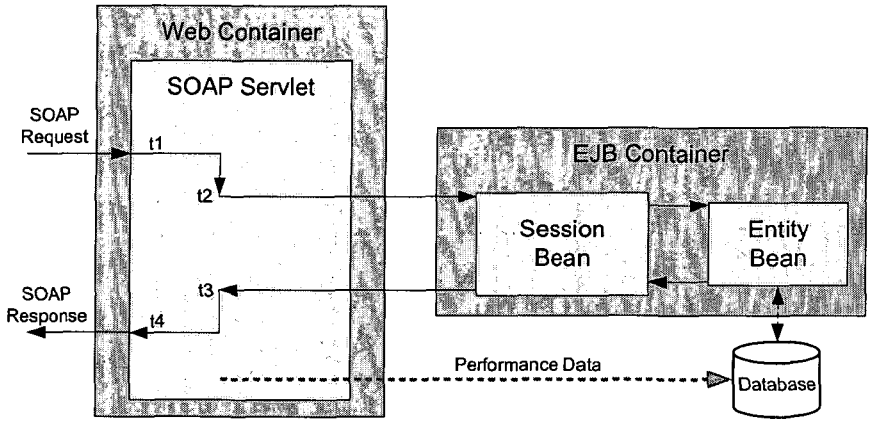


Figure 7.17: Time Measure Points for Performance Measurement

The following times were measured (Figure 7.17):

- EJB processing time (t3 - t2): This is the time span between the call of the service (session bean) and the return of the result.
- Client time: This time was measured by the client. It starts with the Web service invocation, and ends when the client gets the response of the Web service call.
- Web service time ((client time) - (t4 - t1)): Because the server processing

time (t4 - t1) was measured inside the servlet, the time span for the servlet initialisation and further administration work of the Web container was included too.

The client of the test environment sent SOAP requests with different security levels to the SOAP server: NO security, SSL (Secure Socket Layer), DS (Digital Signature), and SSL + DS.

Depending on one parameter of the Web service call, the SOAP server responded with different amount of datasets: 10, 100, or 1000 datasets. The technical data of the used server were: Windows 2000 Server, AMD 1 GHz Processor, and 786 MB RAM.

The Figure 7.18 shows the measurement results for the Web service time - on the one hand for an Intranet scenario (here: institute internally), and on the other hand for the Internet communication (here: between Athens and Vienna using 64 kBit Internet access). The measurement values are median values because of large measurement variation due to various network traffic.

Finally, a test scenario for concurrent Web service invocation was developed using the same client as above mentioned. The result of this performance measurement showed the change of the EJB processing time using different number of clients. The resulting diagram showed a linear growing for the EJB processing time with the rising number of clients.

7.4.2 User Acceptance of Implemented Prototype

Due to the user interface of the implemented prototype was realized with the latest Web technology (Java Server Pages in combination with the Struts framework), the user can use a standard Web browser to handle the software system. A high end user acceptance is given if the user interface is clearly

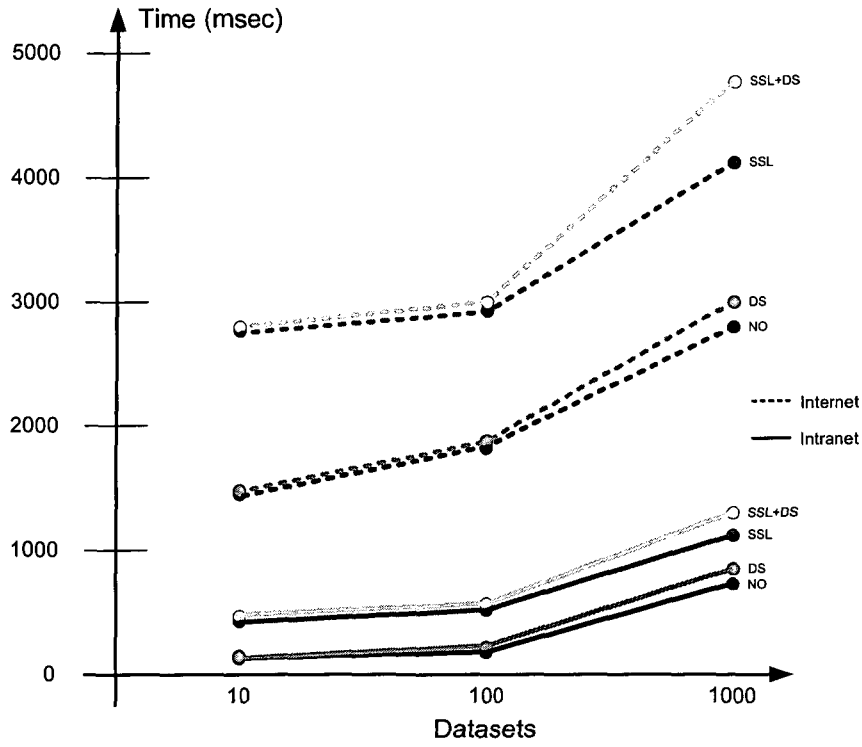


Figure 7.18: Measurement Results for Web Service Time

arranged and easily usable. Exemplary for the realized user interfaces, some screenshots taken from the customer's user interface are discussed in the following.

The underlying structure of the user interface is composed of four areas:

- Top: Shows the logo of the extended enterprise.
- Left: The menu items to enable easy navigation through the system.
- Right: Displays the forms for user input and system results.
- Bottom: Status bar to inform the end user about problems.

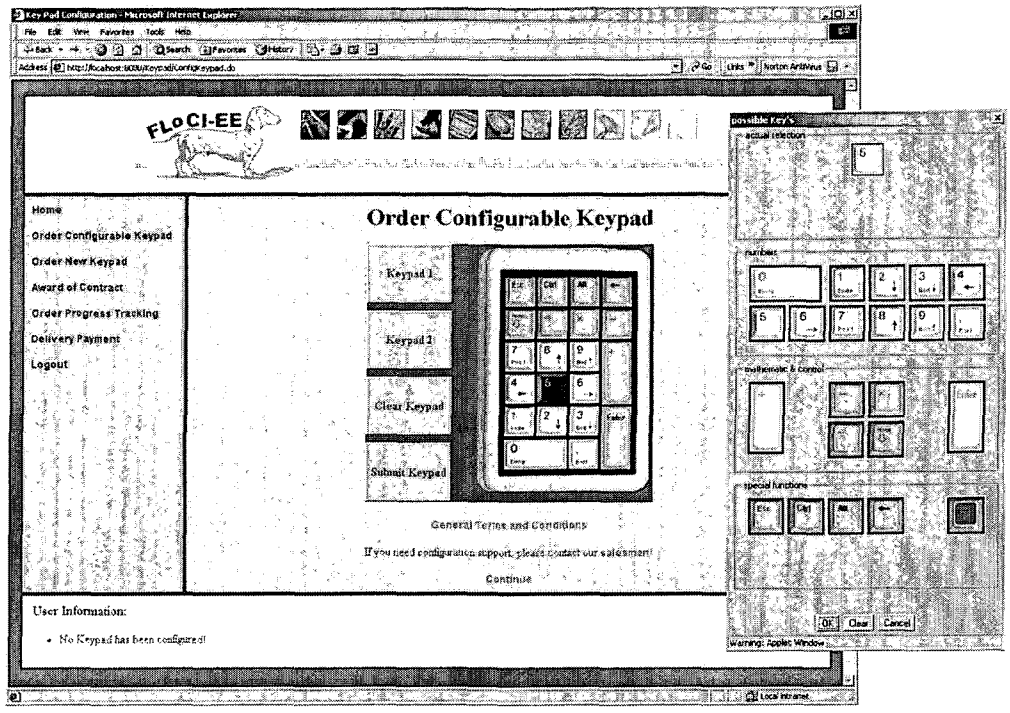


Figure 7.19: Screenshot - Configuration of Keypad

Figure 7.19 shows the screen where the customer has to configure the keypad. For this, a Java applet is displayed in the right area. Using this applet, the customer can configure the keypad either by selecting one of the two standard keypads, or selecting one of the keys and afterwards configuration using the pop-up menu as shown in Figure 7.19. If the customer selects the "continue" link without submitting a keypad, the status bar in the bottom area informs the customer.

Figure 7.20 shows the form where the customer has to enter the detailed order conditions such as delivery address and date, amount, and payment method.

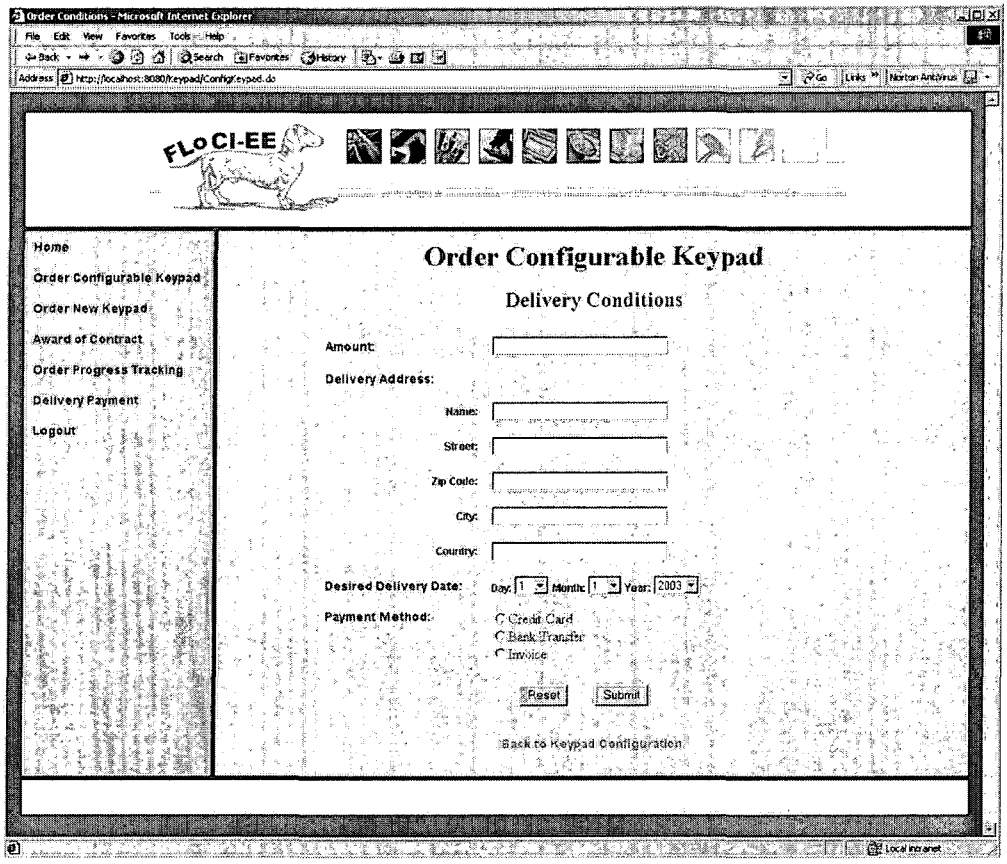


Figure 7.20: Screenshot - Input of Order Conditions

7.4.3 Experiences with Open Source Software

All open source projects might have in common the lack of good documentation. The reason for that is the vitality and dynamics of such projects, caused by the interaction among the project operators, the involved developers, and the needful end users. Everyone brings in her specific views, experiences and requirements. But this dynamics slow down the writing efforts of documentations at least during development, which makes it harder to assess projects and to acquaint oneself with them. Even in some cases documentations di-

verge from the current project release. This weakness forces to put a little more effort into examining a certain development.

Depending on their maturity, the open source software tools are documented accordingly. Therefore, each project has to be considered separately.

To make oneself familiar with the mature J2EE application server JBoss needs the least effort and time, due to well documentation and simple handling. Installation, configuration, and deployment of the first "Hello World" example can be done within one hour. This first feeling of success is very important, and is the first step to learn all features of JBoss during the ensuing weeks.

Before the developer has a good grasp of the MVC framework Struts, a higher barrier has to get over. The reason for that is the complex architecture and the effort for understanding. After few days reading documentation, first applications can be implemented. Before being able to use Struts, the Web container has been set up so that it knows to map all appropriate requests with a certain file extension to the Struts controller servlet. This is done in the "web.xml" file that is read when the Web container starts. To realize Web applications, the developer has to do the following tasks simultaneously, which take getting used to handle that:

- Configuration of the controller servlet via the XML file "struts-config.xml" to determine the flow of action.
- Designing the model objects (JavaBeans) that take the request data from the user and store the results for the duration of the process.
- Writing an "Action class" for each logical request that may be received. The goal of this component is to process a request, and return an object that identifies where control should be forwarded to provide the appropriate response.

- Building the view components of the application, which are primarily be created using JSP technology. In particular, Struts provides support for implementing internationalized applications, as well as for interacting with input form.

Same story with the OFBiz project. After visiting the project homepage revealed first features of the software, the real work starts in elaborating through the documents, manuals and developers mailing lists accompanied by extensive tests of available releases. Simply studying the manuals leaves a couple of questions open, because of they are written on a level of abstraction to prevent permanent changes when something is changed in the project.

Dependent on previous knowledge in the field of Web application servers, database integration, or JSP programming, it could take approximately two to five weeks to become common and being able to fully apply OFBiz in individual applications. For newcomer to the project the provided documentation might be too short. Together with the project objective of high flexibility it takes its time to get a clear picture of OFBiz.

Recapitulating, the experiences with open source software have been very satisfactory. The software operation has proved rather stable and has performed well, at least within our prototype environment. Once familiar with an open source project it reveals the various benefits. Open source projects offer great flexibility concerning application, possibility to enhance the development with personal required features, and the successful, rapid support by the community. But, good experience in programming and used technologies is convenient.

7.5 Profitability Appraisement for Developed System

The goal of this chapter is to outline a realistic exploitation strategy and profitability appraisement of the developed software system described in this work. To simplify matters, the resulting product will be called "FLoCI-EE".

7.5.1 Development Costs

The foundation stone for a commercial FLoCI-EE system has been done within an European research and development project running from January 2001 until December 2003. The consortium is built up of three software developers, three different kind of end-users, and the Automation Control Institute. The total investment of the project partners are more than 1,35 million Euro (nearly 31 man-years) for the alpha-version of FLoCI-EE.

To get a more commercial software system of FLoCI-EE (beta-version), at least twice as much development effort is necessary. From this it follows that the FLoCI-EE consortium has to invest further about 3 million Euro for the further development of FLoCI-EE. Considering the target price (see Figure 7.22), the project members has to sell the FLoCI-EE system about 300 times to recoup their investments.

7.5.2 Unique Selling Proposition

The distinction of FLoCI-EE compared to products from competitors is the unique selling proposition:

- **FLoCI-EE is flexible** and thereby exactly adaptable and expandable to end-user/branch requirements. FLoCI-EE grows with the company

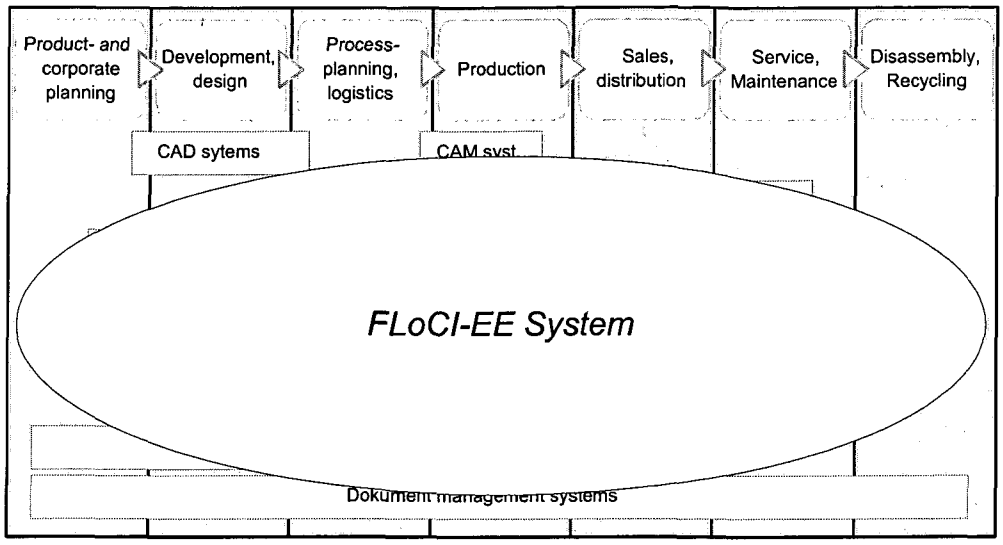


Figure 7.21: FLoCI-EE Support the Product Life Cycle

and can be successfully integrated with existing legacy systems by following the essential industry standards (e.g. XML). As a result of the Internet capability of FLoCI-EE, the system is independent concerning the location (e.g. sales force, home-work).

- **FLoCI-EE is low-priced** through an open-source basis framework. Moreover, the modular structure of FLoCI-EE enables a fast and cost-saving customization and expansion. By the use of a standard Web browser, there will be no license fees or maintenance costs per workstation.

- **FLoCI-EE is short term available**: Customizing will be done in weeks instead of months or even years, which is usual at the competitors.

- **FLoCI-EE is universal** by covering all needed functionality of SMEs (small and medium sized enterprises) inside the basis package. This includes the support of the whole product life cycle from start to finish

(Figure 7.21).

- **FLoCI-EE is independent**: The use of universally accepted technologies and standards should it make possible to establish an international market for FLoCI-EE components. Also the promotion of open-source solutions avoids - beside saved license costs - the presently strong addiction to only one big software vendor.
- **FLoCI-EE is designed for B2B** (business-to-business) which facilitate the use of extended enterprises to strengthen the market position of small and medium sized enterprises.
- **FLoCI-EE is designed for B2C** (business-to-customer) which enables modern e-commerce solutions using the Web.

7.5.3 Target Market

Due to the comparative low costs of FLoCI-EE, this system is primarily interesting for SMEs (small and medium sized enterprises) which have eschewed the high investments for a tailored IT solution up to now. Because the demands of major enterprises (Fortune 500) will be saturated in the near future, the SME market gains in importance. Up to now there is no market leader established.

Analyzing the enterprise landscape of the EU put forth that 99,8% of the European companies are SMEs, and therefore potential customers of FLoCI-EE. That corresponds to a total of 8,7 million firms. If it is possible to establish the brand "FLoCI-EE" at the SMEs, there is nothing to be said against it to market FLoCI-EE also for large companies.

According a market study ordered by KHK Sage, 83% of SMEs are prepared to pay up to 15,000 Euro for a holistic IT solution (Figure 7.22). Therefore, the pricing of FLoCI-EE has to be in this range to meet customer's acceptance

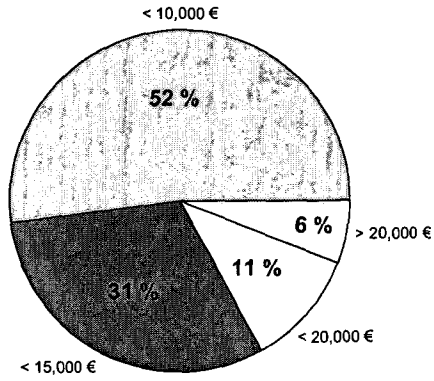


Figure 7.22: SMEs Price Acceptance for Enterprise Software [Mic02]

on the one hand, and to offer the essential price advantage compared to the competitors on the other hand.

7.5.4 Competition Situation

Software vendors can be differed in vendors for big customers (e.g. SAP, Baan, Oracle) and ones for SMEs (e.g. Exact, Navision). The first mentioned companies are not yet direct competitors because they are not "attractive" enough for SMEs up to now. But the second ones are presently active in this market segment and therefore the most important competitors for FLoCI-EE. Due to the fact that FLoCI-EE is more flexible and cost effective, it could beat the others. Figure 7.23 shows the position of FLoCI-EE regarding the target market compared to other software vendors.

7.5.5 Realization Risks

The SWOT analysis shown in Figure 7.24 is a very effective way of identifying the strengths and weaknesses, and of examining the opportunities and threats

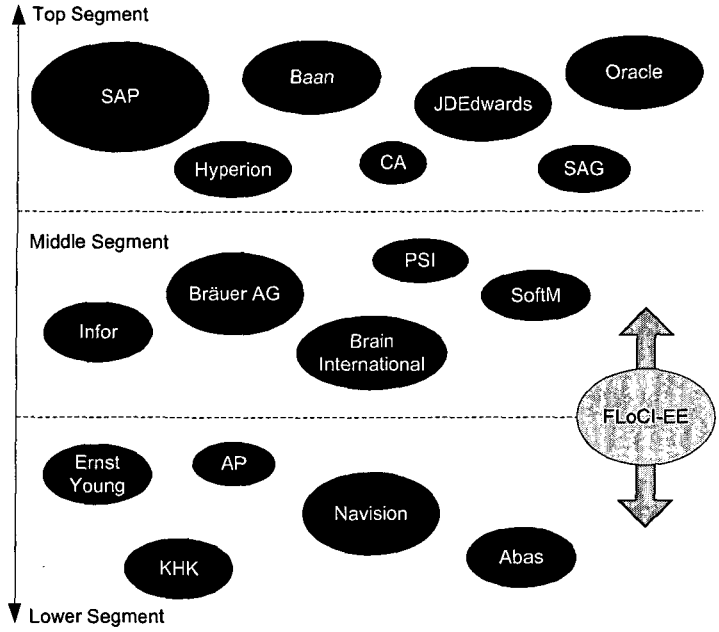


Figure 7.23: Target Market for FLoCI-EE [Mic02]

of FLoCI-EE. This will help to focus on the strengths, minimize weaknesses, and take the greatest possible advantage of the available opportunities.

7.5.6 Introduction Costs

The introduction of the software system FLoCI-EE causes three different kind of costs: primary acquire costs, adjustment costs, and operating costs. For most of the costs it is very difficult to quantify them exactly, because there is a high dependency to the environment conditions in respective companies.

7.5.6.1 Primary Acquire Costs

- Software License: This position is assumed with maximal 15,000 Euro per company (according chapter 7.5.3).

Strength:	**Weakness:**
1. Technology leadership 2. Partial financing by the EC 3. Established customers 4. Flexible assortment of products 5. Based on open-source components 6. Price advantage	1. Barely available branch-specific solutions 2. Scarcely established brand name 3. No common, coordinated marketing and sales 4. High pressure of time for customizing to keep the price advantage
Opportunities:	**Threats:**
1. Increasing IT investments of SMEs 2. New markets in Eastern Europe 3. Consistent European regulations 4. No market leader in this assortment 5. Highly market growth	1. Increasing orientation to the target group of SMEs by other IT vendors 2. Many competitors with sufficient capital

Figure 7.24: SWOT Analysis of FLoCI-EE [Mic02]

- Hardware: Due to FLoCI-EE needs no special hardware specification, existing/standard personal computers and servers can be used.

7.5.6.2 Adjustment Costs

- Integration: Depends on the software landscape of the company. Because FLoCI-EE uses only open, standardized interfaces, the integration can be done fast and cost-effective.
- Employee Training: Mainly the system administrator needs an intensive training program. The teaching effort for the end-users is very low because Web-based user interfaces are applied (client: standard Web browser).

7.5.6.3 Operating Costs

- Maintenance: System administrator(s) are needed for ongoing service.
- Function Enlargement: Due to the service-oriented concept of FLoCI-EE, it can be done very easily and cost-effective.

7.5.7 Benefits

7.5.7.1 Direct Benefits

- Business Processes: Additional to work flow acceleration, realized business processes are transparent and accessible for improvements. The achievable time saving can be applied generally with 10% for all end-users.
- Document Management: Management of technical data and electronic documents. Providing secure data availability. Estimated reduction of costs are at least 5% for all end-users.
- Accessibility: External business partners and customers can access important business information in "real-time". Time and effort savings for each end-user can be assumed with 10%.

7.5.7.2 Indirect Benefits

- Business Process Redesign: In the cause of the FLoCI-EE introduction, the company is able to optimize and slim the existing processes using the provided business process engine. Thus leads to a fundamental reduction of work steps, control effort, documentation effort, etc.

- Time-to-Market Reduction: Through automation and optimization of business processes, total enterprise integration, and collaboration with external partners over Internet, design and lead times can be reduced dramatically. From this it follows that time-to-market can be diminished about 30%.

- Productivity Enhancement: By the decrease of non-productive efforts (e.g. the search and structure time during construction phase is about 40%) there is a high potential for cost savings.

- Quality Improvement: The product quality can be advanced by providing the latest versions of design, production, and calculation documents to the right operator within the company - at any time and without any additional effort. Supplementary the business process engine can force users to create particular documents to improve the quality of the whole product manufacturing process.

- Synergies through System Integration: By the use of one integrated system it is possible to close open loops inside the enterprise (information feedback from quality assurance, sales and marketing, maintenance, etc.). Save expenses can be achieved through the holistic view of an enterprise where information - obtained from the manufacturing process - are used for continuous process planning.

7.5.8 Break-Even Analysis

The best way to visualize the profitability appraisement is the use of a break-even diagram, where the investments are confronted with the revenues of such a software introduction. The break-even is achieved when the cash-flow (revenue minus investment) becomes positive.

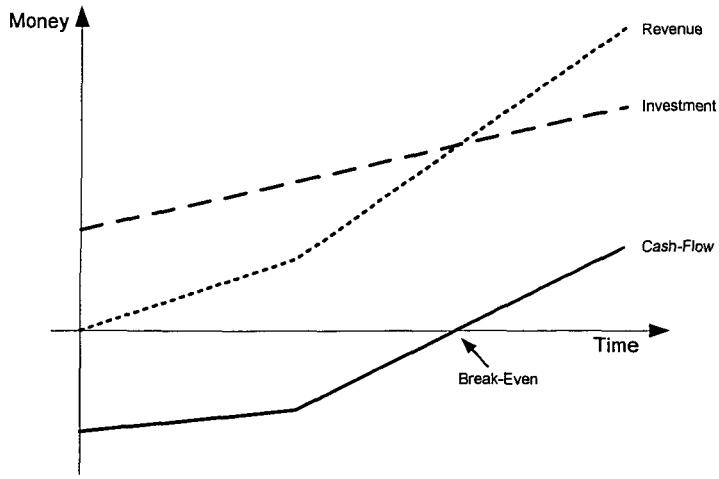


Figure 7.25: Break-Even Diagram

In this work only an exemplary diagram is shown (Figure 7.25) since the exact values for investments and savings depend on the environment conditions of each enterprise. On the outgoing side there are assumed fix costs for the primary acquire costs, and variable costs for adjustment and operating costs. For the revenues there is presumed that the direct benefits affect immediately the revenues, and the indirect benefits impact the revenues after a short work in time.

Chapter 8

Summary and Outlook

During the last decade the way of doing business has changed dramatically as a result of the digital revolution effected by the appearance of the Internet. For enterprises, collaboration is a *must* to stay competitive in nowadays turbulent, globalized markets.

This work describes a holistic, flexible, and open software system to enable seamless collaboration and integration between business partners. For the first time, the discussed approach considers both the horizontal integration between independent enterprises and the vertical integration between intra-enterprise levels.

Chapter 2 of this work outlines the development from Taylor's theory to holistic systems. This evolution is stamped by the focus on market demands and cooperation, instead of partial optimization. The result is the new leading strategy called Holistic Manufacturing.

In chapter 3 important definitions, subsystems, and standards are introduced. Discussed are, among other things, the main business systems supporting the product life cycle (PDM, ERP, SCM, CRM), the latest XML-

based standards to describe business processes, and the concepts behind the upcoming technology Web services.

The international state of the art in enterprise-spanning business process collaboration is analyzed in chapter 4. Beside the methodology for enterprise application integration, the most important extended enterprise projects (e.g. GLOBEMEN) and systems (e.g. SAP) are described.

The fundamental approach for extended enterprise collaboration is shown in chapter 5. The main components of this concept are the business process engines which have to execute the (extended) enterprise business processes. To fulfill the activities of such a process, Web services provided by enterprise information systems or sub-processes are used. The features of this new concept comply with the extended enterprise requirements for an information technology system.

To cast the developed approach into software, in chapter 6 the technical specification for the software system is described. The IT architecture is composed of four tiers: client tier, role-based presentation tier, business process automation tier, and service tier. The key component - the business process engine - is located inside the business process automation tier. Due to the requirement "cost-effective", an open source engine is selected and deployed into the IT environment.

To proof the developed concept, a prototype realization has been done within the European research and development project "FLoCI-EE" (chapter 7). Therefore an exemplary business scenario has been developed, modeled, and implemented. J2EE (Java 2 Enterprise Edition) is the core technology used for the implementation environment. The final part - profitability appraisement - should outline a realistic exploitation strategy for the developed software system.

Although many realization fruits have been achieved in this dissertation, there are still some more interesting research and development areas for the future:

- The design of a security concept for extended enterprises to enable secure exchange and usage of confidential business information. Very important for such a system is the international acceptance of defined standards and interfaces.
- An approach for integrated, consistent, and standardized information flow between the business information systems and holonic production control level, that will become the future technology for the enterprise field level.
- Identification of essential services including their functionality to support the business processes along the whole product life cycle.
- Analyzing and defining of standard business processes that can be used like puzzle pieces. First efforts are done within the ebXML initiative.

The list can be arbitrarily augmented because changes in this specific research area are happen breathtakingly fast. This result in the appearance of new standards, initiatives, projects, and systems in more and more shorter intervals. Such a dynamic environment offers many options to carry out research and development projects.

List of Figures

1.1 Communication Disruptions between Stages of Product Life Cycle 8

1.2 Information demand for decisions and process flows [FH02][Häb03] 10

1.3 Positioning the Emerging System 13

2.1 "Holistic engineering" as the interaction between multiple specialist disciplines from the engineering sciences and economics [ZF00] 22

3.1 Information Systems in the Product Life Cycle [ZF00] 26

3.2 Product data management system [ZF00] 28

3.3 ERP system in an industrial enterprise [ZF00] 30

3.4 Element of a supply chain [ZF00] 32

3.5 CRM system [ZF00] 33

3.6 Different Types of Enterprise Networking [CMA99b] 35

3.7 General Life cycle of Virtual Enterprises [CMA97] 40

3.8 Extended Enterprise Life Cycle [Für02] 41

3.9 WfMC's Workflow Management System Reference Model [Wor99] 45

3.10 Relationships between Basic Terminology [Wor99] 46

3.11 Model Views and UML Diagrams [Alh98] 48

3.12 Web Service Model [SFW02b] 54

3.13 The Structure of SOAP [SFW02a] 56

3.14 Description of WSDL Elements [SFW02a] 58

3.15 Core Information Types of UDDI [FSW02a] 59

4.1 The Enterprise Reality: Network Spaghetti [Pez02] 62

4.2 Types of Middleware [Uni03] 64

4.3 J2EE Multitiered Application [Uni03] 67

4.4 General Approach for a VE environment by PRODNET II [Pro03b] 72

4.5 VERAM - Virtual Enterprise Reference Architecture and Methodology [Glo03] 74

4.6 VERA - Virtual Enterprise Reference Architecture [PvdB00] . . 75

4.7 Three-Tier Client/Server Architecture of SAP R/3 [MZ98] . . . 77

4.8 SAP R/3 Applications and their Technological Basis [MZ98] . . 78

4.9 Interaction between Front- and Back-Office Applications 80

4.10 Architecture of mySAP.com [SAP02] 81

4.11 Siebel eBusiness Application Integration [Güm00] 82

4.12 Hype Cycle by Gartner [DS03] 84

4.13 History of Web Services 87

5.1 Principle of Extended Enterprises 91

5.2 Horizontal and Vertical Integration (according to [ZF00]) 94

5.3 Horizontal Integration of Intra-/Inter-Business Processes 96

5.4 Manual Access to different Business Information Systems 97

5.5 Extended Enterprise Integration Concept 100

5.6 Shop Floor Service Architecture [GSWF03] 102

5.7 Role Mapping inside Extended Enterprise 103

5.8 Basically Approach for Extended Enterprise Collaboration . . . 105

5.9 Qualitative Illustration of Controlled Extended Enterprise . . . 107

6.1 IT Architecture 109

6.2 MVC Architecture [Sun03b] 111

6.3 Business Process Automation Tier enables Horizontal Integration 113

6.4 Service Body 113

6.5 Service Engine Components of OFBiz [OFB03] 121

6.6 Business Process Engine Interface 122

6.7 Collaboration Diagram to Get the Worklist 123

6.8 Collaboration Diagram to Complete the Work Item 124

6.9 Class Diagram of Business Process Interface 125

7.1 Use Case Diagram for the Role EECustomer (@ customer) . . . 132

7.2 Use Case Diagram for the Use Case "Order Specification" . . . 133

7.3 Use Case Diagram for the Use Case "Design New Product" . . . 134

7.4 Use Case Diagram for the Role EESalesman_EE (@ coordinator) 134

7.5 Use Case Diagram for the Use Case "Bid Proposal Management" 135

7.6 Aggregated Use Case Diagram 136

7.7 Activity Diagram of Main Business Process (Part 1) 139

7.8 Activity Diagram of Main Business Process (Part 2) 140

7.9 Business Process Modeling using XPDL 143

7.10 XML Schema for XPDL Element "Package" [Wor02] 144

7.11 XML Schema for XPDL Element "Participants" [Wor02] 145

7.12 XML Schema for XPDL Element "WorkflowProcesses" [Wor02] 146

7.13 Implementation Environment for the Prototype 147

7.14 JSP-To-Servlet Flow [IBM03] 149

7.15 Struts Overview [Pan01] 150

7.16 Types of Enterprise JavaBeans 151

7.17 Time Measure Points for Performance Measurement 153

7.18 Measurement Results for Web Service Time 155

7.19 Screenshot - Configuration of Keypad 156

7.20 Screenshot - Input of Order Conditions 157

7.21 FLoCI-EE Support the Product Life Cycle 161

7.22 SMEs Price Acceptance for Enterprise Software [Mic02] 163

7.23 Target Market for FLoCI-EE [Mic02] 164

7.24 SWOT Analysis of FLoCI-EE [Mic02] 165

7.25 Break-Even Diagram 168

List of Tables

1.1 Data Transmission Structure [ZF00] 9

3.1 Comparison of Modeling Standards [Tec03] 50

3.2 Web Service Stack [Ira02] 55

5.1 Estimated Figures for Attributes of five Products [UE00] 92

6.1 Most Important Decision Criteria 118

6.2 Comparison of Open Source Workflow Engines 119

7.1 Sub-Elements of Element "Package" [Wor02] 144

7.2 Sub-Elements of Element "Participant" [Wor02] 145

7.3 Sub-Elements of Element "WorkflowProcess" [Wor02] 145

Bibliography

[AKY01] Cliff Allen, Deborah Kania, and Beth Yaeckel. *One-to-One Web Marketing: Build a Relationship Marketing Strategy One Customer at a Time.* Wiley Computer Publishing, 2001.

[Alh98] Sinan Si Alhir. *UML in a Nutshell.* O'Reilly & Associates Inc., 1998.

[All00] Rob Allen. Workflow: An Introduction. In Layna Fischer, editor, *Workflow Handbook 2001.* WfMC, 2000.

[Alo03] Alorie Gilbert. Enterprise: Clash of the titans. `http://news.com.com/2009-1001-961436.html`, 9 October 2003.

[Apa03a] Apache Software Foundation. The Apache Struts Web Application Framework. `http://jakarta.apache.org/struts/`, 17 March 2003.

[Apa03b] Apache Software Foundation. Apache Tomcat Project Homepage. `http://jakarta.apache.org/tomcat/index.html`, 18 March 2003.

[BMMF97] P. Bertok, M. Mantyla, J. McGovern, and G. Fernandez. Working group report on information infrastructure for global and virtual enterprises. In *Proceedings of the Sixth IEEE Workshops*

on Enabling Technologies: Infrastructure for Collaborative Enterprises, pages 62–66, 1997.

[Bra00] Thomas Brandstätter. *Analyse des Produktlebenslaufes hinsichtlich der Anforderungen an DV-Systeme und Vergleich mit aktuellen Systemen.* Master diss., Department of Electrical Engineering, Vienna University of Technology, 2000.

[BRJ99] Grady Booch, James Rumbaugh, and Ivar Jacobson. *The Unified Modeling Language User Guide.* Addison Wesley Longman Inc., 1999.

[Bus01] Business Process Management Initiative. *Business Process Modeling Language, Working Draft 0.4*, 8 March 2001. `http://www.bpmi.org/`.

[CMA97] L.M. Camarinha-Matos and H. Afsarmanesh. Virtual Enterprises: Life cycle supporting tools and technologies. In A. Molina, J. Sanchez, and A. Kusiak, editors, *Handbook of Life Cycle Engineering: Concepts, Tools and Techniques.* Chapman and Hall, 1997.

[CMA99a] L.M. Camarinha-Matos and H. Afsarmanesh. The PRODNET Architecture. In L.M. Camarinha-Matos and H. Afsarmanesh, editors, *Infrastructures for Virtual Enterprises: Networking Industrial Enterprises*, pages 109–126, 1999.

[CMA99b] L.M. Camarinha-Matos and H. Afsarmanesh. The virtual enterprise concept. In L.M. Camarinha-Matos and H. Afsarmanesh, editors, *Infrastructures for Virtual Enterprises: Networking Industrial Enterprises*, pages 3–14, 1999.

[Dav92] Thomas H. Davenport. *Process Innovation: Reengineering Work Through Information Technology.* Harvard Business School Press, Boston, USA, 1992.

[DLT00] Dean Dreibelbis and Tony Lacy-Thompson. Interfacing with SAP R/3. *eAI Journal*, pages 20–25, March 2000.

[DP02] Stefan Denninger and Ingo Peters. *Enterprise JavaBeans 2.0.* Addison-Wesley, 2002.

[DS03] Oliver Diedrich and Peter Siering. Zwischen Hype und Wirklichkeit. *c't - Magazin für Computer Technik*, Heft 6:146–147, 2003.

[EGH+01] Ulrich Eisert, Kerstin Geiger, Gerd Hartmann, Helmut Ruf, Stephan Schindewolf, and Ulrich Schmidt. *mySAP Product Lifecycle Management.* SAP Press, 2001.

[Enh03] Enhydra.org. Graphical Java Workflow Process Editor. `http://jawe.enhydra.org/`, 25 March 2003.

[EP00] Hans-Erik Eriksson and Magnus Penker. *Business Modeling with UML: Business Patterns at Work.* Wiley Computer Publishing, 2000.

[Esc96] Rolf Eschenbach. *Controlling.* Schäffer-Poeschel Verlag, Stuttgart, 1996.

[FH02] Karl Fürst and Stephen Häberle. Neue Potenziale durch die automatisierte Unternehmenskommunikation. *Elektrotechnik und Informationstechnik ÖVE Verbandszeitschrift*, pages 294–300, September 2002.

[FRZ01] Karl Fürst, Odilia Rodrigues, and Gerfried Zeichen. Requirements and basic Technologies for Extended Enterprises. *Elektrotechnik und Informationstechnik ÖVE Verbandszeitschrift*, pages 590–597, November 2001.

[FSW02a] Karl Fürst, Thomas Schmidt, and Gerald Wippel. Enabling collaborative engineering with Web services. In *Proceedings of the 2002 IEEE International Conference on Intelligent Engineering Systems*, pages 501–506, Opatija, Croatia, 26–28 May 2002. Faculty of Organization and Informatics, University of Zagreb, Croatia.

[FSW02b] Karl Fürst, Thomas Schmidt, and Gerald Wippel. Managing Access in Extended Enterprise Networks. *IEEE Internet Computing*, pages 67–74, September/October 2002.

[Für02] Karl Fürst. *Agile Extended Enterprise Systems: Total Process-Flows for Industry.* Habilitation thesis, Department of Electrical Engineering and Information Technology, Vienna University of Technology, 2002.

[GG02] Sumantra Ghoshal and Lynda Gratton. Integrating the Enterprise. *MIT Sloan Management Review*, pages 31–38, Fall 2002.

[Glo03] GLOBEMEN - Global Engineering and Manufacturing in Enterprise Networks. `http://globemen.vtt.fi`, 21 August 2003.

[GNP95] Steven L. Goldman, Roger N. Nagel, and Kenneth Preiss. *Agile Competitors and Virtual Organisations.* Van Nostrand Reinhold, 1995.

[Gor99] H.T. Goranson. *The agile virtual Enterprise: Cases, Metrics, Tools.* Quorum Books, 1999.

[GP99] Charles F. Goldfarb and Paul Prescod. *XML-Handbuch.* Prentice Hall, München, 1999.

[GS02] Stefan Graebe and Michael Schleicher. Holistische Produktion. *Elektrotechnik und Informationstechnik ÖVE Verbandszeitschrift*, pages 274–281, September 2002.

[GSWF03] Klaus Glanzer, Thomas Schmidt, Gerald Wippel, and Karl Fürst. Shop Floor Services for Automation and Integration. In *Proceedings of the 2003 IEEE International Conference on Intelligent Engineering Systems*, Assiut - Luxor, Egypt, 4–6 March 2003.

[Gui03] Jeff Guitard. *Jakarta Tomcat Performance Benchmark.* TheServerSide.com, 19 March 2003. `http://www.theserverside.com/reviews/index.jsp`.

[Güm00] Helmuth Gümbel. *Integrating eBusiness with SAP R/3: A Comparison of Key Vendors.* Strategy Partners International, November 2000. `http://www.strategypartners.com/IntCRM.pdf`.

[Häb03] Stephan Häberle. *???* Ph.d. diss., Department of Electrical Engineering and Information Technology, Vienna University of Technology, 2003.

[Ham96] Michael Hammer. *Beyond Reengineering: How the Processed-Centered Organization is Changing Our Work and Our Lives.* Harper Business, 1996.

[Ham98] Micheal Hammer. *Beyond Reengineering: How the Processed-Centered Organization is Changing Our Work and Our Lives.* HarperCollins, 1998.

[Hen03] Henry Ford Museum. The Model T. `http://www.hfmgv.org/exhibits/showroom/1908/model.t.html`, 14 January 2003.

[Her97] Jose Antonio Hernandez. *The SAP R/3 Handbook.* McGraw-Hill, New York, 1997.

[IBM01] IBM. *Web Services Flow Language, Version 1.0*, May 2001. `http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf`.

[IBM02] IBM developerWorks. *Business Process Execution Language for Web Services, Version 1.0*, 31 July 2002. `ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf`.

[IBM03] IBM developerWorks. Struts: an open-source MVC implementation. `http://www-106.ibm.com/developerworks/library/j-struts/index.html`, 14 March 2003.

[II93] B. Joseph Pine II. *Mass Customization: The New Frontier in Business Competition.* Harvard Business School Press, Boston, Massachusetts, 1993.

[Int03] Intersect Software. *White Paper: Enterprise Engineering Management*, 18 February 2003. `http://www.intersectsoft.com/downloads/Intersect_EEM_Defined_White_Paper.pdf`.

[Ira02] Romin Irani. Architecture for Web Services. In *Professional Java Web Services.* Wrox Press Ldt, Birmingham, UK, 2002.

[IX-03a] Prozesse in Unternehmen. iX - Magazin für professionelle Informationstechnik, Heft 4/2003. page 44.

[iX 03b] iX - Magazin für professionelle Informationstechnik. *Die größten Softwarehersteller*, September 2003. page 36.

[JBB94] Jr. Jordan, J.A. Bunce, and P.M. Bunce. Requirements for next generation manufacturing systems (ngms). In *Proceedings of the Sixteenth IEEE/CPMT International Symposium on Electronics Manufacturing Technology Symposium*, volume 1, pages 247–252, 1994.

[JBo03] JBoss Group. JBoss Homepage. `http://www.jboss.org/`, 18 March 2003.

[jBp03] Java Business Process Management Homepage. `http://jbpm.org/`, 24 March 2003.

[JBS97] Stefan Jablonski, Markus Böhm, and Wolfgang Schulze. *Workflow Management: Entwicklung von Anwendungen und Systemen - Facetten einer neuen Technologie.* dpunkt-Verlag, Heidelberg, 1997.

[Joh02] Lauren Keller Johnson. New Views on Digital CRM. *MIT Sloan Management Review*, page 10, Fall 2002.

[KF98] Angel Kwolek-Folland. *Engendering Business: Men and Women in the Corporate Office, 1870-1930.* The Johns Hopkins University Press, Baltimore and London, 1998.

[Kla00] Walter Klambauer. *Produktdatenmanagement in virtuellen Unternehmen.* Master diss., Department of Electrical Engineering, Vienna University of Technology, 2000.

[KM98] G.L. Kovacs and I. Mezgar. A distributed planning network for manufacturing systems management. In *Proceedings of the*

IEEE International Conference on Robotics and Automation, volume 2, pages 1787–1792, 1998.

[KR99] Ravi Kalakota and Marcia Robinson. *e-Business 2.0: Roadmap for Success.* Addison-Wesley, 1999.

[Lan98] Thomas Lanner. *Dynamic Product Development.* Ph.d. diss., Department of Electrical Engineering, Vienna University of Technology, 1998.

[Lei02] Herwig Leinfellner. Holistic Engineering - Entwicklungsvorgänge in ganzheitlicher Sicht. *Elektrotechnik und Informationstechnik ÖVE Verbandszeitschrift*, pages 260–263, September 2002.

[Lin00] David S. Linthicum. *Enterprise Application Integration.* Addison Wesley Longman, 2000.

[MH00] Richard Monson-Haefel. *Enterprise JavaBeans.* O'Reilly, 2000.

[Mic02] Michael Merler and Martin Holzleitner and Jana Maderova and Michael Kamleitner. FLoCI-EE: Business Proposal, 2002.

[Mic03] Michael Kaib, Booz Allen & Hamilton. Enterprise Application Integration. `http://www.objectarchitects.de/eai/talks.htm`, 19 February 2003.

[MZ98] Florian Matthes and Stephan Ziemer. *Understanding SAP R/3 - A Tutorial for Computer Scientists.* Technical University Hamburg-Harburg, March 1998. `http://www.sts.tu-harburg.de/slides/1998/03-98-MaZi-SAP-EDBT98.pdf`.

[NHH+00] Grant Norris, James R. Hurley, Kenneth M. Hartley, John R. Dunleavy, and John D. Balls. *E-Business and ERP: Transforming the Enterprise.* PricewaterhouseCoopers, 2000.

[NII03] National Industrial Information Infrastructure Protocols (NIIIP). `http://www.niiip.org`, 14 August 2003.

[Obj01] Object Management Group. *OMG Unified Modeling Language Specification, Version 1.4*, September 2001. `http://www.omg.org/cgi-bin/doc?formal/01-09-67.pdf`.

[ÖFA00] Hubert Österle, Elgar Fleisch, and Rainer Alt. *Business Networking: Shaping Enterprise Relationships on the Internet.* Springer Verlag, 2000.

[OFB03] OFBiz.org. The Open For Business Project. `http://www.ofbiz.org`, 14 March 2003.

[O'L00] Daniel E. O'Leary. *Enterprise Resource Planning Systems.* Cambridge University Press, Cambridge, UK, 2000.

[Org03] Organization for the Advancement of Structured Information Standards (OASIS). Universal Description, Discovery, and Integration (UDDI). `http://www.uddi.org`, 28 February 2003.

[Pan01] Sathya Narayana Panduranga. The Struts Framework. In *Professional JSP Site Design.* Wrox Press Ldt, Birmingham, UK, 2001.

[Pat99] Andrew Patzer. Introduction to Servlets. In *Professional Java Server Programming.* Wrox Press Ldt, Birmingham, UK, 1999.

[PB99] Andrew Patzer and Tim Briggs. Introduction to JavaServer Pages. In *Professional Java Server Programming.* Wrox Press Ldt, Birmingham, UK, 1999.

[Pez02] Massimo Pezzini. Application Integration: The Inner Engine of Real-Time Enterprise. In *Gartner Symposium ITxpo 2002*, Cannes, France, 4–7 November 2002.

[PR97] Don Peppers and Martha Rogers. *Enterprise One-to-One: Tools for Building unbreakable Customer Relationships in the Interactive Age.* Piatkus Ltd., London, 1997.

[Pre03] Jürgen Pretschuh. *Quantifizierungsmethoden für ein regelbares Holistic Manufacturing.* Ph.d. diss., Department of Electrical Engineering and Information Technology, Vienna University of Technology, 2003.

[Pro03a] PROFIBUS Documentation. `http://www.profibus.com/profibus.html`, 13 August 2003.

[Pro03b] PRODNET II - Production Planning and Management in an Extended Enterprise. `http://www.uninova.pt/~prodnet/index.html`, 14 August 2003.

[PT02] Jürgen Pretschuh and Rene Traxl. Ein neuer Modellansatz zur Quantifizierung von Unternehmensfunktionen für Holistic Engineering. *Elektrotechnik und Informationstechnik ÖVE Verbandszeitschrift*, pages 286–293, September 2002.

[Pun95] Dieter Punzengruber. *Die Fabrik als dynamisches System.* Ph.d. diss., Department of Electrical Engineering, Vienna University of Technology, 1995.

[PvdB00] Jens Dahl Pedersen and Roelof J. van den Berg. Supporting Partner Selection for Virtual Enterprises. In John P.T. Mo and Laszlo Nemes, editors, *Global Engineering, Manufacturing and Enterprise Networks*, pages 95–102, 2000.

[Pyk01] Jon Pyke. E-Process Technology. In *AIIM 2001*, New York, USA, 30 April – 3 May 2001.

[Ran03] Ranger Technical Resources. IT Solutions - Enterprise Application Integration. `http://www.rangertech.com/its_eai.html`, 19 February 2003.

[Rau01] Michael Rauch. *Holistic product development.* Ph.d. diss., Department of Electrical Engineering and Information Technology, Vienna University of Technology, 2001.

[Rog97] Dale Rogerson. *Inside COM.* Microsoft Press, Redmond, 1997.

[Rom99] Ed Roman. *Mastering Enterprise JavaBeans and the Java 2 Platform, Enterprise Edition.* Wiley Computer Publishing, 1999.

[Ros98] Jeremy Rosenberger. *CORBA in 14 Tagen: Ihr professioneller Schritt-für-Schritt-Einstieg in CORBA.* SAMS, 1998.

[SAP02] SAP AG. *SAP R/3 Enterprise*, 2002. `http://www.sap.com/southafrica/partners/pres/R3%20Enterprise%200verviews/R3%20Enterprise%20White%20Paper.pdf`.

[Sch01] August-Wilhelm Scheer. *ARIS - Modellierungsmethoden, Metamodelle, Anwendungen*, volume 4. Springer Verlag, 2001.

[Sel93] Siegfried Selberherr. *Betriebssysteme und Anwendungssoftware.* Institut für Mikroelektronik, Wien, 1993.

[SFW02a] Thomas Schmidt, Karl Fürst, and Gerald Wippel. e-Technologies for the Web Business. In *Proceedings of the 2002 IEEE International Engineering Management Conference*, volume I, pages 96–100, Cambridge, UK, 18–20 August 2002. IEEE Engineering Management Society.

[SFW02b] Thomas Schmidt, Karl Fürst, and Gerald Wippel. Agile e-Business Process Assembly and Development. In Vladimir Marik, Luis M. Camarinha-Matos, and Hamideh Afsarmanesh, editors, *Knowledge and Technology Integration in Production and Services*, pages 375–382. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002.

[Sha00] Michael J. Shaw. Electronic Commerce: State of the Art. In Micheal Shaw, Robert Blanning, Troy Strader, and Andrew Whinston, editors, *Handbook on Electronic Commerce.* Springer-Verlag, 2000.

[SLKSL00] David Simchi-Levi, Philip Kaminsky, and Edith Simchi-Levi. *Designing and Managing the Supply Chain.* McGraw-Hill Higher Education, 2000.

[Sma03] Smart Solutions Ltd. *White Paper: Choosing a PDM Solution*, 18 February 2003. `http://www.mlc-cad.com/documents/choosingapdm.pdf`.

[Ste03] Manie Steyn. *An Implementation of a Real-Time Message-Oriented Middleware.* Communications Computer Intelligence Integration Systems Ltd., 20 February 2003. `http://www.ccii.co.za/rnd/ims02.pdf`.

[Sun03a] Sun Microsystems. Sun Open Net Environment (Sun ONE). `http://www.sun.com/software/sunone`, 28 February 2003.

[Sun03b] Sun Microsystems Inc. Designing Enterprise Applications with the J2EE Platform. `http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/app-arch/app-arch2.html`, 14 March 2003.

[Tay98] Frederick Winslow Taylor. *The Principles of Scientific Management.* Dover Pubns, 1998.

[Tec03] Technical University Eindhoven. *Standards' Evaluation*, 27 February 2003. `http://tmitwww.tm.tue.nl/research/patterns/start_stand.htm`.

[Tex00] Texas.Net. ERP vs. SCM - What's the difference? `http://lonestar.texas.net/~eallen/erp/ERP.htm`, 31 January 2000.

[Tha01] Satish Thatte. *XLANG - Web Services for Business Process Design.* Microsoft, 2001. `http://www.gotdotnet.com/team/xml%5Fwsspecs/xlang%2Dc/`.

[Tra02] Rene Traxl. *Ein neues Decision Support System zur Analyse und Optimierung integrierter Fertigungsanlagen.* Ph.d. diss., Department of Electrical Engineering and Information Technology, Vienna University of Technology, 2002.

[UE00] Karl T. Ulrich and Steven D. Eppinger. *Product Design and Development.* McGraw-Hill, 2000.

[Uni03] University Erlangen. *Enterprise Application Integration: Building Next Generation Enterprises*, 19 February 2003.

http://www.wi3.uni-erlangen.de/lehre/lv/ws2001/BIT/SEuTM-BIT-ws01-8b.pdf.

[vdA98] W.M.P. van der Aalst. Modeling and Analyzing Interorganizational Workflows. In *Proceedings of the IEEE International Conference on Application of Concurrency to System Design*, pages 262–272, 1998.

[vdAtHKB03] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. *Workflow Patterns.* Technical University Eindhoven, 27 February 2003. http://tmitwww.tm.tue.nl/research/patterns/wfs-pat-2002.pdf.

[Wei94] Alexander Weinmann. *Regelungen, Bd.1, Systemtechnik linearer und linearisierter Regelungen auf anwendungsnaher Grundlage.* Springer, Wien, 1994.

[Wil03] Willamette University. Henry Ford. http://www.willamette.edu/~fthompso/MgmtCon/Henry_Ford.html, 14 January 2003.

[Wir03] Wirtschaftskammer Wien. 10 sehr gute Gründe für Open Source Software. http://www.open-source.co.at, 7 March 2003.

[Wor98] Workflow Management Coalition. *Workflow Management Application Programming Interface (Interface 2&3) Specification*, July 1998. http://www.wfmc.org/standards/docs/if2v20.pdf.

[Wor99] Workflow Management Coalition. *Workflow Management Coalition: Terminology & Glossary*, February 1999. http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf.

[Wor00] World Wide Web Consortium (W3C). *Simple Object Access Protocol (SOAP) 1.1, W3C Note*, 8 May 2000. `http://www.w3.org/TR/SOAP/`.

[Wor01] World Wide Web Consortium (W3C). *Web Services Description Language (WSDL) 1.1, W3C Note*, 15 March 2001. `http://www.w3.org/TR/wsdl/`.

[Wor02] Workflow Management Coalition. *Workflow Process Definition Interface - XML Process Definition Language, Version 1.0*, 25 October 2002. `http://www.wfmc.org/standards/docs/TC-1025_10_xpdl_102502.pdf`.

[Wor03] Working Group on Libre Software. Free Software / Open Source: Information Society Opportunities for Europe? `http://eu.conecta.it/paper/paper.html`, 7 March 2003.

[WSB02] Peter Weill, Mani Subramani, and Marianne Broadbent. Building IT infrastructure for Strategic Agility. *MIT Sloan Management Review*, pages 57–65, Fall 2002.

[Zei95a] Gerfried Zeichen. New Trends in Manufacturing. In *Proceedings of the BRITE-EURAM Conference*, Vienna, Austria, 11-13 October 1995.

[Zei95b] Gerfried Zeichen. Holonic - System of the future? In *Opening Symposium*, Engineering Center Graz, Austria, 25 June 1995.

[ZF00] Gerfried Zeichen and Karl Fürst. *Automatisierte Industrieprozesse.* Springer, Wien, New York, 2000.

[ZH98] Gerfried Zeichen and Paul G. Huray. Creative strategies of businesses with the holistic eigensolution in manufacturing industries. In *Intelligent Systems in Design and Manufacturing,*

SPIE Proceedings Vol. 3517, pages 25–34, Boston, USA, 1-6 November 1998.

Appendix A

List of Abbreviations

This appendix contains a list of frequently used abbreviations. Most of the terms expanded in this appendix are contained in the glossary (Appendix B).

ACIN	Automation Control Institute
ARIS	Architecture of Integrated Information Systems
ANSI	American National Standards Institute
API	Application Programming Interface
ASP	Application Service Provider
B2B	Business-to-Business
B2C	Business-to-Customer
BPEL4WS	Business Process Execution Language for Web Services
BPMI	Business Process Management Initiative
BPML	Business Process Modeling Language
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
CAP	Computer Aided Planning
CGI	Common Gateway Interface

CORBA	Common Object Request Broker Architecture
CRM	Customer Relationship Management
(D)COM	(Distributed) Component Object Model
DRBAC	Distributed Role-based Access Control
DRBAC-EE	DRBAC for Extended Enterprises
e-business	Electronic Business
e-commerce	Electronic Commerce
EAI	Enterprise Application Integration
EDI	Electronic Data Interchange
EDIFACT	EDI for Administration, Commerce and Transport
EE	Extended Enterprise
EJB	Enterprise JavaBeans
ERP	Enterprise Resource Planning
FLoCI-EE	Flexible Low-Cost Internet Extended Enterprise
FTP	File Transfer Protocol
GERAM	Generalized Enterprise Reference Architecture and Methodology
GLOBEMEN	Global Engineering and Manufacturing in Enterprise Networks
GUI	Graphical User Interface
HMS	Holonic Manufacturing System
HTTP	Hypertext Transfer Protocol
HTTPs	Secure HTTP
IDOC	Intermediate Document
IIOP	Internet Inter-ORB Protocol
IEEE	Institute of Electrical and Electronics Engineers
IMS	Intelligent Manufacturing Systems
ISO	International Standards Organisation

IT	Information Technology
J2EE	Java 2 Enterprise Edition
JDBC	Java Database Connectivity
JIT	Just-In-Time
JSP	Java Server Pages
LAN	Local Area Network
MOM	Message-Oriented Middleware
MVC	Model View Control
NIIIP	National Industrial Information Infrastructure Protocols and Related Projects
OASIS	Organization for the Advancement of Structured Information Standards
ODETTE	Organization for Data Exchange Through Tele-transmission in Europe
OFBiz	Open For Business
OMG	Object Management Group
ORB	Object Request Broker
PC	Personal Computer
PDM	Product Data Management
PLM	Product Lifecycle Management
RMI	Remote Method Invocation
RPC	Remote Procedure Call
SAP	Systems, Applications and Products in Data Processing
SCM	Supply Chain Management
SME	Small and Medium Enterprise
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol

SQL	Structured Query Language
SSL	Secure Sockets Layer
TP	Transaction Processing
UDDI	Universal Description, Discovery, and Integration
UI	User Interface
UML	Universal Modeling Language
VE	Virtual Enterprise
VERAM	Virtual Enterprise Reference Architecture and Methodology
W3C	World Wide Web Consortium
WAPI	Workflow Application Programming Interface
WfMC	Workflow Management Coalition
WfMS	Workflow Management System
WSDL	Web Service Description Language
WSFL	Web Service Flow Language
XML	eXtensible Markup Language
XPDL	XML Process Definition Language
XSLT	eXtensible Stylesheet Language Transformations

Appendix B

Glossary

One of the main problems of this research area is the missing mathematical basis. Therefore, it is a must to define the used terms exactly.

Business Process

According to DIN 66201, *"a process is the totality of activities in a system which are influencing each other and by which material, energy, or information is transformed, transported, or stored."* [ZF00]

Another definition for the term "process" can be found in [Dav92]: *"a process is simply a structured, measured set of activities designed to produce a specified output for a particular customer or market. It implies a strong emphasis on how work is done within an organization, in contrast to a product focus's emphasis on what. A process is thus a specific ordering of work activities across time and place, with a beginning, an end, and clearly identified inputs and outputs: a structure for action."*

The terms "business process" and "industrial process" are frequently used in the same context, because *"industrial processes can be said to be equivalent*

to business processes, however, the term business process also applies to trading firms and commercial enterprises". [ZF00]

Business Process Description Language

"The representation of a business process in a form which supports automated manipulation, such as modeling, or enactment by a workflow management system. The process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated IT applications and data, etc." [Wor99]

Customer Relationship Management (CRM)

"CRM systems are information, consulting, quotation and ordering systems for sales and marketing." [ZF00]

Enterprise Application Integration (EAI)

According [Lin00], *"EAI is the unrestricted sharing of data and business processes among any connected applications and data sources in the enterprise".* And for the Gartner Group, *"enterprise application integration means making independently designed systems work together"* [Uni03].

Extended Enterprise

Extended Enterprises (EE) are: [Für02]

1. *temporary or permanent networks of*

2. *independent enterprises*
3. *including a coordinator*
4. *cooperating with the aim to*
5. *design, manufacture, and sell a product or service*
6. *in a project-oriented way*
7. *independent of enterprise borderlines,*
8. *automated inter-enterprise communication through the usage of information technology, and*
9. *communicating via Internet-related technologies like XML-based Web Services.*

Enterprise Resource Planning (ERP)

"ERP systems support and automate the organizational planning, control and monitoring of production and assembly sequences, from order receipt (by sales), through capacity scheduling, production scheduling, production data acquisition collection, right up to control of dispatch." [ZF00]

Holistic Engineering

"Holistic engineering enables the inventor or developer of a new product to clearly and efficiently define the structure of and accept responsibility for the production and marketing subprocesses that follow product development" [ZF00].

Manufacturing

"Manufacturing is defined as the sum of all subfunctions that are necessary for the manufacture of a product:

- *product development and design (realised in the product specification list and relevant drawings),*
- *process development (defining the parts production and assembly procedures),*
- *operational tasks such as parts production, assembly, filling, quality assurance and packaging for the whole product,*
- *order processing and communication with sales and suppliers."* [ZF00]

Product Data Management (PDM)

"The role of product data management systems is to manage all of the product data in the product creation process. They form a higher level platform for access to the product data and corresponding documents (drawings, parts lists, NC programs, etc.) as well as for backup, control, distribution, and archiving." [ZF00]

Supply Chain Management (SCM)

"SCM systems are IT tools that support effective management of the supply chain." [ZF00]

Workflow

"The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules" [All00].

Workflow Management System

"A system that defines, creates and manages the execution of workflows through the use of software, running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participants and, where required, invoke the use of IT tools and applications" [Wor99].

Appendix C

List of Publications

Master Thesis

Thomas Schmidt. *Internet Electronic Data Interchange für flexible Logistik in einem Motorenwerk.* Master Thesis, Department of Electrical Engineering, Vienna University of Technology, 1999.

Articles in Journals and Magazines

Karl Fürst and Thomas Schmidt. Turbulent markets need flexible supply chain communication. In *Production Planning and Control*, volume 12, pages 525-533. Taylor and Francis Ltd., December 2001.

Karl Fürst, Thomas Schmidt, and Gerald Wippel. Managing Access in Extended Enterprise Networks. In *IEEE Internet Computing*, volume 6, number 5, pages 67-74, September/October 2002. IEEE Computer Society.

Conference Proceedings

Karl Fürst, Thomas Schmidt, and Franz Auinger. Internet electronic data interchange. In *CPI 1999, Conference Proceedings*, pages 341-350, Tangier, Morocco, 25-26 November 1999. Faculte des Sciences et Techniques.

Karl Fürst and Thomas Schmidt. Internet electronic data interchange with XML and Java. In Pamela Gennusa, editor, *XML Europe 2000, Conference and Exhibition*, pages 803-809, Paris, France, 12-16 June 2000. GCA, Graphic Communications Association.

Karl Fürst and Thomas Schmidt. Low-cost Internet electronic data interchange for SME-logistics. In Jan Ola Strandhagen and Erlend Alfnes, editors, *Conference Proceedings, IFIP WG 5.7: Information and Communication Technology (ICT) in Logistics and Production Management*, pages 37-43, Tromso, Norway, 28-30 June 2000. The Department of Production and Quality Engineering, Norwegian University of Science and Technology.

Karl Fürst and Thomas Schmidt. Advanced supplier integration for an engine manufacturer. In *ISATA 2000, Advanced Manufacturing*, pages 69-76, Dublin, Ireland, 25-27 September 2000. ISATA-Düsseldorf Trade Fair, Epsom House.

Karl Fürst and Thomas Schmidt. XML-based Internet communication for advanced external logistics. In Brian Stanford-Smith and Paul T. Kidd, editors, *E-Business: Key Issues, Applications and Technologies*, pages 515-521, Amsterdam, The Netherlands, 2000. IOS Press.

Karl Fürst and Thomas Schmidt. Internet-based data interchange with XML. In Nina M. Berry, editor, *Network Intelligence: Internet-based Manufacturing, Conference Proceedings SPIE 2000*, volume 4208, pages 114-121, Boston, USA, 6-8 November 2000. SPIE - The International Society for Optical Engineering.

Karl Fürst and Thomas Schmidt. Low-cost system for supply chain management. In John P.T. Mo and Laszlo Nemes, editors, *Global Engineering, Manufacturing and Enterprise Networks*, pages 156-163, Dordrecht, The Netherlands, 2001. Kluwer Academic Publishers.

Karl Fürst and Thomas Schmidt. System to improve the process of parts delivery. In *Conference Proceedings WCTR 2001*, Seoul, Korea, 22-27 July 2001. WCTR-S - World Conference on Transport Research Society.

Karl Fürst, Thomas Schmidt, and Gerald Wippel. Enabler for the agile virtual enterprise. In Angappa Gunasekaran and Yahaya Y. Yusuf, editors, *Internet-based Enterprise Integration and Management*, volume 4566, pages 217-227, Newton, USA, 31 October - 1 November 2001. SPIE - The International Society for Optical Engineering.

Karl Fürst, Thomas Schmidt, and Gerald Wippel. Collaborative product engineering in the extended enterprise. In *Proceedings of the Product Data Technology Europe 2002*, pages 195-201, Turin, Italy, 7-9 May 2002. Quality Marketing Services.

Karl Fürst, Thomas Schmidt, and Gerald Wippel. Enabling collaborative engineering with Web services. In Alen Lovrencic and Imre J. Rudas, editors, *Proceedings of the 2002 IEEE International Conference on Intelligent Engineering Systems*, pages 501-506, Opatija, Croatia, 26-28 May 2002. Faculty of Organization and Informatics, University of Zagreb, Croatia.

Karl Fürst, Thomas Schmidt, and Gerald Wippel. e-service based business information systems. In Kulvant S Pawar, Frithjof Weber, and Klaus-Dieter Thoben, editors, *Proceedings of the 8th International Conference on Concurrent Enterprising: Ubiquitous Engineering in the Collaborative Economy*, pages 381-388, Rome, Italy, 17-19 June 2002. University of Nottingham, Centre for Concurrent Enterprising.

Thomas Schmidt, Karl Fürst, and Gerald Wippel. e-technologies for the Web business. In *Proceedings of the 2002 IEEE International Engineering Management Conference*, volume I, pages 96-100, Cambridge, UK, 18-20 August 2002. IEEE Engineering Management Society.

Thomas Schmidt, Karl Fürst, and Gerald Wippel. Agile e-business process assembly and development. In Vladimir Marik, Luis M. Camarinha-Matos, and Hamideh Afsarmanesh, editors, *Knowledge and Technology Integration in Production and Services*, pages 375-382, Dordrecht, The Netherlands, 2002. Kluwer Academic Publishers.

Karl Fürst, Thomas Schmidt, and Gerald Wippel. Enterprise Networking with SOAP, UDDI, and WSDL. In *Proceedings of the 2002 IEEE/IFIP International Conference on Information Technology for Balanced Automation Systems in Manufacturing and Services*, Cancun, Mexico, 25-27 September 2002. IFIP - International Federation for Information Processing.

Klaus Glanzer, Thomas Schmidt, Gerald Wippel, and Karl Fürst. Shop Floor Services for Automation and Integration. In *Proceedings of the 7th IEEE International Conference on Intelligent Engineering Systems*, pages 539-544, Luxor, Egypt, 4-6 March 2003. Assiut University, Egypt.

Thomas Schmidt, Gerald Wippel, Klaus Glanzer, and Karl Fürst. Security System for Enterprise-spanning Collaboration. In Liang-Jie Zhang, editor, *Proceedings of the 2003 International Conference on Web Services*, pages 299-305, Las Vegas, USA, 23-26 June 2003. CSREA Press.

Klaus Glanzer, Thomas Schmidt, Gerald Wippel, and Christoph Dutzler. Integration of Shop Floor Holons with Automated Business Processes. In Vladimir Marik, Duncan McFarlane, and Paul Valckenaers, editors, *Proceedings of the 2003 First International Conference on Industrial Applications of Holonic and Multi-Agent Systems*, pages 146-155, Prague, Czech Republic, 1-3 September

2003. Springer Verlag.

Thomas Schmidt, Gerald Wippel, Klaus Glanzer, and Karl Fürst. Utilizing Open-Source Technologies Prevent Vendor-Dependent Business Solutions. In Paul Cunningham, Miriam Cunningham, and Peter Fatelnig, editors, *Building the Knowledge Economy - Issues, Applications, Case Studies*, part 2, pages 1283-1289, Amsterdam, The Netherlands, 2003. IOS Press.

Invited Sessions

Thomas Schmidt. Praxisbeispiel für XML-basiertes EDI. *Automobil-Cluster.* Steyr, Austria, 29 February 2000.

Karl Fürst and Thomas Schmidt. Neue Wege im Supply Chain Management: Internet electronic data interchange. *Future Network.* Vienna, Austria, 6 June 2000.

Thomas Schmidt. Developing a Security Architecture for an Extended Enterprise using Web Services. *Web Services Summit 2002.* London, 24 October 2002.

Appendix D

Curriculum Vitae

Persönliche Daten

Name:	Dipl.-Ing. Thomas Schmidt
geboren:	am 29.05.1974 in 2130 Mistelbach, Österreich
Familienstand:	verheiratet
Adresse:	1220 Wien, Langobardenstrasse 191/55

Ausbildung

1980 - 1984:	Volksschule, 2191 Gaweinstal
1984 - 1988:	Bundesrealgymnasium, 2230 Gänserndorf
1988 - 1993:	TGM, Abteilung Nachrichtentechnik - Biomedizinische Technik, 1200 Wien (Abschluss mit gutem Erfolg)
1993 - 1999:	Diplomstudium an der Technischen Universität Wien, Fakultät für Elektrotechnik und Informationstechnik, Studienzweig Computertechnik (Abschluss mit Auszeichnung)
1999 - 2003:	Doktoratsstudium an der Technischen Universität Wien,

Fakultät für Elektrotechnik und Informationstechnik

Grundwehrdienst

05/00 - 12/00: MilwEx (Militärisch wissenschaftlicher Experte)
in der ABC-Abwehrschule, 1020 Wien

Praxis

Juli 1990: OMV Erdgasbetrieb Auersthal, Ferialpraxis

August 1992: Siemens Medizinische Technik, Ferialpraxis

07/93 - 08/93: Kremsmüller, Ferialarbeiter

07/94 - 08/94: Post AG, Mittlerer Dienst

01/96 - 12/97: Elektro Silvestre, Technischer Zeichner

07/98 - 03/99: TBWA, eingesetzt bei Schwarzkopf und Henkel Cosmetics, Praktikant

12/99 - 04/00: TU-Wien, Institut für Flexible Automation, Forschungsassistent

01/01 - 06/01: TU-Wien, Institut für Flexible Automation, Forschungsassistent

07/01 - 11/01: TU-Wien, Institut für Flexible Automation, Universitätsassistent (Karenzvertretung)

seit 12/01: TU-Wien, Institut für Automatisierungs- und Regelungstechnik (vormals Institut für Flexible Automation), Forschungsassistent

Wien, November 2003