HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Electrical and Communications Engineering
S-72 Communications Engineering

Kari Koivuniemi

The Effect of Virtual Enterprises on Business-to-Business Integration

Master's thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology

Supervisor: Docent Timo Korhonen

Instructor: Docent Timo Korhonen

Espoo, 18th December 2008

HELSINKI UNIVERSITY OF TECHNOLOGY

Abstract of the Master's Thesis

Author: Kari Koivuniemi

Name of the Thesis: The Effect of Virtual Enterprises on Business-to-Business Integration

Date: 18.12.2008

Number of pages: 130 + viii

Department: Electrical and Communications Engineering

Professorship: S-72 Communications engineering

Supervisor: Docent Timo Korhonen

Instructor: Docent Timo Korhonen

This master's thesis is about virtual enterprises and business-to-business integration. In this thesis, virtual enterprise is defined as dynamic, as well as temporary, network of independent companies that come together to exploit a limited business opportunity. Their interaction is based on modern information and communication technology. Business-to-Business integration aims to solve the issues that arise from an attempt to automate heterogeneous cross-organizational business processes and flow of information between heterogeneous applications. Achieving proper interoperability between companies of a virtual enterprise poses considerable challenges for business-to-business integration, since a virtual enterprise is dynamic by nature and is likely to accentuate the issues that arise from heterogeneity of companies' applications and business processes.

This master's thesis is devoted to researching the effect of virtual enterprises on business-to-business integration. This effect is examined by defining the requirements that virtual enterprises set on business-to-business integration. Moreover, the main approaches to and the central components of business-to-business integration are uncovered and reflected to the requirements set by virtual enterprises. This master's thesis also includes the design of an intelligent routing solution to TradeXpress integration server.

Keywords: Virtual enterprise, business-to-business integration, intelligent routing solution

TEKNILLINEN KORKEAKOULU

Diplomityön tiivistelmä

Tekijä: Kari Koivuniemi
Työn nimi: Virtuaaliyritysten vaikutus yritysten väliseen sovellusintegraatioon Päivämäärä: 18.11.2008 Sivumäärä: 130 +viii
Osasto: Sähkö- ja tietoliikennetekniikan osasto Professuuri: S-72 Tietoliikennetekniikka
Työn valvoja: Dosentti Timo Korhonen Työn ohjaaja: Dosentti Timo Korhonen
Tämä diplomityö käsittelee virtuaaliyrityksiä ja yritysten välistä sovellusintegraatiota. Virtuaaliyritykset on tässä työssä määritelty useista yrityksistä muodostuviksi dynaamisiksi organisaatioiksi, jotka on luotu kestoltaan rajoitetun liiketoimintamahdollisuuden hyödyntämiseksi. Niiden toiminta perustuu keskeisesti modernin informaatioteknologian suomiin mahdollisuuksiin. Yritysten välinen sovellusintegraatio puolestaan pyrkii ratkaisemaan ongelmia, joita syntyy kun yritysten välisiä liiketoimintaprosesseja ja tiedonsiirtoa pyritään automatisoimaan kahden tai useamman epäyhtenäisen tietojärjestelmän välillä. Virtuaaliyritysten integrointi luultavasti tuottaa erityishaasteita perinteiselle yritysten väliselle sovellusintegraatiolle, sillä virtuaaliyritykset ovat rakenteeltaan sekä joustavia että muuttuvia ja niissä todennäköisesti korostuvat tietojärjestelmien epäyhtenäisyydestä johtuvat ongelmat. Tässä diplomityössä tutkitaan virtuaaliyritysten vaikutusta yritysten väliseen järjestelmäintegraatioon. Vaikutuksia tutkitaan selvittämällä ne erityisvaatimukset, jotka virtuaaliyritykset mahdollisesti asettavat yritysten väliselle sovellusintegraatiolle. Diplomityössä esitellään myös vallitsevat yritysten välisen sovellusintegraation ratkaisumallit ja niiden keskeiset komponentit, sekä pohditaan, miten ne soveltuvat virtuaaliyritysten integraatioon. Lisäksi diplomityössä kuvataan sisältöpohjaiseen sanomareititykseen perustuva ratkaisu, joka on suunniteltu toteutettavaksi TradeXpress-järjestelmään.
Avainsanat: Virtuaaliyritys, yritysten välinen sovellusintegraatio, sisältöön perustuva sanomanreititysratkaisu

TABLE OF CONTENTS

KEY CONCEPTS

Virtual enterprise: *A virtual enterprise is a dynamic, as well as temporary, network of independent companies that come together to exploit a limited business opportunity and base their interaction on modern ICT. (see Byrne 1993; Goranson 2003; Putnik et al. 2005)*

Virtual breeding environment: *Virtual breeding environments are long-term organizations that nurture the creation and operation of virtual enterprises. The carrying idea is that a group of corporations that have the potential and will to build virtual enterprises make preliminary cooperation arrangements* and form a virtual breeding environment*. When the conditions for cooperation are set, dynamic virtual enterprises can be derived from the virtual breeding environment. (see Putnik et al. 2005; Vallejos et al. 2007; Camarinha-Matos & Afsarmanesh 2007)*

Business-to-business integration: *Business-to-Business integration aims to solve the issues that arise from an attempt to automate heterogeneous cross-organizational business processes and flow of information between heterogeneous applications. The main obstacles that business-to-business integration tries to solve are a result of heterogeneity of trading partners' application semantics, information content and platforms. (see Samtani 2002; Linthicum 2004)*

Integration server: *The fundamental idea of integration servers is to enable all types of integration within an enterprise and between trading partners. Integration servers aim to provide an all-in-one solution for integration technology. (see Samtani 2002; Linthicum 2004)*

Intelligent routing solution: *Intelligent routing solution is a system designed to TradeXpress integration server that addresses an ISP's need to create flexible message-based interfaces between its customers and their business partners. The intelligent routing solution is founded on content-based routing.*

ABBREVIATIONS

AVE	Agile Virtual Enterprise
API	Application Programming Interfaces
B2Bi	Business-to-Business integration
BPEL4WS	Business Process Execution Language for Web Services
BPOI	Business Process -Oriented Integration
BPML	Business Process Modeling Language
BPMS	Business Process Management Systems
CA	Certificate Authority
DMZ	Demilitarized Zone
DTD	Document Type Definition
EAI	Enterprise Application Integration
ebXML	electronic business Extensible Markup Language
EDI	Electronic Data Interchange
EDIFACT	EDI for Administration, Commerce and Transport
EDI-INT	EDI over the Internet
ERP	Enterprise Resource Planning (System)
ESB	Enterprise Service Bus
FR	(TradeXpress) File Router
FRR	(TradeXpress) File Router Routing Rules
ICT	Information and Communications Technology
IOI	Information-Oriented Integration
IR	(TradeXpress) Intermediary Router
IRR	(TradeXpress) Intermediary Router Routing Rule
IRS	Intelligent Routing Solution
ISP	Integration Service Provider
MAS	Multi-Agent System
OVT	Organisaatioiden Välinen Tiedonsiirto
OWL	Web Ontology Language
PIP	(RosettaNet) Partner Interface Process
PMS	Performance Measurement System
POI	Portal-Oriented Integration
RDF	Resource Description Framework
RNIF	RosettaNet Implementation Framework

RP	(TradeXpress) RTE Routing Program
RPC	Remote Procedure Call
RTE	(TradeXpress) Reliable Transaction Engine
SME	Small and Medium-Sized Enterprise
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SOI	Service-Oriented Integration
SR	(TradeXpress) Syntax Router
SRR	(TradeXpress) Syntax Router Routing Rule
SSL	Secure Socket Layer
SQL	Structured Query Language
SCM	Supply Chain Management (System)
TX	TradeXpress
UBL	Universal Business Language
UDDI	Universal Description, Discovery and Integration
UI	User Interface
URI	Uniform Resource Identifier
VAN	Value Added Network
VBE	Virtual Enterprise Breeding Environments
VE	Virtual Enterprise
VO	Virtual Organization
WSDL	Web Services Description Language
XI	(SAP) Exchange Infrastructure

1. Introduction

1.1 Background

Enterprises are currently encountering difficulties in aligning with the highly dynamic marketplace. Progressive globalization and rapidly evolving technology have made the operational environment for companies faster-paced as well as more unpredictable than ever before (Grönroos 2003, p.1). Simultaneously, advances in information and communications technology (ICT) have opened new business opportunities and prepared the way for companies to establish new forms of inter-company collaboration. One of the most intriguing new collaborative organizational forms is virtual enterprise (VE). The emergence of VEs has been triggered by both the external pressures from the dynamic marketplace (Weisenfeld et al. 2001, p.326) and by modern ICT, which makes sophisticated inter-company cooperation possible (Grönroos 2003, p.2).

In a VE, a number of independent companies combine their specialties to form a reconfigurable inter-company network using modern ICT, in order to exploit a limited business opportunity. The most recent form of VE is characterized by inexpensive and opportunistic formation, as well as highly dynamic configuration. Dynamic configuration means changing partners, partner roles and business processes after VE formation and eventual dissolution when business objectives are met. (Goranson 2003, p.113) The property of dynamic configuration also promises VEs the flexibility required to cope with the turbulent market. However, heterogeneity of partnering companies poses a significant threat to the promised flexibility. Basically, a VE must be able to deal with considerable variance in organizational processes, information systems, data formats and business semantics of the partnering companies, both during formation and operation. In order to achieve sufficient level of interoperability between VE partners, the heterogeneity issues need to be overcome. Hence, an important VE success factor is integration, spanning both organizational and technological levels.

One part of the integration domain is business-to-business integration (B2Bi), which essentially aims at integrating trading partners' business processes, applications and systems. One of the main obstacles of achieving integration between any two partners is the heterogeneity of their systems, applications and business processes. Moreover, many of the B2Bi solutions have been originally designed to best suit the needs of stable partnerships, which VEs evidently are not. Hence, VEs are likely to cause extra concern for B2Bi, because of the aforementioned heterogeneity of VE partners and the dynamic configuration of VEs. Viewed from another perspective, B2Bi may represent an obstacle for VEs, which inhibits them from reaching their full potential. Researching this intertwined relationship between VEs and B2Bi is the main objective of this thesis.

1.2 Objectives of the thesis

In this thesis, the emphasis is on analyzing VE formation and operation as B2Bi problems. The problems are approached by answering three specific research questions. Firstly, what kind of requirements do VEs set to B2Bi? VEs are likely to differ considerably from traditional partnerships and collaborative networks in numerous aspects. Thus, they are also likely to have a unique set of requirements for underlying support infrastructures and inter-enterprise integration in general. To answer the first question, the distinctive characteristics of VEs, their life cycle and requirements they set on ICT infrastructures need to be uncovered.

Secondly, what are the main approaches to B2Bi and the central components of B2Bi, and how they support VE integration? B2Bi is not a new application area, and many different approaches, as well as technologies, have been created to address the varying needs of different enterprises and trading communities that seek inter-enterprise integration. As a response to the second research question, the main B2Bi approaches and technologies are presented. Consideration is put on how these conform to the requirements set by VEs. Currently, B2Bi landscape is evolving and new integration solutions are being developed to address the changing needs of companies. The main trends and emerging components of B2Bi are also discussed in light of VE requirements.

Thirdly, how a company's preparedness to join a VE is potentially affected in case it decides to use the services of an integration service provider (ISP), which runs an integration server, as its B2Bi strategy? This strategy to B2Bi is likely to entail advantages but inevitably also some disadvantages. These aspects are discussed in VE context to address the last research question.

1.3 Structure of the thesis

This thesis is divided in five chapters (see Figure 1). After the introduction, in chapter 2, VE is introduced as a distinctive organizational form. This includes presenting the evolution of the VE concept, the main incentives for VE formation, the distinguishable characteristics of VEs, the central stages of VE life cycle and finally, the technological requirements that VEs set on the support infrastructures. As the objective of this thesis is to evaluate B2Bi in VE context, the emphasis of chapter 2 is on finding the requirements that VEs set on B2Bi.

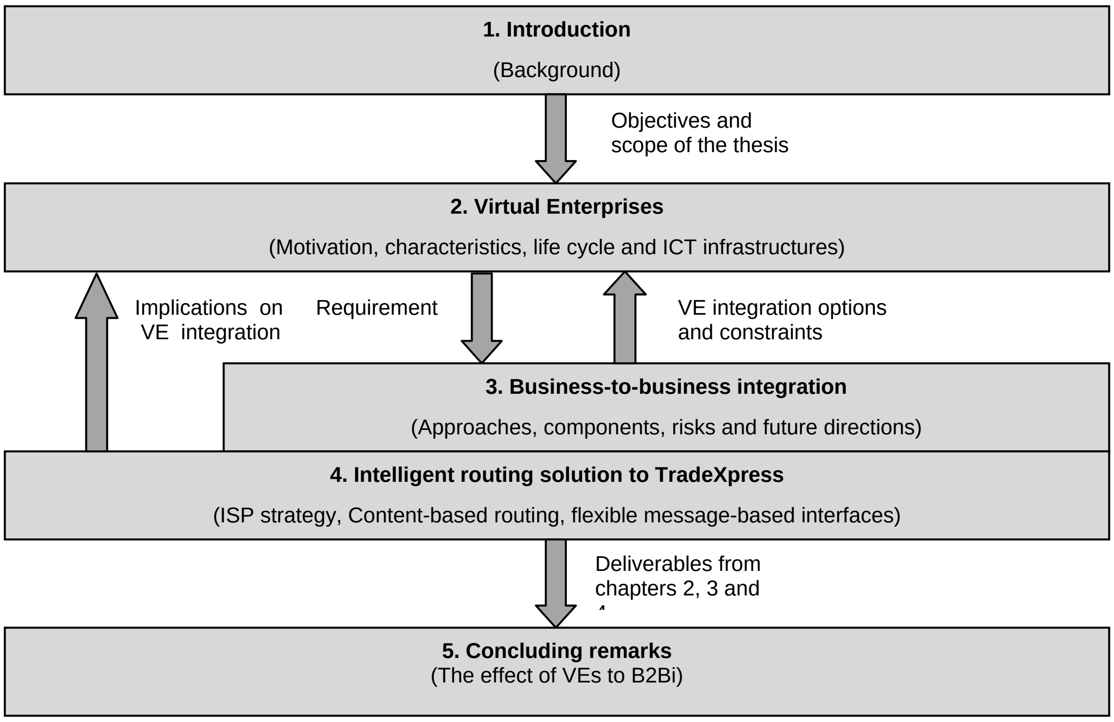

1. Introduction
(Background)
Objectives and scope of the thesis
2. Virtual Enterprises
(Motivation, characteristics, life cycle and ICT infrastructures)
Implications on VE integration
Requirement
VE integration options and constraints
3. Business-to-business integration
(Approaches, components, risks and future directions)
4. Intelligent routing solution to TradeXpress
(ISP strategy, Content-based routing, flexible message-based interfaces)
Deliverables from chapters 2, 3 and 4
5. Concluding remarks
(The effect of VEs to B2Bi)

Figure 1. Structure of the thesis.

The aim of chapter 3 is to map the problem domain of B2Bi and synthesize the findings to the ones made in the previous chapter. The objective is to go about B2Bi by identifying the main approaches that exist for B2Bi, the central components of B2Bi used to achieve satisfactory integration solutions between business partners and the biggest pitfalls that exist in B2Bi projects. Additionally, chapter 3 entails an attempt to recognize the major trends and emerging components of B2Bi and consideration, how these are relevant for VEs. The bottom line of this chapter is to analyze whether the current, as well as prospective, B2Bi solutions conform to the requirements set by VEs.

In chapter 4, the B2Bi problem domain is viewed from the perspective of ISPs that offer a wide range of integration services to a variety of customers. Chapter 4 entails the experimental part of this thesis, in which an intelligent routing solution (IRS) is designed to a specific integration server, TradeXpress (TX). This solution will address the need of an ISP to effectively and flexibly configure the message-based connections between its customers and their business partners on TradeXpress. Additionally, the implications of IRS and B2Bi outsourcing strategy to VE integration are discussed in chapter 4. Lastly, conclusions of this thesis are drawn in chapter 5.

2. VIRTUAL ENTERPRISES

Virtual enterprise is a relatively new organizational concept, but its roots are lying deep in the theory of network organizations. The concept could be perceived as “the further optimization and perfection of the basic ideas about dynamic networking” (Putnik et. al 2005, p.3). However, virtual enterprise most evidently is an evolving concept, as advances in logistics, as well as ICT, have made it possible for companies to produce more advanced forms of VEs. The development has led to a new organizational paradigm that is expected to drive enterprises towards closer alignment with the dynamic market (Cunha and Putnik 2006, p.vii).

In this chapter, the focus is on capturing the essence of VEs, that is, the reason why they are formed, what they are like, how they operate, as well as evolve, and finally, the requirements they set on ICT infrastructures. The findings of this chapter will serve as a basis for researching the effect of VEs to B2Bi.

2.1 FROM DYNAMIC NETWORKS TO AGILE VIRTUAL ENTERPRISES

“These are turbulent times in the world of organizations”, Raymond E. Miles and Charles C. Snow wrote in their article, published in California Management Review in 1986. Rapid technological change and shifting patterns of international trade and competition were identified as important causes for the turbulence. (Miles & Snow 1986, p.62) More than twenty years later, it seems that the same factors, rapidly evolving technology and intensified global competition, are forcing companies to constantly rethink their competitive approaches.

Miles and Snow (1986, p.62) found out that a unique mixture of strategy, structure and management processes appeared to produce a considerable result - a new organizational form to be called the dynamic network. The main characteristics of a dynamic network were, quite interestingly, vertical disaggregation, brokers, market mechanisms and full-disclosure information systems. (Miles & Snow 1986, pp.62-65) Interestingly, because the theory that involves dynamic networks seems to have an effect also on the theory of VEs, a concept that was introduced few years later.

In the beginning of 1990's a new organizational network approach referred to as virtual organization (VO), begun to gain ground. This was much due to the cover story of Business Week in February 1993, written by John A. Byrne and also the book the article was based upon, The Virtual Corporation, written in 1992 by William H. Davidow and Michael S. Malone (Franke 2002, p.2). Byrne defined the virtual corporation as a network of independent companies – suppliers, customers, sometimes even rivals - linked by information technology to exploit a specific business opportunity. The carrying idea is that companies bring in their core competences to create a flexible "best-of-everything" organization. Moreover, Byrne stated that information technology will provide the necessary link between the companies and it must at minimum enable communication between current and potential partners, until networks and standards make more sophisticated forms of inter-company communication possible. (Byrne 1993, pp.37-40)

Since the introduction of the VO concept, many variations of it have emerged. One of the most central is the virtual industrial enterprise, introduced by Martin Hardwick et al. (1996). The virtual industrial enterprise manifests that independent companies unite to combine their specialties in order to exploit a world-wide product manufacturing opportunity, and go their separate ways as the opportunity is met (Hardwick & Bolton 1997, p.59). Additionally, Hardwick et al. (1996) pointed out the importance of standards as an enabler of communicating industrial information between partners of a virtual industrial enterprise. Standards, in turn, serve as an important building block for B2Bi, as chapter 3 reveals.

The concept of virtual industrial enterprises can be used to narrow down the application area of VOs, concentrating on the manufacturing industry. Moreover, the term VO can refer to, for instance, virtual municipality organizations, associating via a computer network, whereas VE always refers to an alliance of enterprises (Camarinha-Matos & Afsarmanesh 2001, p.337). To keep the focus on inter-enterprise relations and to avoid confusion, the term VO is not used later on in this thesis. An overview of the central concepts that involve inter-company collaboration is given in Figure 2.

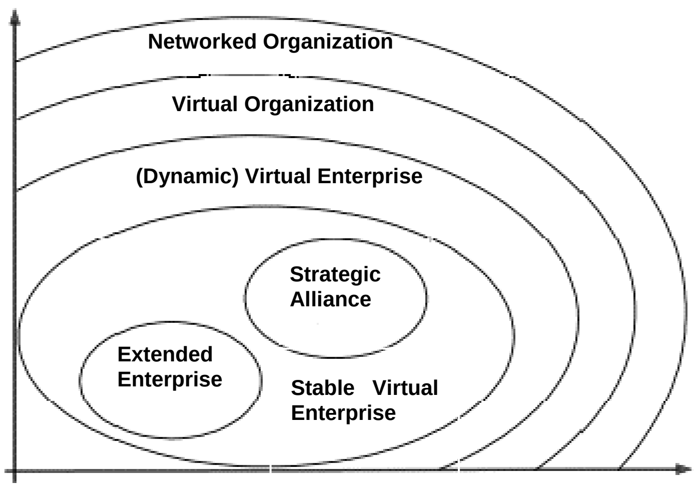


Figure 2. Central concepts of inter-enterprise collaboration (Rocha et al. 2005, p.231).

Recent development has led to the emergence of agile virtual enterprises (AVEs), one of the most advanced organizational paradigms of today's (Cunha & Putnik 2006, p.viii). A VE is conceived agile only if it is created with the intent of dissolving or reconfiguring at the later stages of its existence (Goranson 1999 cited in Putnik et al. 2005, p.6). An agile, or advanced, VE is highly dynamic by its configuration, which allows switching partners or partner roles after the partnership network is established (Goranson 2003, p.113). This dynamicity can generally be seen as one of the most important drivers towards VE setups (Weisenfeld et al. 2001, p.326).

To date, a commonly agreed definition of VE is still missing (Putnik et al. 2005, p.3; Camarinha-Matos & Afsarmanesh 2001, p. 336). Hence, the following definition was developed to be used throughout this thesis:

> *A virtual enterprise is a dynamic, as well as temporary, network of independent companies that come together to exploit a limited business opportunity and base their interaction on modern ICT.*

2.2 INCENTIVES FOR VIRTUAL ENTERPRISE FORMATION

In previous section 2.1, VEs were identified to be formed in response to specific business opportunities. These opportunities may include, for example, one-time manufacturing of specific landing gear in case of airplane damage, manufacturing a large batch of products for a promotional activity or serving the needs of an emerging market niche (Aerts et al. 2002, p.311). In this thesis, the incentives for creating or joining VEs are grouped into two categories – internal and external. Internal incentives correspond to building up the strengths of a company and alleviating the weaknesses. External incentives, on the other hand, refer to leveraging the opportunities and avoiding the threats that exist in the market.

Actually, already Byrne (1993) did bring up many of the factors that influence a firm’s decision to form or join a VE. He stated that companies need to ally to take advantage of specific business opportunities with limited existence. Opportunities, such the companies could not exploit by themselves, due to internal factors, including the lack of resources or skills, excess risk or limited geographical reach. (Byrne 1993, pp.36-37). In other words, resource complementarity, risk-sharing and gaining access to partners’ markets is among key motivations for forming VEs.

The internal incentives presented above hold particularly true for small and medium-sized enterprises (SMEs). SMEs try to cope with reduced size and geographical reach and maintain a high degree of specialization in order to stay competitive. Unlike merging to a larger company, VE setups allow SMEs to better retain their business autonomy (Gou et al. 2003, p.333). On the other hand, larger companies may isolate parts of their businesses to gain more flexibility and better efficiency by creating or joining a VE (Cardoso & Oliveira 2005, p.15). Hence, a company might be partly conventional and partly virtual (Mowshowitz 2002, p.125). Moreover, there are no obvious reasons why a VE could not be both formed by and consist of SMEs, as well as larger companies. The disabling factors for VE formation are likely to result from other issues than just mere company size.

An important external incentive for VE formation is building a consortium that is able to bring the partnering companies into close alignment with the changing market, in an agile and dynamic manner (Cunha & Putnik 2006, p.vii). Agility, in turn, is needed to endure the unpredictable changes in the marketplace by creating fast responses in a cost-efficient way (Camarinha-Matos & Afsarmanesh 2003, p.139). Agility is also considered as a source of competitive advantage in multiple domains, including industrial manufacturing (Wu & Su 2005, p.119). Besides rapidly changing markets, the fast pace of technological evolution is a cause for concern for many companies, as noted in section 2.1. This rapid technological evolution has, among other things, a noticeable decreasing impact on product life-cycles. And with the shrinking life-cycles, companies need to respond with more products and product variants to market, faster than ever before (Cunha & Putnik 2006, p.9). Additionally, companies need to react quickly to changing customer requirements and fluctuations in product demand. These are situations where VEs are expected to deliver considerable results.

A VE's effect on innovation has also been considered. Technological innovation, which can be very complex, may often require collaboration (Weisenfeld et al. 2001, p.324). Being part of a network organization can make companies more susceptible to confrontation and exchange of new ideas, which is a basis for innovation (Camarinha-Matos & Afsarmanesh 2003, p.139). However, being virtual and thereby having weak ties to partners, is not necessarily always virtuous when it comes down to innovation. Chesbrough and Teece (2002, p.132) have argued that companies must assess two parameters, that is, capabilities needed to produce innovation and the type of innovation, before organizing a business for innovation (see Figure 3). This includes evaluating whether the pursued innovation is autonomous or systemic, that is, whether it can be pursued independently or it requires complementary innovation. Additionally, companies must determine if the capabilities required producing innovation need to be created in-house or accessed through partnerships. (Chesbrough & Teece 2002, p.132)

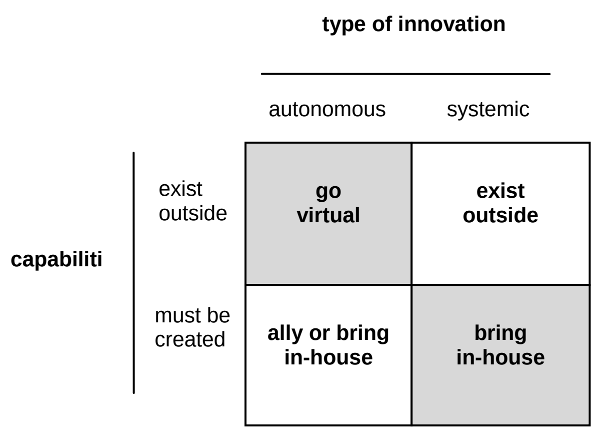


Figure 3. Autonomous and systemic innovation (Chesbrough & Teece 2002, p.132)

What kind of industries do benefit from the VE paradigm, then? There are several examples of VEs from different industries. The application area of VEs can range from travel agencies using dynamic packaging applications for automated travel product configuration (Salam & Stevens 2007), to clothing industry leveraging e-business technology for improved competitiveness (Tatsiopoulos et al. 2002) and industrial manufacturing (Camarinha-Matos & Afsarmanesh 2001) aiming for improved supply chain efficiency. In clothing industry, there has been a strong tendency to keep only the core competence in-house and outsource rest. Tatsiopoulos et al. (2002) have researched a real-life VE case, where clothing industry companies sought improved lead-times with ICT-enabled collaboration. While the companies were able to create value through reduced lead-times and costs, the case also highlighted some of the realities of a VE. For instance, many of the potential subcontractors were not ready on either technological or organizational level to join the VE. (Tatsiopoulos et al. 2002) Hence, even if the incentives for joining or creating a VE are strong, the inadequate preparedness of companies dictates the feasibility of a VE.

2.3 Characteristics of a virtual enterprise

VEs were identified in previous sections to be formed in response to market opportunities, to keep company's focus on core competences and to alleviate internal weaknesses and external threats. But what are the main characteristics of VEs? And which features differentiate VEs from other collaborative arrangements, like extended enterprises, strategic alliances and strategic networks?

According to O'Connell and Nixon (2000, p.453), for a consortium to be referred as a VE, it must base its co-operation or sheer existence on the use of ICT. Indeed, modern ICT seems to play an important role in a VE as an enabler of, for example, application integration, information management and daily operations. The extensive use of ICT, however, does not sufficiently separate VEs from more traditional networked organizations, as these generally use computer networks as inter-links between organizations (Camarinha-Matos & Afsarmanesh 2001, p.337).

A distinctive characteristic of a VE can be derived from the term virtual enterprise. Virtuality in an organizational context is seen as an unconventional social configuration whose structure and functions rely on ICT. Additionally, these configurations are not as constrained by location and time as traditional ones. This property enables activities to be geographically spread and conducted both asynchronously and synchronously. Virtuality also facilitates creating corporate structures with amorphous internal and external boundaries. (Mowshowitz 2002, p.25) In other words, virtuality can be seen as a property realized through ICT, which helps enterprises alleviate spatial, as well as temporal, constraints and create flexible company interfaces.

VEs are often established to take advantage of short-lived business prospects and characterized by their opportunistic behavior (Byrne 1993, p.36; Aerts et al. 2002, p.312). However, other collaborations may also behave opportunistically when they, for example, try to reach their strategic objectives. The fact that a VE's configuration may be changed frequently (Goranson 2003, p.113; Putnik et al. 2005, p.4) in order to capitalize on these opportunities helps better in distinguishing VEs from other network organizations, for example, strategic networks. Strategic networks consist of inter-organizational ties that are of strategic importance and enduring, include strategic alliances, joint ventures, and other long-term relationships (Gulati et al. 2000, p.203). Dynamicity in VE setups and possible dismantling after the objectives are met will most evidently lead into shorter-term inter-company linkages than in strategic networks.

Typically, a VE is characterized by its ability to be dynamically reconfigured to meet, for example, changed resource (Wu & Su 2005, p.120), customer or market requirements (Cunha & Putnik 2006, p.120). The reconfiguration of a VE may include changing participant roles, replacing existing partners or adding completely new ones (Camarinha-Matos & Afsarmanesh 2001, p.337). The dynamicity basically means that the changes to the VE configuration can be made while the VE is in operation. This reconfiguration characteristic can be referred to as reconfiguration dynamics (Putnik et al. 2005) or more simply, switching (Mowshowitz 2002). For example, in manufacturing it may be necessary to manufacture 1 to 1000 products simultaneously or adapt lot sizes from 1 to 1 000 000 (Putnik et al, 2005, p.23). Additionally, reconfiguration may occur when a product has to be changed and new competencies need to be acquired, or when a partner needs to be removed due to poor performance (Cunha &Putnik 2006, p.127). According to Chalmeta and Grangel (2003, p.146), switching applies to the partners that do not have high strategic relevance in the network.

With switching, a VE is able take advantage of lower prices for required goods and services. Switching also promotes efficient allocation of resources, as well as organizational reflection, as resource allocation requires clearly defined objectives. (Mowshowitz 2002, pp.44-54) In VE terminology, when the reconfiguration occurs as a response to changed conditions, the property is called flexibility or adaptivity. When the reconfiguration is done also proactively, the property is called agility. (Putnik et al. 2005, p.4) However, reconfiguration dynamics does not guarantee that a VE can automatically predict changes in the market or even detect them. This is likely to require a certain level of finesse from the underlying organizations.

While VE reconfiguration dynamics may sound appealing, it is important to note that it introduces a new set of costs - called transactions costs (Putnik et al. 2005, p.5). Every time switching occurs in a VE, a number of administrative and logistics changes are needed. And changes, in turn, consume both time and resources. Hence, if the configuration of a VE is changed too often, the transaction costs may exceed the potential benefits of reconfiguration dynamics. (Mowshowitz 2002, p.44-54) Besides transaction costs, reconfiguration dynamics may be hindered by companies' tendency to protect their core competences from other partners. Therefore, VEs are forced to develop mechanisms to minimize these reconfiguration disablers. (Putnik et al. 2005, p.5)

As a summary of sections 2.1-2.3, opportunistic, as well as inexpensive, formation, limited existence, building upon core competences and extensive use of ICT, reconfiguration dynamics and flexible company interfaces are central characteristics of VEs. Throughout this thesis, these characteristics are used as a basis for analysis, for example, when considering VE life cycle or the effect of VEs on B2Bi.

2.4 Virtual enterprise life cycle

In VE setups, companies need to band together quickly in order to better respond to business opportunities. Some VEs are established towards a single opportunity and once the opportunity is met, the VE will most likely dismantle (Byrne 1993, p.36; Camarinha-Matos & Afsarmanesh 2001, p.339). It is also possible that VEs endure for an indefinite number of business processes or for a predetermined time-span (Camarinha-Matos & Afsarmanesh 2001, p.339). The common denominator for both of these options is that a VE is likely to evolve during its life cycle.

The phases of the life cycle can be referred to as creation and configuration, operation and finally, dissolution (Camarinha-Matos & Afsarmanesh 2001; Wu & Su 2005; Nayak et al. 2001). The VE life cycle model used in this thesis is visualized in Figure 4. Other definitions of the life cycle phases exist in (Cunha & Putnik 2006; Cardoso & Oliveira 2005; Kanet et al. 1999; Chalmeta & Grangel 2003), but the activities involved are widely overlapping. The main differences between the definitions are in how the pre-operation tasks are categorized and in how much emphasis is given to VE evolution. In this section, the steps that are performed during the three different phases of the VE life cycle are presented.

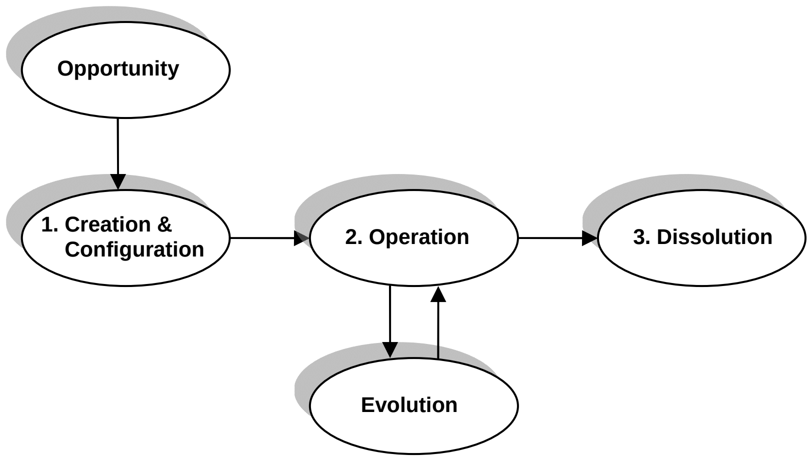


Figure 4. Phases of VE life cycle (Camarinha-Matos & Afsarmanesh 2001, p.337, modified).

2.4.1 Creation and configuration

The life cycle of a VE is triggered by the identification of a business opportunity (Cunha & Putnik 2006, p.126; Camarinha-Matos & Afsarmanesh 2007, p.129). Once the opportunity is recognized by the VE initiator or initiators, some coarse-grained planning is needed in order to track the required competences and resources for the given opportunity (Camarinha-Matos & Afsarmanesh 2007, p.129). Naturally, without a real and realizable business opportunity, the formation of a VE is gratuitous. Other activities of the pre-operation stage involve search for partners and partner selection, contract negotiations, setting the VE objectives, strategy and governance principles, designing the cross-organizational business processes and developing the VE ICT infrastructure (Chalmeta & Grangel 2003, p.145; Camarinha-Matos & Afsarmanesh 2007, pp.129-131). The VE strategy and cross-organizational processes may induce adjustments on companies' internal strategy and processes. For example, new internal processes may need to be created or improvements made to the existing ones, because in a VE, work may be conducted in parallel or resources need to be relocated. Additionally, individual companies' ICT systems may need to be extended to support the VE "process map". (Chalmeta, & Grangel 2003, pp.145-146)

Companies tend to spend much time and resources on establishing traditional partnerships. The VE creation phase may prove to be very costly, too. To reduce the time and effort invested in VE creation, some of the most ambitious VE research projects have sought to automate parts of the process with help of modern technology, namely multi-agent systems (MAS) (Cardoso & Oliveira 2005, pp.14-15). Among other things, missing standardized information about potential partners, lack of common collaboration infrastructure and insufficient level of readiness of companies to join VE setups inhibit this automation effort. Additionally, overcoming the mismatches that result from heterogeneity in, for instance, ICT infrastructures, corporate cultures, working methods and business practices require considerable effort during the first phase of the VE life cycle. (Camarinha-Matos & Afsarmanesh 2007, p.119)

Particularly, when the window of opportunity is short, the issues listed above suggest that there may need to be some form of pre-formation collaborative work conducted between a set of potential VE participants. This pre-formation collaborative work should address preparedness of companies to join a VE. Moreover, it should promote fast as well as inexpensive formation and reconfiguration of a dynamic consortium. According to Camarinha-Matos and Afsarmanesh (2007, p.120), longer-term VE setups and VEs that are formed from niche sector companies, already using the same tools and business practices are likely to be less affected. The preparedness of companies and the pre-formation work are discussed more closely in sections 2.4.1.1 and 2.5, respectively.

2.4.1.1 Partners search and selection

The identification of a business opportunity and initial partner selection are important prerequisite activities for VE formation. Selecting the right mixture of companies that possess the needed complementary capabilities and resources and establishing trust-based partnerships in a potentially very short period of time is essential to the success of a VE (Franke 2002, p.ix). Additionally, there is a need to develop procedures for searching and selecting partners that support both formation and reconfiguration dynamics. As reconfiguration dynamics characteristic of VEs suggests, partners may need to be added and removed from the consortium during the operation phase.

According to Camarinha-Matos and Afsarmanesh (2007, p.119), partners search and selection is more than a matching process based on competences and capabilities. They state that different elements, including previous collaboration experiences and factors of subjective nature influence the choice of partners. Additionally, the companies’ preparedness to join a VE, both technological and organizational, should be considered. Katzy and Löh (2003) also acknowledge the importance of pre-existing relationships for the success of a VE. These factors support the suggestion that the VE creation process, including partners search, cannot be fully automated. Instead, a computer-assisted framework could be used to help human planner in the process (Camarinha-Matos & Afsarmanesh 2007, p.120).

Besides complementary capabilities and subjective factors, many other elements influence the partners’ selection. These include cost, quality, delivery time and reliability. Although one of the goals of a VE may be to alleviate spatial constraints, the costs and time involved in transporting materials from one physical location to another cannot be avoided. (Wu & Su 2005, p.120) Additionally, the similarity of potential partners’ ICT infrastructures needs to be taken into account. Once the relevant factors are known, multi-criteria models can be used to run analysis on a set of potential partners. (Camarinha-Matos & Afsarmanesh 2007, p.124) These models can serve as a basis for optimization efforts and the models are relevant also for minimizing the costs related to partners’ selection both in the formation phase and when the VE is reconfigured.

How does the search for partners occur, then? Are the partners selected from an open universe of corporations? And can technology be used to assist the process? Byrne (1993, p.38) interviewed Roger N. Nagel, who predicted that an information infrastructure, referred to as communications superhighway, would provide means for remote units to rapidly locate suppliers, designers and manufacturers through an information clearinghouse. Currently, information with normalized and updated profiles about potential partners is not widely available through any information clearinghouse (Camarinha-Matos & Afsarmanesh 2007, p.119). This, in turn, makes computer-assisted partner search a tough task. Additional implication is that VE partners cannot be selected from an unlimited set of companies, because the time consumed in searching suitable companies and integrating them to the VE would be too high to give VEs the potential to be truly flexible.

To address the problems related to fast, as well as inexpensive, formation and dynamic reconfiguration of a VE, specific organizational frameworks have been created. Also referred to as external entities, these frameworks are long-term organizations that nurture the creation and operation of VEs. It is argued that without external entities, the rapid formation and dynamic reconfiguration of VEs is next to impossible. (Putnik et al. 2005, p.8) Virtual webs (Goldman et. al 1995, cited in Franke 2002, p.3), market of resources (Cunha et al. 2000, cited in Putnik et al. 2005, p.8) and virtual enterprise breeding environments (VBE) (Camarinha-Matos, Afsarmanesh 2003) are examples of external entities.

A VBE can be formed, for instance, from companies belonging to the same industrial district or cluster. The companies of a VBE may also be geographically dispersed. (Vallejos et al. 2007, p.588) Also other types of organizations, for example, research institutes may be included (Camarinha-Matos & Afsarmanesh 2007, p.125). The carrying idea is that a group of corporations that have the potential and will to cooperate make preliminary cooperation arrangements and declare their core competences, which are “registered” in a directory. The cooperation arrangements include agreeing on business rules, developing basis for common interoperable infrastructures, as well as ontologies, and building trust, which is necessary element for the success of any given VE. (Camarinha-Matos & Afsarmanesh 2003, p.156) This way, the companies’ readiness for VE setups is likely to be improved and the cost associated to partners search, as well as VE configuration, reduced (Camarinha-Matos & Afsarmanesh 2007, p.125). A real-life VBE example can be found from Brazilian mould and die sector (Vallejos et al. 2007).

When the conditions for cooperation are set, dynamic VEs can be derived from the VBE. Multiple VEs can exist within a given VBE and companies may take part in several VEs simultaneously (see Figure 6) (Vallejos et al. 2007, p.593). A VBE has an open, but controlled border, so that companies can freely join the given VBE, but need to comply with the principles of the VBE.

VEs may be formed when the initiator of the VE, called broker, detects the opportunity and selects the most suitable partner candidates from the directory or declares the opportunity to the members of the VBE and waits for the emergence of the potential candidate consortia. (Camarinha-Matos & Afsarmanesh 2007, pp.121-126) If the required competences are not found within the VBE, external partners can be brought into the consortium. The requirement is that this company commits to the operating principles of the VBE. Naturally, it is more costly to bring companies outside the VBE to the VE, because more time is spent in searching and the companies are less prepared than the ones inside the VBE. (Camarinha-Matos & Afsarmanesh 2003, p.156)

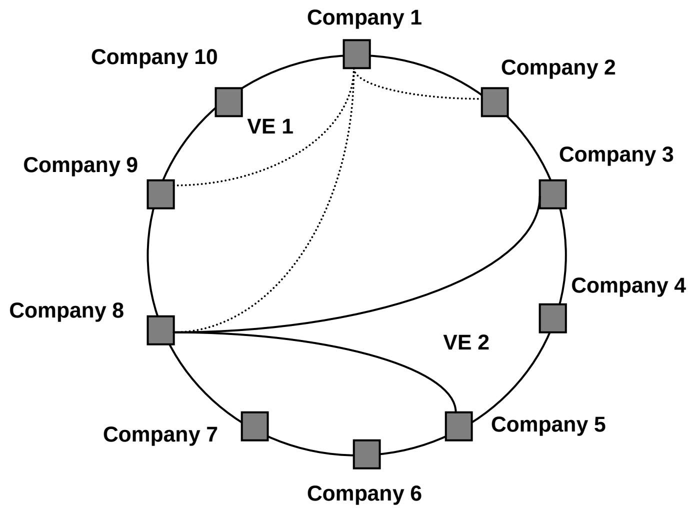


Figure 5. A VBE with two VEs in operation (Vallejos et al. 2007, p.594, modified).

All in all, a VBE can be a two-edged sword. If partners are selected from a VBE, an advanced organization with a set of competences can be generated quickly as a response to a short-framed opportunity. On the other hand, the range of companies within a VBE is likely to be limited when compared to the open universe of organizations, and therefore the VEs may be formed from trade-off companies, not best possible. Another possible downside is that the cooperation efforts that are required a priori to the VE formation consume both time and resources, and there are no promises that joining a VBE guarantees any new business opportunities. For instance, structuring the VBE for Brazilian mould and die sector was a result of a project that lasted more than two years (Vallejos et al. 2007, p. 593). Additionally, a VBE administrator needs to be assigned for recruiting new VBE members and managing the cooperation inside a VBE (Camarinha-Matos & Afsarmanesh 2007, p.126). This is likely to induce running costs in addition to the costs that result from the initial VBE cooperation efforts. The emergence of standardized, as well as computer-processable, information about companies and a dominant technology that would allow companies to be simply "plugged" into the VE could diminish the relevance of these support organizations and give an additional boost to VE formation.

2.4.1.2 Configuration: Design and integration

Once the initial set of partners for a given opportunity have been selected and agreements reached, the subsequent activities include setting the VE objectives and strategy, defining the cross-organizational business processes and configuring the ICT infrastructure, performance measurement system (PMS) and governance mechanisms (Chalmeta & Grangel 2003, pp.145-146; Camarinha-Matos & Afsarmanesh 2007, p.131). In other words, the initial participant companies need to be integrated as one temporary consortium. Since a VE presumably evolves during operation, the business processes, ICT infrastructure and management mechanisms need to be tailored to support change. In the foreseeable future, VE contracts as well as cooperation agreements may be based on electronic contracting (Camarinha-Matos & Afsarmanesh 2007, p.125), which is expected to support automation and to be more efficient than standard paper-based contracting (Cardoso, Oliveira 2005). In management domain, selection of significant indicators of performance is also necessary (Camarinha-Matos, Afsarmanesh 2007) for tracking the performance of individual VE partners and VE as a whole. A PMS can be created to support VE management and to reflect recent performance to the VE strategy (Chalmeta & Grangel 2003, pp.145-146).

As noted in section 2.4.1, the VE initiator may already have sketched a rough plan, which needs to be revised, for instance, for more detailed partner roles, business processes definitions and thereby VE structure. The structure of a VE shows the relationships and the business processes carried out between partnering companies. Katzy and Löh (2003, p.3) identify three possible topologies of VEs, displayed in Figure 6. By examining the topologies, it becomes quite apparent that some VE network nodes are in more strategic positions than others, and therefore less likely replaced. For example, in hub and spoke topology, the company acting as a "hub" is in this kind of strategic position (Katzy & Löh 2003, p.3). Additionally, the topologies may give implications on how the power is distributed among the VE participants. If a dominant company exists, it may enforce its own conventions, including business process models and information exchange mechanisms. (Camarinha-Matos & Afsarmanesh 2001, p.339).

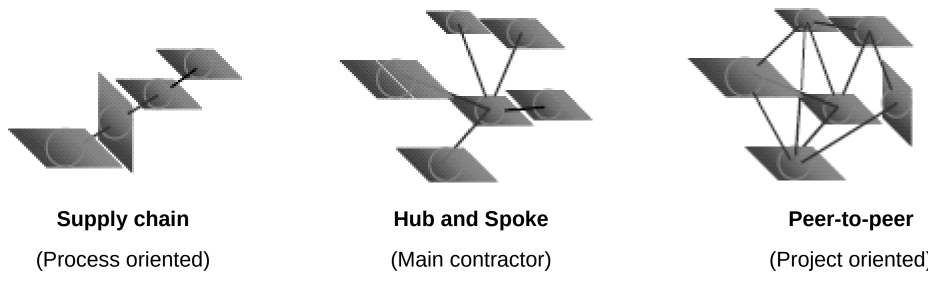


Figure 6. Topologies for virtual enterprises (Katzy & Löh 2003, p.3).

As noted above, the design of cross-organizational business processes partly determines the structure of a VE network. Additionally, the processes specify the tasks required to achieve VE business objectives (Camarinha-Matos & Afsarmanesh 2001, p.350). A sample point-to-point cross-organizational business process, where enterprise A interacts with partner enterprise B for a product part supply, is presented in Figure 7. Other examples of collaborative

VE processes include collaborative design, collaborative order fulfilment, collaborative planning, et cetera (Nayak et al. 2001, p.84).

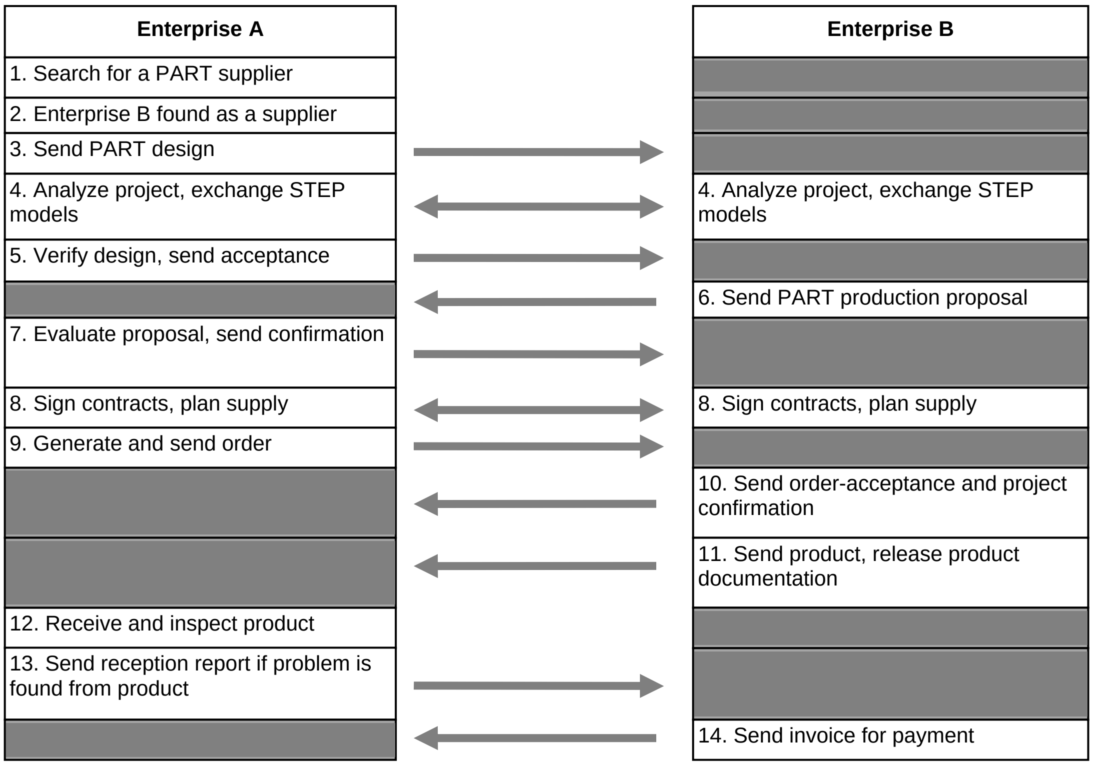

Enterprise A
Enterprise B
1. Search for a PART supplier
2. Enterprise B found as a supplier
3. Send PART design
4. Analyze project, exchange STEP models
4. Analyze project, exchange STEP models
5. Verify design, send acceptance
6. Send PART production proposal
7. Evaluate proposal, send confirmation
8. Sign contracts, plan supply
8. Sign contracts, plan supply
9. Generate and send order
10. Send order-acceptance and project confirmation
11. Send product, release product documentation
12. Receive and inspect product
13. Send reception report if problem is found from product
14. Send invoice for payment

Figure 7. A sample cross-organizational business process (Camarinha-Matos & Afsarmanesh 2003, pp.142-143, modified)

In a VE, the business processes are typically distributed. They comprise of a set of dynamic, as well as temporary, business processes conducted by different VE members. (Pereira Klen et al. 2001, p.186) Normally, the company that initiated the VE is in charge of the distributed business process coordination (Rabelo et al. 1996, cited in Pereira Klen et al. 2001, p.186). Another view is that a VE coordinator is assigned for this particular task (Camarinha-Matos & Afsarmanesh 2001, p.351). According to Mowshowitz (2002, p.43), the trick in VEs is "the categorical separation of abstract requirements from concrete satisfiers, which supports switching as a systematic management procedure." This may include, for example, separating components of a complex product from the potential suppliers or dividing complex production tasks into smaller, manageable subtasks, so that they can be executed by different partners. This way, it is also possible to reassign an abstract requirement to different concrete satisfiers and thereby support dynamic reconfiguration. (Mowshowitz 2002, pp.43-44)

The defined business processes also have an influence on VE ICT infrastructure requirements. At bare minimum, the ICT infrastructure needs to enable safe as well as coordinated communications and information exchange between the VE partners. The communication may include business and technical transactions, as well as generic information. (Camarinha-Matos & Afsarmanesh 2003, p.142) Thus, the VE ICT infrastructure is playing the role of interoperation and integration enabler among the VE participants (Hardwick & Bolton 1997, pp.59-60; Camarinha-Matos, Afsarmanesh 2003, p.87; Vallejos et al. 2007, p.588). Figure 7 shows an example of information exchange between two VE partners. It also illustrates the need to integrate the business processes between companies. The interaction, and thus the requirements for the infrastructure are likely to become more complex, as more parties are involved in one process. But what may mitigate the complexity is that in a real VE situation the partners need to integrate only in a specific context, consisting of processes that will represent only a few of all the possible conditions. (Goranson 2005, p.284) Hence, there is no need for complete integration between the partners. The issues related to ICT infrastructure configuration are considered in more detail in section 2.5.

2.4.2 OPERATION

In operation phase, a VE conducts the business processes that were designed in the creation stage in order to achieve its objectives (Camarinha-Matos & Afsarmanesh 2001, p.337). Although the basis for VE agility is created during the formation phase, according to Gou et al. (2003, p.333), the decisive phase for the success of a VE is the operation phase. During operation, the primary activities involve managing operations and change (Chalmeta & Grangel 2003, p.145). The performance of a VE and its individual partners needs to be constantly monitored, for example, with the help of a PMS. If the measured performance does not satisfy the requirements set in the formation phase, the network may need to be reconfigured. (Carvalho et al. 2005, p.362) Due to transaction costs, the management needs to weigh both the benefits and costs of reconfiguration. For instance, if the performance of a strategic partner is low, but he costs involved in integrating a substitutive partner to the network are high, other ways of increasing performance need to be considered. Additionally, in case of removing partners, contracts need to be followed in order to avoid legal sanctions and to ensure that future cooperation is possible.

Besides measuring the internal performance of a VE, the external environment needs to be monitored too. Agility of a VE requires constant tracking for changes in the marketplace and in customer requirements. Besides reacting to internal performance indicators, the consortium may also be reconfigured as a response to changes in the external environment. For example, increased customer demand may require adding more manufacturing capacity to the VE through a new partner.

It is likely that the amount of exchanged information is substantial during the operation phase of a VE both within and between VE participant companies (Vallejos et al. 2007, p.591). In previous section 2.4.1.2, the VE ICT infrastructure was identified as a central enabler of inter-company collaboration. The infrastructure should be designed in a way that it is not dependent on a particular enterprise or the number of enterprises in a VE (Aerts et al. 2002, p.314). This way, the infrastructure is likely to inhibit less the evolution of a VE, by providing more scalable and flexible basis for VE operation. Additionally, the created infrastructure, and also a VE in general, should be able to cope with irregular events that may occur during VE operation (Camarinha-Matos & Afsarmanesh 2001, p.341; Vallejos et al. 2007, p.591). More discussion about ICT infrastructure requirements follows in section 2.5.1.

2.4.3 Dissolution

A VE is established for a specific business opportunity and it is likely to dissolve when the business opportunity is met or when it becomes clear that the given opportunity can or could not be met. Additionally, a VE can stop existing when the partnering companies agree to go their separate ways (Camarinha-Matos & Afsarmanesh 2001, p.338; Cardoso & Oliveira 2005, pp.16-23), regardless of the opportunity. It has also been suggested that when a VE business venture has stable market potential, the participants may build longer-term relationships (Aerts et al. 2002, p.312; Nayak et al. 2001, p.82). In the aforementioned case, a VE stops existing and another form of collaboration is created.

In the dissolution phase, it is necessary to share the profits that were created during the VE operation, according to the plan and individual performance of partners (Carvalho et al. 2005, p.362), and following the contracts signed by the partnering companies (Cardoso & Oliveira 2005, p.24). It is as necessary to share the losses, as there are many real life examples of failed VEs, that don’t make the headlines as often as the successful ones (Chesbrough & Teece 2002, p.127). The participants should also try to benefit from the new information acquired during VE operation (Vallejos et al. 2007, p.591), in order ensure success of future VE cooperation. It is also wise to preserve the technological components, for instance, “integration components” that were created or obtained for VE ICT infrastructure configuration. It is possible that these can be recycled if companies decide to join other VEs or used in other collaborative projects. Hence, the preparedness of an individual company to join VEs is likely to improve after the first VE experience.

Dismantling a VE can be a bit tricky on some occasions, especially if it has succeeded and a product or service is generated as a result of the collaboration. For instance, the responsibility of a product manufacturer remains to the end of the product life cycle (Camarinha-Matos & Afsarmanesh 2003, p.338). The general liabilities after dissolution should be determined in the contracts that are signed when companies join VEs. There is also a need to take care of the after-sales service even if the VE has stopped existing (Kanet et al. 1999, p.32). After-sales service may require that some partners stay committed to the VE even if most of the companies have dissolved. Some of the processes that were set-up during the formation and operation phases may therefore need to be kept alive. For example, spare part orders and invoices may need to be exchanged between a set of partners. Hence, some parts of the ICT infrastructure need to still remain connected and information flow existing between a selected set of partners. Due to VE dissolution being in some cases only partial, it is said that the dissolution phase is a special case of reconfiguration (Cunha & Putnik 2006, p.126).

2.5 ICT INFRASTRUCTURES FOR VIRTUAL ENTERPRISES

Nagel envisioned in an interview (Byrne 1993, p.38) that technology could make the formation of VEs as easy as it is to connect home video system components made by different manufacturers. This kind of interoperability essentially requires that the components have standard interfaces. As identified in previous sections, the current situation is that VEs are formed out of heterogeneous companies, in different industries, to serve different purposes and undergo different life cycles. The heterogeneity derives from different factors, including differences in companies' internal information systems, working methods and business practices (Camarinha-Matos & Afsarmanesh 2007, p.119). Additionally, the preparedness of companies to participate in a VE has been identified to vary considerably, depending on multiple factors, including whether they are formed of companies that are part of a VBE. In order to reap the benefits promised by agile VE, flexible as well as generic VE ICT infrastructure supporting the whole VE life cycle needs to be set up. This has proven to be a tough task, among other things, due to the aforementioned heterogeneity. (Camarinha-Matos & Afsarmanesh 2003, p.140)

This section maps the requirements for VE ICT infrastructures, for instance, what is needed technology-wise to support the dynamicity promised by the VE paradigm. Mapping the requirements is followed by discussion about the technologies used and the major obstacles in VE ICT infrastructure configuration. The findings in this section are used to evaluate the impact that the VE paradigm has on B2Bi.

2.5.1 REQUIREMENTS FOR THE ICT INFRASTRUCTURE

In section 2.4.1.2, enabling safe and coordinated exchange of technical as well as business data, support for different business processes and integration between VE companies were identified as focal duties of the VE ICT infrastructure. Above, it was stated that the infrastructure should be designed flexible, as well as generic, and to support the whole VE life cycle, namely the phases of creation, operation and dissolution. Flexibility in this context means the degree to which the ICT infrastructure resources are shareable and reusable (Duncan 1995, p.42). Moreover, flexibility is regarded as the ability of an ICT infrastructure to be rapidly adapted to new business processes (Aerts et al. 2002, pp.313-314; Camarinha-Matos & Afsarmanesh 2003, p.142) The support for new processes needs to be realized in a way in which the existing components of an infrastructure can be replaced and new ones added without changing the overall architecture (Aerts et al. 2002, p.314). This may include, for instance, integrating new application-level components to the infrastructure or replacing existing ones transparently (Umar, Missier 1999). Aerts et al. (2002, p.314) have identified VE change cases where VE ICT infrastructure is required to be flexible:

- An enterprise is added to the VE.
- An enterprise is removed from the VE.
- A VE adds new products to its portfolio.
- A VE merges with another VE.

The first three scenarios represent basic events that occur when a VE is dynamically reconfigured, while the last option of two VEs merging is more unlikely. As stated in section 2.4.2, the infrastructure should not be dependent on any particular company or the amount of companies in order to fully support dynamic reconfiguration. In other words, the infrastructure should be designed “self-annealing”, which means that removing a component from the infrastructure does not break the whole system (Goranson 2003, p.116).

Another possible requirement for a VE ICT infrastructure is the aforementioned generality. Generality refers to the ability to deal with the needs of various application domains (Camarinha-Matos & Afsarmanesh 2003, p.142). Generality is required when a VE comprises of companies that represent multiple fields of business. Additionally, the ICT infrastructure has been identified as the enabler of inter-company interaction and integration. To fully enable the interaction, VEs may require integration on four distinct levels (Camarinha-Matos & Afsarmanesh 2003, p.142):

- ***Level 1:*** Safe communication and information exchange, including business and technical information transactions.
- ***Level 2*** Integration of enterprise applications that run at different enterprises.
- ***Level 3:*** Business process integration, including coordination of distributed business processes.

- ***Level 4:*** Teams integration, including support for collaboration among professional teams with members from separate organizations.

The first three levels fall largely into the domain of B2Bi. Additionally, coping with enterprises' legacy systems is seen as a necessity for any given VE ICT infrastructure (Camarinha-Matos & Afsarmanesh 2001, p.342). The role of enterprises' existing IT systems as well as IT infrastructure is twofold. While the individual systems and IT infrastructure are seen essential to the competitiveness of a firm (Duncan 1995; McKenney 1995, cited in Broadbent et al. 1999, p.158), they may also represent a significant barrier for business process redesign (Grover et al. 1993, cited in Broadbent et al. 1999, p.158; Wastell et al. 1994, cited in Broadbent et al. 1999, p.158). Given the requirement of rapid, as well as low-cost, formation of VEs and the significant investments required for new IT systems, it is unrealistic to assume that companies would acquire new systems for individual VE collaborations. Instead, extending the functionality of existing systems may be required, for instance, because many legacy systems have not been designed to support B2B interaction. To have the VE enterprises to interact with the rest of the VE via the ICT infrastructure, each node may be required to develop an interface or a mapping layer. (Camarinha-Matos & Afsarmanesh 2001, p.342) The infrastructure potentially has to satisfy many other requirements, including knowledge management (Vallejos et al. 2007, pp.594-595), agreement negotiations (Camarinha-Matos & Afsarmanesh 2007, p.132), management and support services for VE administration (Umar & Missier 1999, p.10). However, these are not discussed further in this thesis.

2.5.2 CONFIGURATION OF THE ICT INFRASTRUCTURE

The configuration of a VE ICT infrastructure is not a trivial task, as noted above. A question arises, how to set up an interoperable infrastructure, given the heterogeneity of VE participants (Camarinha-Matos and Afsarmanesh 2007, p.120)? In addition to conforming to the requirements discussed in the previous section, the VE ICT infrastructure configuration should not inhibit rapid formation of a VE, or considerably increase transaction costs caused by VE reconfiguration. Basically, VEs should be able to configure a more flexible infrastructure in a potentially shorter period of time than more traditional collaborations.

While there are many proposals for building the technological foundation of a VE, (see Camarinha-Matos and Afsarmanesh 2003, pp.140-141) the VE domain is lacking a widely accepted reference infrastructure (Camarinha-Matos and Afsarmanesh 2003, p.141). Reference models are important in assisting the design and implementation of an integrated enterprise system with structured methodology and formalization of operations, as well as support tools (Burkel 1991, cited in Chalmeta & Grangel 2003, p.142) What this means is that there are no generic guidelines how to utilize the prospective technologies to support the configuration of a VE ICT infrastructure. Hence, every VE project needs to invest resources in developing its own infrastructure from scratch, causing deviation from its main focus (Camarinha-Matos & Afsarmanesh 2003, p.140). The VEs that are derived from support organizations, for instance VBEs, are likely to be less affected because they may have developed basis for common interoperable infrastructure a priori to the VE creation.

According to Umar and Missier (1999, p.5), “architectures for VEs sit at the crossroad of several well-known areas or research and engineering, namely automatic access to heterogeneous data sources, interoperability and standardization of distributed systems’ interfaces and access mechanisms, integration of distributed processes (workflow), and component-based software engineering.” Some of the main domains of technology and standards that are seen relevant for VE ICT infrastructures are presented in Table 1. A number of the domains and examples presented in Table 1 are currently active in B2Bi field, and thus paid more attention to in section 3. This discussion also involves more recent technologies and standards.

Table 1. Potential enabling standards and technologies for VE ICT infrastructures (Camarinha-Matos and Afsarmanesh 2003, pp.142-143, modified).

Domain of technology or standardization	**Examples**
Open interoperable network protocols	TCP/IP, SOAP, HTTP, RMI
Open distributed object-oriented middleware services	J2EE Framework, CORBA Framework, ActiveX Framework
Information / object exchange mechanisms and tools	XML, ebXML, WSDL

Standardized modeling of business components, processes and objects	EJBs, OAG
Business process modeling tools and languages	UML, PSL
Open and standard business process automation and workflow management systems	WfMC, OMG JointFlow, many commercial products
Standard interfacing to federated multi-databases	ODBC, JDBC
Intelligent mobile agents	FIPA, OMG-MASIF, Mobile Agents
Open and standard distributed messaging middleware	JMS, MQ-Series, FIPA-ACC
XML-based e-commerce protocols	Biztalk, RosettaNet, OBI,
Web-based integration technologies	Servlets, JSP, XSL

There are several proposed approaches for VE ICT infrastructure configuration. For example, a VE ICT infrastructure can be built by extending the existing enterprise systems of VE partners with a cooperation layer handling the cooperation events of a VE, as was done in the PRODNET project. The PRODNET cooperation layer consists of an information management system, a workflow-based coordination engine and safe communications infrastructure, which is built on Internet technology. The cooperation layer is installed at each enterprise, and internal integration is needed between the enterprise applications and the cooperation layer. (Camarinha-Matos & Afsarmanesh 2003, pp-114-145) Another suggested approach is leveraging mobile agent technology. This approach may require, for example, installing docks and service bridges at each enterprise that provide the basis for mobile agent operation. Again, internal integration is needed between the service bridges and internal enterprise systems. (Aerts et al. 2002, pp.317-319)

Given the intricate requirements, it is not surprising that several VE research projects have sought a solution to the VE infrastructure configuration and integration problem by introducing state-of-the-art technologies, including agent technologies (see Camarinha-Matos & Afsarmanesh 2001; Aerts et al. 2002; Shen et al. 2007) and semantic Web services (see Shen et al. 2007; Gang et al. 2007). A variable to be considered when configuring the VE ICT infrastructure is the preparedness of partners, especially SMEs and non-ICT firms, to implement solutions based on such technologies. In the clothing sector VE example referenced in section 2.2, majority of the subcontractors were not ready to adopt the new technologies that were introduced. Camarinha-Matos and Afsarmanesh (2003, p.142) state that non-IT companies may commonly conceive these technologies as quite disturbing. Albeit some of the aforementioned technologies are openly available and relatively low-cost, the total cost of implementation needs to be evaluated. Moreover, can companies trust on solutions that are built on technology that is still in its infancy? Should the infrastructure be built on technology that is proven, for example, in the field of electronic business? The discussion that revolves around technology is carried on in section 3.

2.6 Requirements of virtual enterprises for business-to-business integration

VE paradigm is promising companies the flexibility that is needed to cope with the requirements of the dynamic market and global competition. A VE is characterized by fast and cheap formation, network reconfiguration, flexible company interfaces and limited existence. A VE should be realized through extensive use of modern ICT. ICT is a prime enabler of interaction between VE companies and most important feature of a VE, that is, reconfiguration dynamics. However, the heterogeneity and varying preparedness of VE companies significantly complicates the task of ICT and the integration between VE companies.

A technological solution that seeks to integrate VE companies needs to first tackle the aforementioned heterogeneity. Moreover, VEs may require inter-company integration in four different levels: safe transactions, applications, business processes and professional teams. The first three levels fall into the domain of B2Bi. The solution should also provide support for the entire VE life cycle, including partners search and selection, as well as dynamic reconfiguration. To enable dynamic reconfiguration of a VE, for instance, changing business processes and partners, the solution should be designed flexible and generic. Additionally, the solution should not be dependent on a single company. To further add challenge, the initial configuration and changes should be made with minimum transaction costs, including time.

The requirements listed above constitute a unique set of requirements to the B2Bi. However, they are rather generic and thus expected to vary depending on multiple issues, such as the companies that form a VE and the business opportunity a VE is formed for. The following chapter 3 focuses on the VE integration issue from the perspective of B2Bi. Chapter 3 introduces the main approaches, components, risks and trends of B2Bi. These are reflected to the unique requirements set by VEs and consideration is put on how these can be used to solve the integration issues that revolve around VEs.

3. Business-to-business integration

According to Samtani (2002, p.xii), the objective of business-to-business integration is to “achieve end-to-end automation and integration of cross-organizational business processes, data, applications and systems”. A prerequisite for B2Bi is that the information being conveyed to trading partners is automatically extracted from the back end systems and the data coming from the partners is again automatically inserted to these systems (Bussler 2002a, p.163). Hence, value in B2Bi is essentially created through automation. En route to B2Bi, three considerable barriers that arise from the heterogeneity of partners’ application semantics, information content and platforms need to be overcome (Linthicum 2004, p.454).

In chapter 2, both VE formation and reconfiguration were identified as events that call for integration. Integration is required between VE participants at different levels, ranging from simple information exchange and applications to business process and teams integration. When a VE is reconfigured, adding new applications or new processes require integration. The challenges that the VEs pose for integration are mainly a result of the heterogeneity of VE participants. The severity of these challenges depends on multiple variables, including duration of the opportunity, number of cross-organizational business processes and partners, flexibility requirements and preparedness of partners. The flexibility requirement and short-term relationships are intrinsic characteristics of VE integration problem.

VE integration is seen by some to differ from traditional integration to such a degree that a new paradigm for VE integration is proposed (Putnik et al. 2005, p.24). However, in this thesis the VE integration is analyzed in the context of B2Bi. In this chapter, the main approaches to B2Bi, as well as the central building blocks of B2Bi, are presented and these are reflected to the requirements set by VEs. Consideration is put on how these can be used in solving integration problems, also in the context of VEs. Additionally, this section tries to map the largest risks that lurk in B2Bi projects, as well as outline the future directions of B2Bi, and evaluate how these are relevant to VE integration.

3.1 Approaches to business-to-business integration

Before choosing an approach to B2Bi and scoping the integration project, several factors need to be considered. These include, for instance, coupling or cohesion between systems to be integrated (Linthicum 2004, p.27), degree of synchronization needed between B2Bi parties, integration agreements (Samtani 2002, p.25), timeliness, format, size, as well as volume, of data exchanged between B2Bi parties, communication infrastructure, security and resiliency (Lam & Shankararaman 2004, p.44). Designing for integration is likely to become rather complex, when these factors are combined with the unique set of requirements set by VEs. In this section, the generic approaches to B2Bi are presented and these are reflected to the requirements set by VEs.

3.1.1 A unique approach for a unique integration problem

Every enterprise or trading community has a unique set of integration problems and therefore each solution will require its own approach (Linthicum 2004, p.6). As noted in chapter 2.4.1.2, VE companies need to integrate in a specific context, consisting of processes that represent only a portion of all the possible conditions. For instance, the integration of companies' ICT systems may be limited to exchange of information on availability, prices and products (Aerts et al. 2002, p.315). Thus, VEs do not necessarily need to seek for full integration, which may be required, for instance, when two companies merge. Additionally, VEs should be both formed and reconfigured rapidly, as well as with minimum expenditures. The aforementioned factors alone are likely to crop out the most exhaustive integration approaches, at least in case of short-term VEs.

Generally, four distinct approaches to B2Bi can be defined: information- or data-oriented, portal-oriented, business process-oriented (Samtani 2002, p.25; Jhingran et al. 2002, p.555) and service-oriented integration. (Linthicum 2004, p.6) It is possible, and often necessary, to combine these approaches (see Figure 8) to achieve appropriate level of integration (Linthicum 2004, p.32, Samtani 2002, p.47). These approaches to a great extent cover the full spectrum of B2Bi, ranging from simple information integration to integration of complex inter-company business processes. Therefore, it seems unlikely that VEs can, or even need, completely surpass these approaches, when they seek a solution to their integration problems.

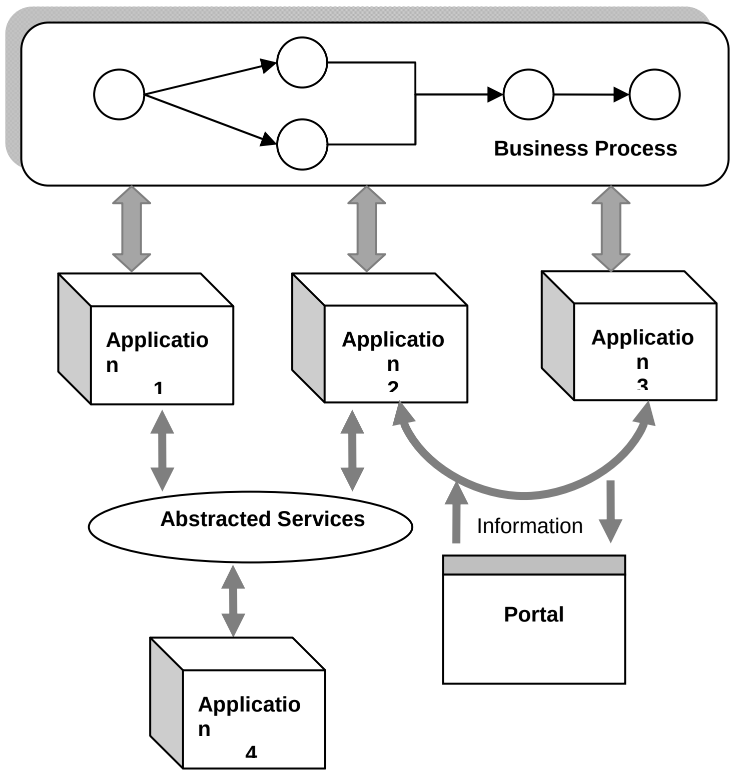


Figure 8. A mixture of different integration approaches (Linthicum 2004, p.33, modified).

3.1.2 INFORMATION-ORIENTED INTEGRATION

Enterprise information typically resides in multiple places, including different databases, enterprise resource planning (ERP) systems, supply chain management (SCM) systems and legacy systems. The objective of information-oriented integration (IOI) approach is to integrate the data extracted from these systems, as well as other sources, with the external data sources of trading partners (Samtani 2002, p.49). In the IOI approach, integration takes place at a relatively low level of abstraction, that is, mainly between databases or between application programming interfaces (APIs) of the data sources (Linthicum 2004, p.6).

Multiple ways to realize IOI have been created, including data replication, data federation and interface processing. Data replication is a process of extracting data from the source database and placing it into the target database (see Figure 9). (Linthicum 2004, p.7) Hence, replication involves maintaining a physical copy of data in each of the databases (Haas 2007, p.31). A middleware solution is needed between databases to account for the differences in schemas and platforms, as well as for providing the infrastructure to exchange data (Linthicum 2004, p.7). The data is typically exchanged in the form of messages. It is possible to leverage, for instance, an XML layer on top of IOI, with which data can be transformed to XML format and exchanged between trading partners. (Samtani 2002, p.51-66) A key prerequisite is to understand both source and target data, in order to be able to create an integrated, as well as standardized, data representation. The significance of standards, including XML, is considered in the context of B2Bi and VE integration in section 3.2.5.

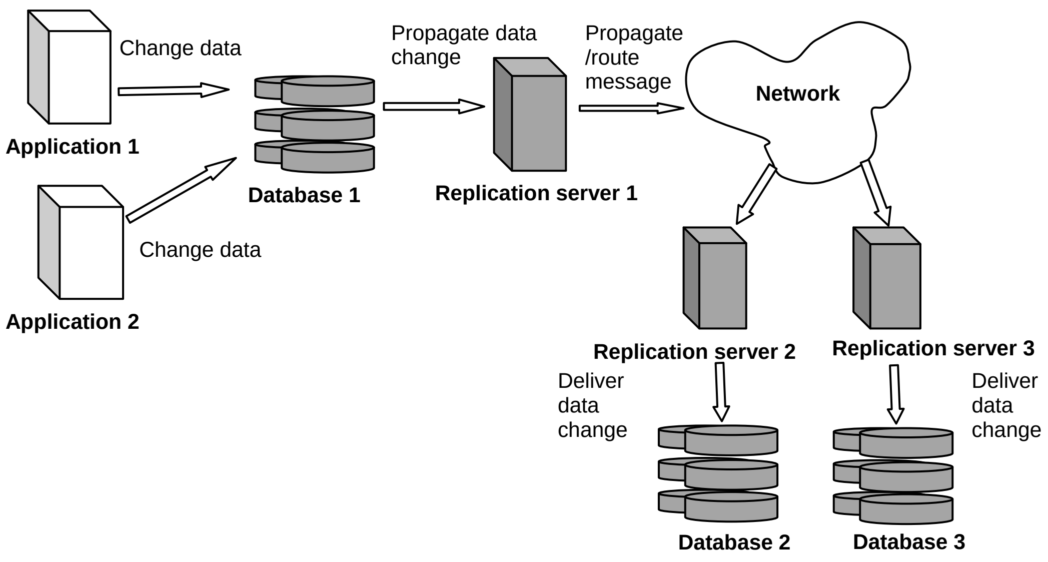


Figure 9. An example scenario of data replication (Samtani 2002, p.51, modified).

In database federation, a virtual representation of multiple physical data sources is created to address integration needs (Haas 2007, p.31). A federation solution places a layer of middleware, including a relational database management system (Haas et al. 2002, p.580), between the physical distributed data sources and applications that use the data (Linthicum 2004, p.9). With this type of a solution, applications can search and manipulate data of the sources through a single structured query language (SQL) interface, as if it were in a single database management system. Besides databases, the data sources may be, for instance, flat files, XML documents, message queues or even Web services. (Haas et al. 2002, p.580) One possibility for the database federation architecture is presented in Figure 10. This type of integration approach may be desirable in a more stable inter-company relationship, for example, in case two companies merge through acquisition and need to consolidate their operations (Haas 2007, pp.31-33). However, it is not clear where a database federation solution should reside in a naturally distributed and volatile environment, such as, in a VE.

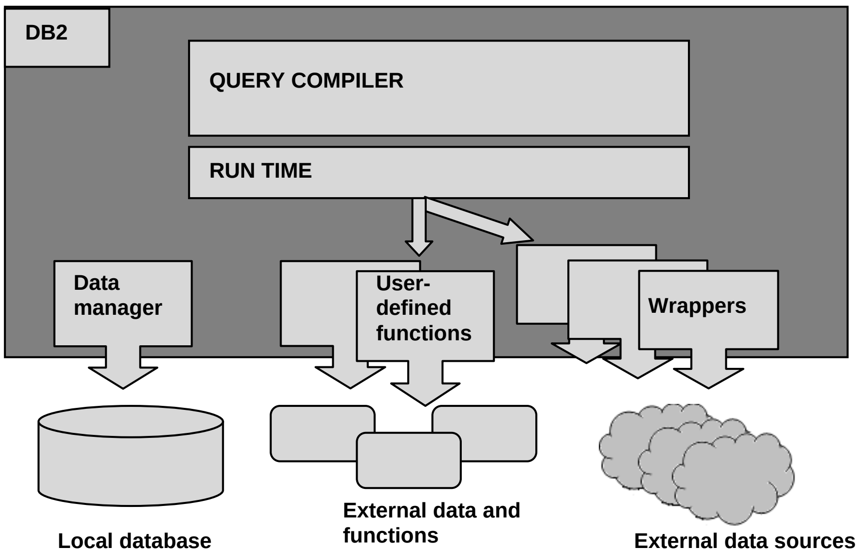


Figure 10. DB2 architecture for database federation (Haas et al. 2002, p.582, modified)

Interface processing allows custom or packaged applications from different vendors, such as SAP and Oracle, to be connected through well-defined, but often proprietary APIs. A specific type of middleware, integration server, typically offers a wide range of pre-built application adapters that allow dissimilar applications to communicate in a relatively short period of time. (Linthicum 2004, pp.9-10) The role of adapters is to enable non-invasive integration by separating the API from the messaging infrastructure, thereby promoting loose coupling (Papazoglou & Van den Heuvel 2007, p.400). Besides adapters, the interface processing middleware solution needs to deliver data transformation capabilities to account for the differences in data content and application semantics (Linthicum 2004, pp.9-10). A simplified example on how interface processing externalizes information from packaged applications is given in Figure 11.

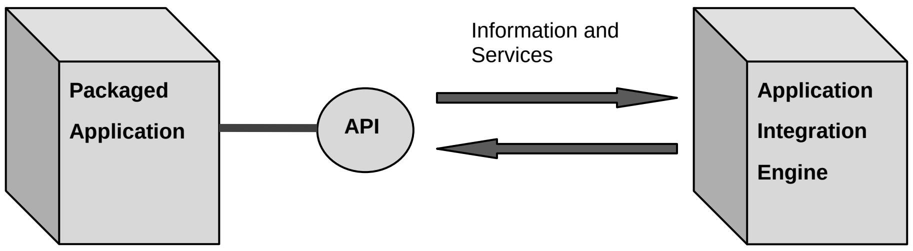


Figure 11. Information externalization using interface processing (Linthicum 2004, p.9)

The bottom line of IOI is to deal with the exchange of simple information, for instance, orders coming in to a system and confirmations going out, without altering the source and target systems (Linthicum 2004, p.5-27). Additionally, this approach aims to eliminate non-standard interfaces and enable sharing corporate data over the Internet (Samtani 2002, p.49). With IOI, it is possible to cope with different kinds of back end systems, since they generally know how to produce and consume data (Linthicum 2004). The solutions based on this approach conform to level 1 and level 2 VE integration that were defined in section 2.5.1. Given the diffusion of this approach (Linthicum 2004, p.26), many companies have already mechanisms in place to externalize business information to their partners. The question still remains, as much for VEs as traditional B2Bi, how to quickly overcome the semantic mismatches that exist between data sources and how to effectively bring integration to a higher level, namely the level of business processes?

3.1.3 SERVICE-ORIENTED INTEGRATION

Service-oriented integration (SOI) aims to bind data sources together using services as integration building blocks. The promise of service-oriented computing is to allow low-cost and rapid assembling of distributed software components into a loosely coupled network of interacting services that create flexible business processes, as well as agile applications, and span different organizations and platforms (Papazoglou et al. 2007, p.38). This promise seems intriguing also from the perspective of VEs, due to the implications on low-cost, flexibility and distributed nature of interacting services.

In this context, services are described as platform-independent and self-describing software components, which are able to perform a variety of tasks, including simple requests, as well as complex business processes

(Papazoglou 2003, p.3). The idea is that an application or any piece of code can be exposed as a service (Papazoglou et al. 2007, p.38) through well-defined interfaces that are implementation-independent and based on open standards. The independence of implementation means that the interface is separated from the actual implementation, so that the service consumer need not know how its requests to the service provider are executed. Additionally, services are loosely coupled, can be dynamically discovered and aggregated with other services to constitute a composite service. (Mahmoud 2005) Moreover, services can be provided by different companies and communicate over a network, which makes them a viable alternative for providing a distributed infrastructure for both intra-enterprise and inter-enterprise integration (Papazoglou 2003, p.3).

An architectural approach that has been developed to address the needs of service-oriented integration is called service oriented architecture (SOA). A typical implementation of SOA relies on specific service types, called the web services (Mahmoud 2005). Web services are considered as integration components (Samtani 2002, p.29), and thus discussed in section 3.2.3. The promise of SOA is interoperability between heterogeneous applications and technologies, while enabling the reuse of existing systems and development of new services from these (Mahmoud 2005). The basic SOA-model consists of three actors, service provider, service requester and service registry and additionally. Moreover, it entails three actions, publish, subscribe and bind, taking place between the actors (see Figure 12). In the SOA-model, the service provider describes the services it offers, using a standard language, and publishes the descriptions in a registry. The Service requester uses the find-action to search for a suitable service in the registry and once a match is found, it binds to the service provider using the information received from the registry. This information includes the protocol that is used to invoke the service, as well as the structure of the messages for invocation and response messages. (Leymann et al. 2002, p.199)

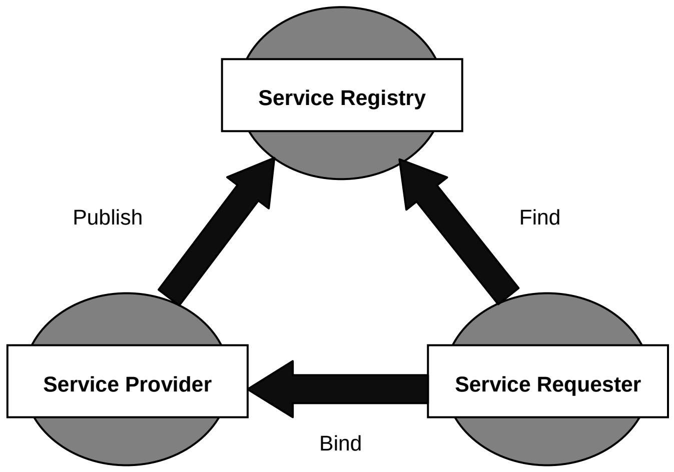


Figure 12. The basic SOA model.

The SOA-model has also been considered to provide support for the VE life cycle. A suggestion is to consider potential VE members as "service providers" and the realization of their collaborative behavior as a set of interacting services (Afsarmanesh & Camarinha-Matos 2000, cited in Camarinha-Matos & Afsarmanesh 2007, pp.122-123). This approach requires that one entity keeps a registry of services offered by companies (Camarinha-Matos & Afsarmanesh 2007, p.123). According to Papazoglou (2003, p.3), companies can publish their core competences programmatically over the Internet through services. Camarinha-Matos and Afsarmanesh (2007, p.123) have raised justified doubts against this statement by questioning whether even all competences can be represented as services. As identified in section 2.4.1.1, VE partners search and selection is more than a matching process based on competences and depends also on matters of subjective nature.

To implement SOA, some form of infrastructure is required to enable inter-service communication. Multiple solutions for the infrastructure exist, as the services may reside in a single server or span multiple networks. (Papazoglou 2003, p.4) Basic Web service technologies enable point-to-point integration by leveraging existing Intranet or Internet infrastructures (Keen et al. 2004, p.63). However, this approach requires an interface to be developed for each connection, which leads into tighter coupling and is more difficult to maintain due to the point-to-point topology (Papazoglou & Van den Heuvel 2007, p.393). An alternative that promises to provide a run-time infrastructure and a proper set of tools for the materialization of SOA is the enterprise service bus

(ESB) (see Figure 13) (Schmidt et al. 2005, p.781). The idea is that all participating components or services can be plugged into the bus, which provides a reliable infrastructure for messaging. The ESB builds on middleware technology, and it provides service request routing, message translation and infrastructure management capabilities, support for different integration styles or adapters, with proper quality of service and security levels, as well as conformance to the SOA principles. (Keen et al. 2004, pp.74-85)

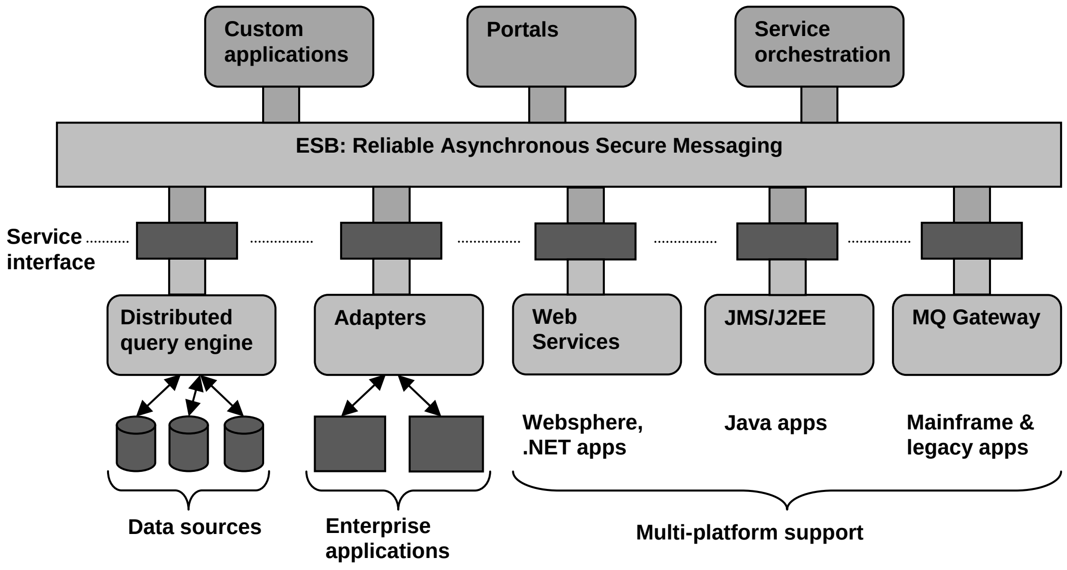


Figure 13. An example of an ESB that connects diverse applications and technologies (Papazoglou & Van den Heuvel 2007, p.393).

While the SOI approach provides an interesting proposal for B2Bi, it also incorporates some concerns. Firstly, the effort required to describe the service interfaces, to implement the services and to configure the infrastructure, has to be taken into account. The required effort is one of the reasons why companies are instructed to take an incremental approach for moving towards services and SOA (Linthicum 2004, p.85; Mahmoud 2005). Secondly, the service requestors that can be, for instance, applications or processes and either internal or external to a company, need to create mechanisms for finding services, as well as for invoking them. Thirdly there are issues related to the dynamic reconfiguration of the infrastructure, end-to-end security, differences in service description semantics, service composition for enabling adaptive processes and both service management and monitoring (Papazoglou et. al 2007, pp.41-44). These issues need to be reflected before exposing companies' capabilities as services.

At the most primitive form, SOI approach enables integration between heterogeneous and distributed software components. This corresponds to the VE integration levels 1 and 2, identified previously in section 2.5.1. Most essentially, it solves the problem of platform heterogeneity with well defined interfaces based on open standards. As such, it does not solve the semantic discrepancies that exist between different services. Additionally, business process level integration, which is desired both in case of VEs and traditional B2Bi, is not achieved through basic services or SOA-model implementations. In addition to the effort required to create the service interface, as well as its actual implementation, SOI approach often requires costly changes to existing systems, also in case of Web services (Linthicum 2004, p.81). Hence, if a VE chooses SOI approach for integration, it may be necessary that the potential VE participants have their services ready and waiting, in order to better support rapid creation, as well as dynamic reconfiguration, of the VE. All in all, the advantages of this approach seem to have superseded the concerns, as services and SOA remain a hot topic in the integration domain. Discussion about services in the VE context is carried on in sections 3.2.2 and 3.4.

3.1.4 Business process -oriented integration

As noted in section 3.1.1, the IOI approach goes about the integration problem at a low level, by simply providing access to or moving information. SOI approach is focusing attention from the underlying technologies to a higher layer of the "integration stack", namely services. Business process oriented integration (BPOI) sits on top of these two approaches and partially portal-oriented approach as well, by providing a common business process model for controlling the invocation of services and movement of data across different systems, both inter- and intra-company. (Linthicum 2004, pp.55-56) The carrying idea of BPOI is to leverage software tools for modeling business processes while simultaneously connecting them to the underlying data sources (Gruden & Strannegard 2003, p.8). This approach has emerged as an aggregation of preexisting concepts and tools, including business process modeling, business process automation and workflow tools (Linthicum 2004, p.63).

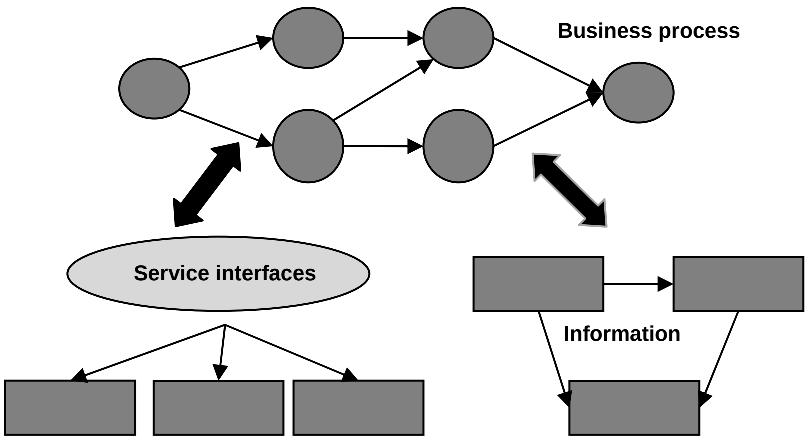


Figure 14. BPOI adds a control layer on top of information- and service-oriented integration (Linthicum 2004, p.59, modified).

Consider the following situation: when a component manufacturer is shipping its products, it needs to communicate with both the consignee and the logistics provider. It first initiates a shipping order to the logistics provider and receives a confirmation. Next, the manufacturer informs the consignee of the shipment, and as the goods arrive, the consignee sends a receipt to the manufacturer. Lastly, the manufacturer initiates an invoicing process. Rather than just concentrating on the exchange of required data or messages between the business partners, BPOI focuses on controlling the whole process. (Tsai et al. 2006, pp.48-49) The control is gained with an additional control logic layer, which is placed on top of integration technology. This technology enables the underlying systems to be bound into a single multistep business process model that can carry out the unique functions required by the business process. The steps of a given business process model must be executed in the correct order, with proper information, control sequences and state maintenance. (Linthicum 2004, p.56)

As noted above, BPOI uses a common process model for controlling the cross-organizational business processes. Creating this model involves drawing a diagram that represents the processes, resources, sequence and logical order of events, as well as the movement of information between the systems. The requirement is that the process model is able to produce events that the underlying systems are able to understand, and react to the events that the systems communicate back. The process model needs to handle exceptions and maintain state of a process for possibly quite long period, as the component manufacturer example given above indicates. A common process model also enables monitoring the status of processes and optimization of them. Optimization may occur at any given time, because the model is independent of the underlying systems that are involved in the process. Additionally, a graphical representation of the process hides the complexities of the underlying systems, thus allowing business personnel to get more involved in the integration. (Linthicum 2004, pp.60-74)

Of course, creating a process model does not automatically connect the systems involved. Neither does it execute by itself. The technology that is required for modeling the processes, as well as executing and monitoring them is discussed in section 3.2.4. This technology, or the control logic, is at the upmost level of hierarchy in BPOI. It needs to be associated with traditional middleware, which provides the needed functionalities, such as information routing, transformation, rules processing and messaging, thus constituting the bottom layers of the BPOI hierarchy (Linthicum 2004, pp.69-70).

BPOI represents a more business-led approach to the integration problem than other approaches discussed, and it is expected to be the “next wave” of integration (Linthicum 2004, p.55; Gruden & Strannegard 2003, p.8). However, this approach requires that the complexity of binding the back end systems to the process model is significantly reduced (Linthicum 2004, p.70). Thus, in VE context, BPOI is unlikely to alleviate the burden of integration. It does, however, offer an approach for integration, which corresponds to VE integration level 3, discussed in section 2.5.1. Employing Web service technologies in conjunction with BPOI is regarded as one possible solution that helps business process integration tools’ interfacing with the underlying systems and unleashing the full potential of this approach (Tsai et al. 2006, p.49). Further, process-oriented standards, including RosettaNet and Electronic Business Extensible Markup Language (ebXML) are expected to be valuable for BPOI, as they provide common mechanisms and semantics for BPOI (Linthicum 2004, p.71; Tsai et al. 2006, pp.49-51). These standards are discussed in section 3.2.5.

An interesting question is where the business process models should reside in a VE? Different process topologies have been identified – peer-to-peer, hierarchical and mixtures of these – that partly answer this question. In a peer-to-peer structure (see Figure 15A) the inter-process interaction requires that the operations of a requesting process (A, C, and T in Figure 15A) are linked with the "performing" operations (O, R, and E in Figure 15A) in the other process. Thus, the processes are modeled within each enterprise and these models interact to achieve a common goal. The linking can be done, for instance, by exchanging messages between the two processes. Additionally, the necessary synchronization between the involved processes requires cooperation of the participating partners, which can be achieved with private agreements, or it can be based on public standards. For instance, the ebXML business process specification schema could be used to define the required interactions and partner roles. The strength of this topology is that it is likely to support realization of dynamic business relationships. (Leymann et al. 2002, pp.204-206)

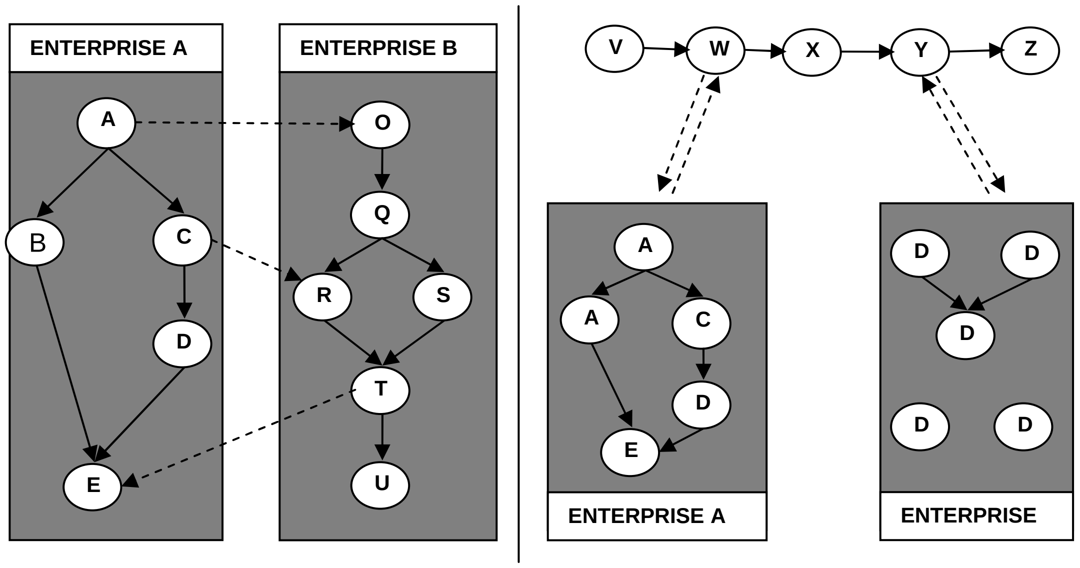


Figure 15. A. Peer-to-peer business process. B. Hierarchical business process. (Leymann et al. 2002, pp.205-206, modified)

In hierarchical topology (see Figure 15B), a top-level business process is created to describe the interactions between business partners and the steps needed to achieve a business goal. Since there is a top-level process, both overall process management and monitoring are likely to become easier than in peer-to-peer topology. However, in case of VEs, it's not clear who should host the overall process. As noted in chapter 2, the VE integration solution shouldn't be dependent on any single partner. One option is that a third party is assigned to this task and another one is that one of the involved parties is selected. This choice can also be made dynamically, but requires that a process needs to be modeled in standardized way so that it can be run in heterogeneous process execution environment. (Leymann et al. 2002, pp.205-206) Business process modeling language (BPML), which provides an abstracted execution model for collaborative and transactional business processes (Linthicum 2004, p.71), could prove valuable for this task. As in IOI and in SOI, standards appear to take the center stage.

3.1.5 Portal-oriented integration

According to Chandran et al. (2004, p.12), portals give their users a secure single point of entry, through an integrated and personalized Web-based interface, to a wide variety of information, applications and collaborative services. By leveraging portal-oriented integration (POI), companies are able to evade the often so exhaustive process of back-end system integration by "extending the user interface of each system to a common user interface", which typically is a Web browser (Linthicum 2004, p.99). Hence, this approach is fundamentally different than the other three approaches discussed previously in sections 3.1.2-3.1.4. With the use of a browser, portal users that belong to other organizations can interact with a trading partner's internal systems through a controlled interface. (Linthicum 2004, pp.99-103) Portals support business functions, involving online transactions, such as, sales, marketing and finance, as well as enterprise-wide communication and information sharing with different stakeholders. Currently, portals are an integral part of companies' B2Bi strategy, as they provide a fast and efficient way to establish and maintain business relationships. (Samtani 2002, p.68)

A typical portal use case in a B2B context involves a buyer logging into her company's account on a supplier's portal, checking product details and placing an order. Thereafter, it is possible to monitor the order and shipment status through the portal. (Samtani 2002, p.68) Practice has shown that it is not advisable to try satisfying the needs of all users with a single portal, due to complexity, cost, and required effort (Clarke & Flaherty 2003, p.17). This partially explains why portals are categorized of into horizontal and vertical. The domain specific vertical portals represent a specific portal instance, while horizontal portals provide the infrastructure upon which the vertical portals are built (Chandran et al. 2003, p.16). The relationship between vertical and horizontal portals is illustrated in Figure 16.

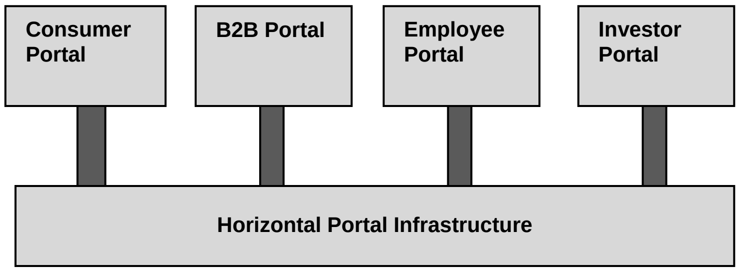


Figure 16. Horizontal portal infrastructure, upon which the vertical portals are built (Chandran et al. 2003, p.16, modified).

Creating a portal application involves designing the user interface and portal functionality, as well as deciding which information needs to be accessed and manipulated within the back-end systems with the portal application. A complete portal solution is normally realized with the following components: Web clients, Web servers, database servers, back-end applications and middleware, that is, application servers. (Linthicum 2004, pp.99-103) A specific type of application server, called portal server, is used in POI (Chandran et al. 2003, p.48). The portal server framework consists of three layers: presentation layer, services layer and connection layer (see Figure 17). The presentation layer delivers portal users the front-end Web interface through which they can access the data. The foundation upon which the portals are built is provided by the services layer. (Samtani 2002, pp.70-71) The service layer entails functionality, including personalization services for different users, collaboration services for teamwork purposes, information search facilities, portlets for attaching software modules, workflow management, security and administration, an API for integration purposes and publish/subscribe services (Samtani 2002, pp.71-74; Chandran et al.

2003, pp.23-24). Finally, the connection layer exists for attaching the portal framework to the back-end systems (Samtani 2002, p.74).

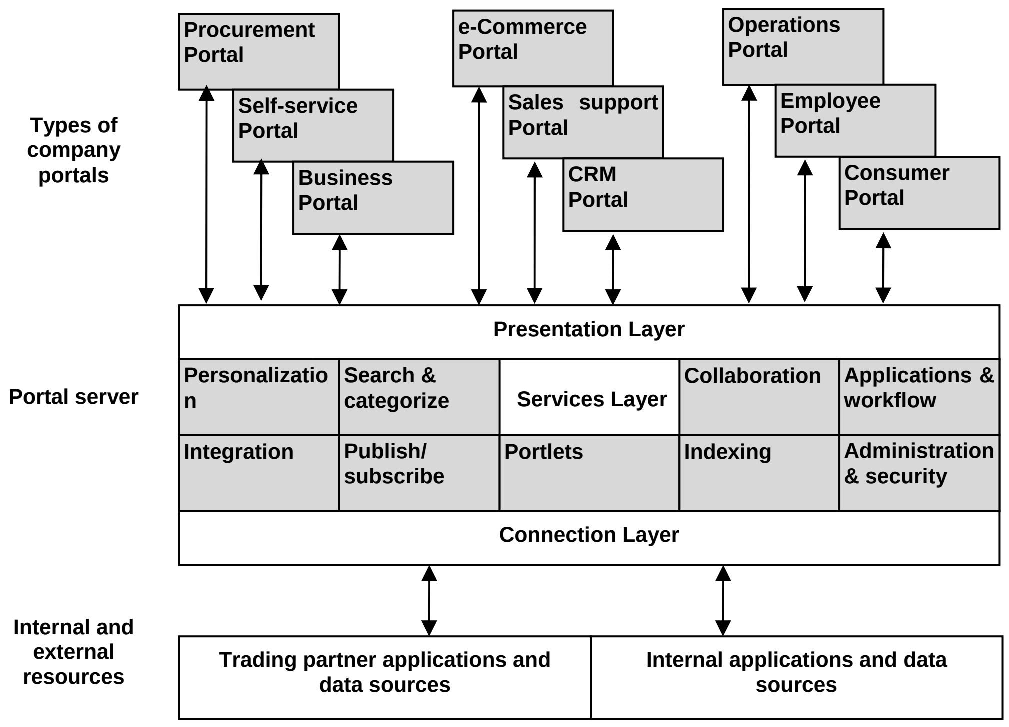


Figure 17. Portal framework and services (Samtani 2002, pp.70-79; Chandran et al. 2003, p.24).

In case of SMEs, system-to-system integration may not be a viable solution due to their primitive back-end systems or the lack of them the altogether. A portal can be built for SME users, which allows the same functionality manually through a Web interface that others are able to achieve through other forms of integration. (Kotinurmi 2007, p.53) Additionally, when the amount of transactions or exchanged information between business partners is small, the integration of back-end systems might be overkill. Besides serving SME users, portals can be built for serving customers as well. A portal functioning as a customer interface in VE context is considered, for instance, in (Aerts et al. 2002, pp.315-316) and (Vallejos et al. 2007, p.595). However, from the perspective of B2Bi, portal-oriented approach fails to satisfy the requirement of end-to-end automation of business processes. Partly for this reason, portals should not be perceived as a complete solution for VE integration (Camarinha-Matos & Afsarmanesh 2003, pp.145-146), especially if VEs seek to automate cross-enterprise data exchange and business processes.

3.2 Integration components

As discussed in section 3.1.1, several integration approaches have been created in order to solve different integration problems. A solution desiderata for B2Bi potentially entails many components, such as, enterprise application integration (EAI), Web services, business process management systems (BPMS), middleware, standards and security. In the following sections 3.2.1-3.2.6, these major B2Bi components are presented and their role in B2Bi, and VE integration, is evaluated.

3.2.1 Enterprise application integration

In B2B interaction, such as in VEs, there is a need to externalize intra-enterprise information to business partners. The prerequisite is that companies develop an unconstrained information flow across a company's internal systems (Erasala et al. 2003, p.70), since inter-organizational business processes may span several of these systems simultaneously (Samtani 2002, p.97). This seamless intra-enterprise flow can be achieved through EAI. The aim of EAI is to create a hub that allows the integration of data and processes inside an enterprise by enabling a variation of integration approaches, and supporting multiple technologies used by the enterprise currently and to be used in the near future (Charlesworth & Jones 2003, p.14).

The intra-enterprise integration is often realized through a dedicated EAI middleware solution, which enables autonomous data sources to communicate through a common interface layer (Lee et al. 2003, p.58). This solution is not to be confused with an ERP system, which aims to integrate internal business information, including financial and accounting, human resource, supply chain and customer information, under one software package (Davenport 1998, p.121). The known issues, an ERP system fostering inflexibility by pushing companies to toward generic processes (Davenport 1998, 123) and an ERP system being not able to fulfill all functionalities required by a company (Samtani 2002, p.109), prevent them in most cases from being effective solutions for EAI. In fact, ERP systems can be viewed as components that need to be integrated by EAI (Samtani 2002, p.109)

Conventionally, EAI has focused on intra-enterprise and B2Bi on inter-enterprise integration (Gulledge 2006, p.15) However, the gap between these two is closing as EAI and B2B services are being offered under one product by the software vendors (Samtani 2002, p.121). Additionally, major ERP vendors are offering EAI and B2Bi capabilities to extend their products to support integration with external systems. In this context, external means other than ERP. For example, SAP has developed its Exchange Infrastructure (XI) solution to address external integration issues (SAP 2007). Additionally, new integration approaches, for instance SOI approach, aim to bind heterogeneous systems together irrespective of their locations.

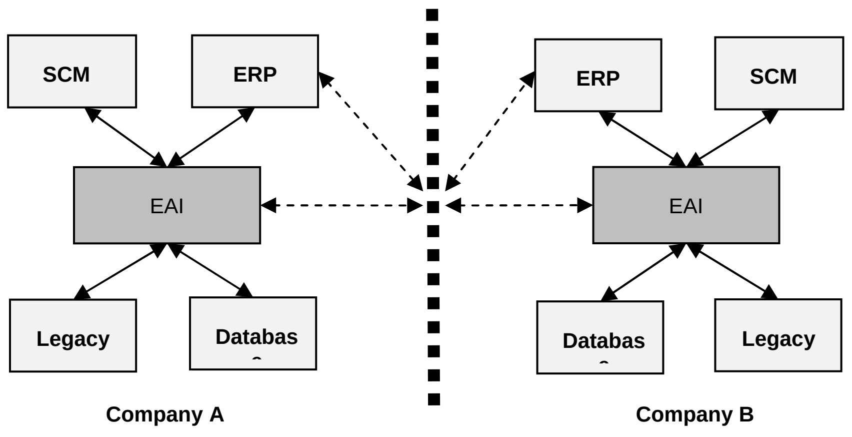


Figure 18. An extended view to EAI.

EAI is likely to affect VEs in two ways. Firstly, the preparedness of companies to enter VEs is likely to be hindered if intra-enterprise information is disconnected, since VE processes may require information from multiple systems simultaneously. Additionally, reconfiguration of a VE may require adding new processes and access to internal information sources that were not defined in the formation phase. As EAI is a rather complex process, the intra-company integration issues that may arise during VE formation or operation cannot be tackled on the fly, or at least cause integration time and costs to spiral upwards. Secondly, the ability of a company's information systems or EAI middleware to interface with external systems is also likely to enhance or inhibit the company's readiness for B2B interaction, and thus for VE participation.

3.2.2 Web services

In section 3.1.3, services were recognized as primary components of SOI and Web services as a preferred way of implementing SOA. Web services are essentially services identified by Uniform Resource Identifiers (URIs) (Papazoglou 2003, p.4), described by open, Internet-oriented and standards-based interfaces (UDDI Consortium 2001) and capable of interacting with other software components using XML-based messages that are exchanged via Internet-based protocols (W3C 2004). The central promise of Web services is low-cost composition of distributed applications (Papazoglou 2003, p.3), as well as accelerated application integration, both within and outside enterprise boundaries (Gottschalk et al. 2002, p.170).

To implement the three roles defined in the SOA model, that is, service provider, service requester and service registry, Web service technologies rely on three standards, Simple Object Access Protocol (SOAP) for communication, Web Services Description Language (WSDL) for describing the services and Universal Description, Discovery and Integration (UDDI) for the service registry specification. The common base language for Web services is XML. (Alonso et al. 2004, pp.151-152)

SOAP provides an XML-based protocol for messaging and remote procedure calls (RPCs) between service requesters and service providers over the Web (Curbera et al. 2002, p.86). A SOAP message contains two parts, an optional header that entails information about routing, security or handling of the message, and a body that carries the actual XML message or procedure call (Roy & Ramanujan 2001, p.70). To convey information over the Web, SOAP works with existing transports, such as HTTP, SMTP and MQSeries (Curbera et al. 2002, p.86). SOAP itself is designed for one-way asynchronous communication, which means that any more complex communication pattern, including request-response, requires exploiting the capabilities of an underlying transport protocol or middleware (Alonso et al. 2004, 156). Hence, SOAP covers the communication aspects, but the definition of what kind of messages are to be exchanged during service interaction is left for WSDL.

WSDL defines a format for an XML-document for describing Web services as a set of communication end points that can exchange certain messages. It consists of an abstract part and a concrete part. The abstract part provides a set of data type definitions, the operations supported by the service and input, as well as output, message formats used in service interaction. The concrete part gives information of what communication protocol to use and where to terminate the communication, namely the network address. (Curbera et al. 2002, pp.88-89) Finally, the service registry specification, UDDI, comprises of two parts: registry and APIs. The registry contains service descriptions in catalogs, while the APIs determine how to publish services, how to discover them and how to register to a service. Currently, private UDDIs are preferred over public ones, because Web service interactions tend to occur either within companies or among trusted parties (Alonso et al. 2004, p153-184).

However, WSDL, SOAP and UDDI are not all there is to Web services. An extended set of Web service components and infrastructure capabilities is presented in Figure 19. The three aforementioned technologies can be reinforced with mechanisms that involve service composition. At the most primitive form, service composition involves a Web service calling another Web service, thereby constituting a composite service (Alonso et al. 2004, p.141). Web services can also be used to build distributed business processes. A standards initiative based on Web services for providing a mechanism to describe, execute and share processes is Business Process Execution Language for Web Services (BPEL4WS) (Linthicum 2004, p.279). This is presented at the top of the stack in Figure 19. The vertical towers on the right side of the Figure 19 represent the infrastructure capabilities, including policy, security, transaction and management, that need to be implemented in order meet the requirements of today’s e-business (Gottschalk et al. 2002, p.172). A possible infrastructure solution is the ESB that was discussed in section 3.1.3.

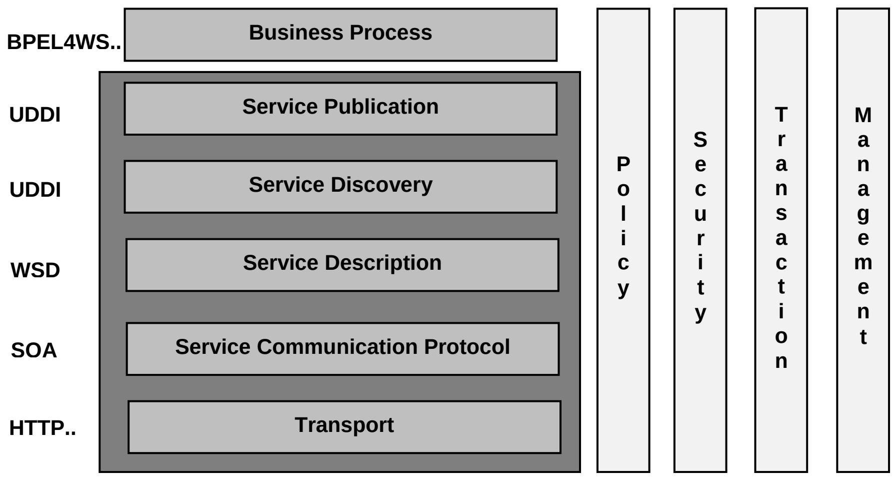


Figure 19. Basic Web service components and infrastructure capabilities (Gottschalket al. 2002, p. Van de Putte et al. 2004, p.2, modified).

It has been argued that Web services would offer a good technological foundation for integrating companies in a VE, because of the wide adoption of XML and HTTP, ease of integration and also support for partners search (Khoshafian 2002, p.11). As mentioned in this section, global UDDIs are to a great extend missing, thus limiting the partners search. As such, WSDL does not define the semantics of a Web service in such way that automatic service invocation and response handling would be possible. As discussed in section 3.1.3, an interface needs to be developed for each connection, leading to a point-to-point topology and tighter coupling. Moreover, a human programmer is always needed to interpret the XML-based description (Martin et al. 2007, p.244). For instance, Martin et al. (2007, p.245) have proposed accompanying Web services with semantic Web technologies to enable dynamic interoperability instead of "standard" interoperability of "basic" Web service technologies. The problem with Web service semantics can also be alleviated with, for instance, vertical standards. Further discussion of standards and semantic Web technologies follows in sections 3.2.5 and 3.4.

3.2.3 BUSINESS PROCESS MANAGEMENT SYSTEMS

Integration of different systems at the process level requires a technology solution that provides mechanisms for modeling the common processes and is capable of interacting with the systems that are involved in the process. Such solution entails multiple components, constituting a software suite that can generically be referred to as BPMS. These kinds of systems are provided by EAI and B2B vendors as a part to their integration solution portfolios (Samtani 2002, p.139). Additionally, there are also pure-play BPMS vendors that concentrate on this domain (Khoshafian 2002, p.9).

A typical BPMS solution includes a graphic modeling tool, a business process engine, a business process monitoring interface, a business process engine interface and an integration technology, such as an integration server (Linthicum 2004). The relationships of these components are presented in Figure 20. The graphic modeling tool provides means to design the business processes, by drawing process diagrams that represent the business logic and information movement between systems, while hiding the actual implementation (Samtani 2002, p.139). The heart of the system is the process engine, which controls the execution of processes, maintains process state and interacts with the underlying integration infrastructure (Linthicum 2004, p.57). The task list of the engine also includes keeping an audit trail of the executed processes (Khoshafian 2002, p.9). The monitoring interface allows users to track processes and take necessary optimization steps, without interrupting the process. The engine interface permits other applications to access the process engine. Finally, the integration technology provides the required connections to the underlying source and target systems. (Linthicum 2004, pp.57-58)

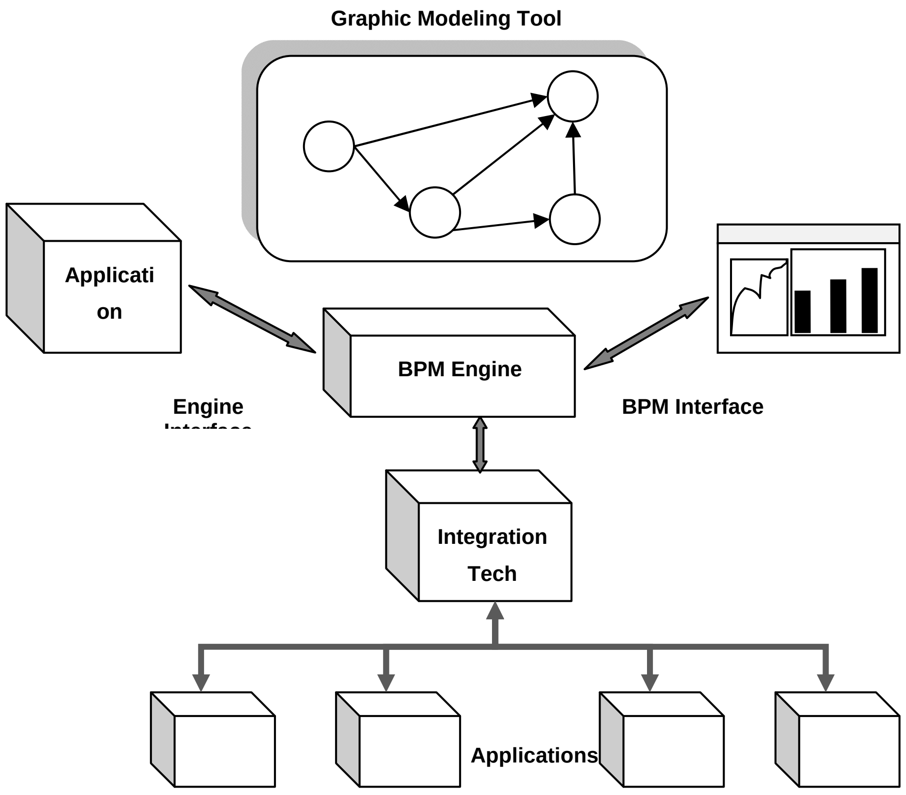


Figure 20. The components of a typical business process integration solution (Linthicum 2004, p.58).

Currently, many integration solution providers are promoting BPMSs as integral components to their integration platforms (Linthicum 2004, p.13; Samtani 2002, p.151). However, BPMSs have yet to gain the acceptance of market masses. According to Baeyens and Fricke (2006), this is a result of three things. Firstly, this lack of acceptance is due to market fragmentation and lack of common foundation to build on. Secondly, implementing BPMS systems tend to be expensive due to licensing and consulting fees. Thirdly, widely accepted standards, excluding BPEL, are missing from this domain. They identify open source foundation and decoupling integration middleware from BPMS as possible means of boosting the adoption of these systems. (Baeyens & Fricke 2006) BPMSs not being widely adopted and the lack of standards that support interchangeability of process models may also be seen as potentially inhibiting factors of VE process integration.

3.2.4 MIDDLEWARE AND INTEGRATION SERVERS

When reviewing the different integration approaches, it becomes quite apparent that middleware has a central role in B2Bi. Middleware provides connectivity to applications through adapters, data transformation capabilities, service request routing, support for different types of inter-application communication, and is sometimes associated with a BPMS. But what exactly is middleware? It is possible to generally define middleware as any type of software that facilitates and manages interaction between two or more heterogeneous software systems (Linthicum 2004, p.116; Alonso et al. 2004, p.29). Middleware comes of different types, including transaction processing monitors, message-oriented middleware, distributed components middleware, application servers and integration servers (Linthicum 2004, Samtani 2002). Some even argue that there is too much middleware (Stonebraker 2002). It would be impractical to dissect all middleware types in this section. Instead, a specific type of middleware, integration server, which provides an almost complete solution to the needs of B2Bi, is outlined here.

The fundamental goal of integration servers is to enable all types of integration within an enterprise and between trading partners (Samtani 2002, p.267). They are able extract the required information from the source system, transform it to a correct format and transfer it securely to the target system using proper communication mechanism. Integration server solutions are offered by multiple vendors, meaning that not all products provide the exact same functionality or share the same architecture. However, it is possible to define some general services these products provide and two major architectural approaches they build upon. Some features of integration servers have already been mentioned in section 3. Figure 21 provides a more complete view to the basic services typically provided by an integration server.

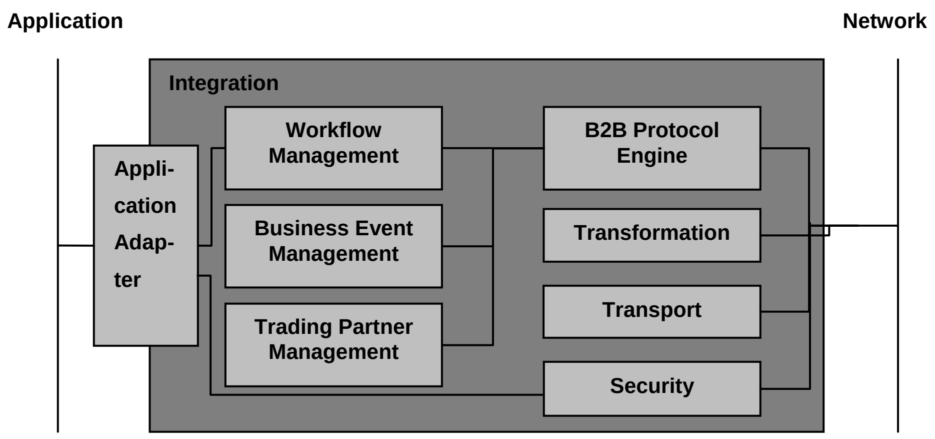


Figure 21. Integration server components (Bussler 2002b, p.3922).

As mentioned earlier in section 3.1.2, integration servers provide a set of ready-made adapters that enable communication between back-end systems and the integration server. Typical mechanisms used by adapters include sequential file access, invocation of component methods using APIs, RPCs and direct database access. The carrying idea is to provide adapters to as many leading enterprise applications as possible. Besides adapters, integration servers typically include different types of communication services, including capabilities for asynchronous, as well as synchronous, communication, message queuing, flow control, publish-and-subscribe and support for multiple security mechanisms. They should also support multitude of communication protocols, including FTP, HTTP, HTTPS, SMTP and TCP/IP. (Samtani 2002, pp.260-268) Additionally, integration servers often entail a novel way to route information, called intelligent or content-based routing. Intelligent routing allows integration servers to analyze the incoming messages, and based on their content and a set of predetermined rules, the integration servers are able to process the messages and route them to correct destinations. This facility builds on the data transformation component and rules processing capability of the integration server. (Linthicum 2004, p.203) What makes the routing task a challenging one, is that the information required for the routing decision may reside virtually anywhere in the message.

An integration server contains a data transformation component that allows information extracted from a source system to be translated on the fly to a format that is understandable to the target system. This involves both schema and data conversion. (Linthicum 2004, pp.198-202) Again, an integration server should understand and be able to convert multiple data formats, including standards based on XML and EDI. (Samtani 2002, p.260-268) Typically, integration servers include a workflow management component, which in a way binds the adapters, transformation and communication together, by allowing the dynamic modeling of integration between systems. One workflow step may include data extraction using an adapter, while another step may execute a local function, such as data transformation and the final step executes the sending of a message to a system hosted by a trading partner. (Bussler 2002b, p.3921) As suggested in section 3.1.4, current development may lead towards more complete BPMSs.

Integration servers also employ different types of repositories. The information that travels through the integration server may be stored in a message warehouse, which is a dedicated database. This message persistence facility is created in order to fulfill the needs of message mining, integrity, archiving and auditing. (Linthicum 2004, p.205) Another type of data store typically included in an integration server is a metadata repository, which contains information, such as communication protocol, message format, and security mechanism unique to a specific trading partner. Additionally, an administration tool with a graphical user interface should be included for controlling and monitoring the integration server behavior. (Samtani 2002, pp.266-270)

The two architectural approaches for integration servers are hub-and-spoke and network bus (Sanchez et al. 2001, cited in Erasala et al. 2003, p.79). In a hub-and-spoke architecture, the integration server acts as the hub, providing necessary integration services to the connected systems, that is, the spokes (Erasala et al. 2003, p.79). The management of this centralized solution should be relatively simple, as well as plugging new systems to the hub (Samtani 2002, p.256). However, in this type of architecture the integration server often represents a single point of failure and a possible bottleneck (Erasala et al. 2003, pp.79-80). These two issues can be circumvented with a clustered solution, in which multiple integration server instances run on different physical machines (Samtani 2002, p.257).

In the network bus model, adapters are installed at each system that is to be integrated. The bus backbone is then used for interacting with the integration server and other systems that are connected to the bus. (Sanchez et al. 2001, cited in Erasala et al. 2003, p.80) Rather than hub, the integration server should be regarded as an additional service on the bus (Samtani 2002, p.258). While being harder to manage as the number of integrated systems grows, this distributed model should alleviate the scalability concerns (Erasala et al. 2003, p.80).

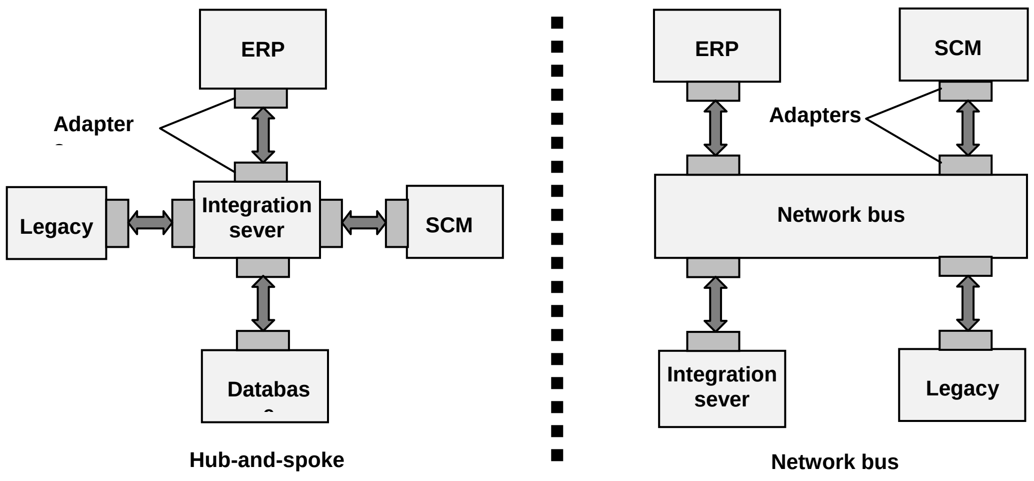


Figure 22. Hub-and-spoke and network bus integration server architectures (Samtani 2002, pp.257-258, modified).

All in all, integration servers provide “one-stop-shopping” for integration technology (Linthicum 2004, p.203). Integration servers allow companies to leverage their existing infrastructure, reduce in-house custom coding required for integration and, according to Gartner Group, speed up the development of interfaces between applications (Samtani 2002, p.285). Hence, they should also be valuable for a company’s preparedness to join a VE, regardless of its integration approach. However, they do also have a hefty price-tag attached to them (Samtani 2002, p.285). The price of an integration server implementation combined with the expertise required to leverage its capabilities force companies to think the return of investment of integration servers. SMEs and companies with lesser need for B2Bi may decide to purchase required integration server capabilities from a service provider that runs one, or settle for more light-weight middleware solution.

3.2.5 *Standards*

To overcome the interoperability issues in B2Bi, the adoption of standards takes an important role. Standards ease the B2Bi problem by providing a ready template, common terminology and structure for the information that needs to be shared among business partners (Preist et al. 2005, cited in Kotinurmi 2007, p.33), by reducing variety (David 1995, cited in Kotinurmi 2007, p.33) and by defining bindings to communication protocols, business process conversations and security mechanisms (Medjahed et al. 2003, p.60). Because multiple approaches to B2Bi exist, standards are needed on multiple layers, including information, service and process layers (Linthicum 2004, p.279). Moreover, standards-based B2Bi solutions are likely to be more scalable than solutions relying on proprietary links (Kotinurmi 2007, p.33).

Among the most discussed standards in the B2Bi domain are derivatives of Electronic Data Interchange (EDI) and XML. The fundamental idea behind EDI is secure electronic exchange of business transactions, including purchase orders, invoices, inquiries, planning and financial reporting (Samtani 2002, p.154), which are modeled in a standard digital format (Meadors 2005, p.83). Moreover, EDI is fundamentally based on inter-company agreements. Conventionally, the two most popular standards linked to EDI have been X12 and EDI for Administration, Commerce and Transport (EDIFACT) (Meadors 2005, p.83). Moreover, there has been a tendency to use value added networks (VANs) for data exchange between partners, rather than direct links. The costs associated to the use of VANs, in addition to the required software and hardware investments have limited the adoption of EDI (Samtani 2002, p.154), especially in case of smaller companies (Meadors 2005, p.83). However, the Internet era has also led to developments in EDI, namely, EDI over the Internet (EDI-INT). Fundamentally, EDI-INT aims to deliver secure business transactions over the Internet and is more of an option to the smaller companies as well (Meadors 2005, p.83). Besides enabling data exchange, in case of VEs, EDI has been considered as a driver that facilitates switching (Mowshowitz 2002, p.47).

Another result of the Internet era is the diffusion of XML. During chapter 3, it has become apparent that XML has a role in different B2Bi approaches from information-oriented to business process –oriented integration and in integration components, such as Web services. XML is essentially a data definition language (Samtani 2002, p.161), which can be used to define the format of data exchanged in B2Bi (Kotinurmi 2007, p.18). In B2Bi context, the strengths of XML are that it is founded on flexible open standards and it promotes loose integration of components (Samtani 2002, p.161). However, if two applications produce data in XML format, there are still no guarantees that these two applications understand each other (Kotinurmi 2007, p.18). Therefore, the business parties that engage themselves to B2Bi need to explicitly agree on data semantics or comply with an XML-based standard, such as RosettaNet, or ebXML, which defines the necessary document formats, information allowed in a document and includes descriptions of processes (Samtani 2002, p.188). Of these two, RosettaNet is a vertical standard originally aimed at the electronic component and IT industry (Samtani 2002, p.188), whereas ebXML is a horizontal standard that aims for global use of e-business information (Kotinurmi 2007, p.25).

Fundamentally, RosettaNet aims to align the business processes of two or more partners of a supply chain. It builds upon Partner Interface Processes (PIPs), dictionaries and RosettaNet Implementation Framework (RNIF) (RosettaNet 2008). PIPs define certain intercompany public processes, including “Request Quote“ (PIP 3A1 in Figure 23) and “Request Purchase Order” (PIP 3A4 in Figure 23) and the business documents associated to these. The internal private processes of trading partners then interact with PIPs to either initiate or receive business documents (see Figure 23). (Kotinurmi 2007, p.25) RNIF includes specification of packaging, routing and transport of PIP business messages and signals. RosettaNet Business Dictionary specifies the properties used in basic business activities and RosettaNet Technical dictionaries include properties for defining products. (RosettaNet 2008) RosettaNet, and other standards of similar type, is an interesting standard from the perspective of VEs, since it aims to standardize and interface specific business processes, rather than just business information. This kind of standardization is likely to increase the level of readiness of companies to build and join VEs in certain industries or industry clusters.

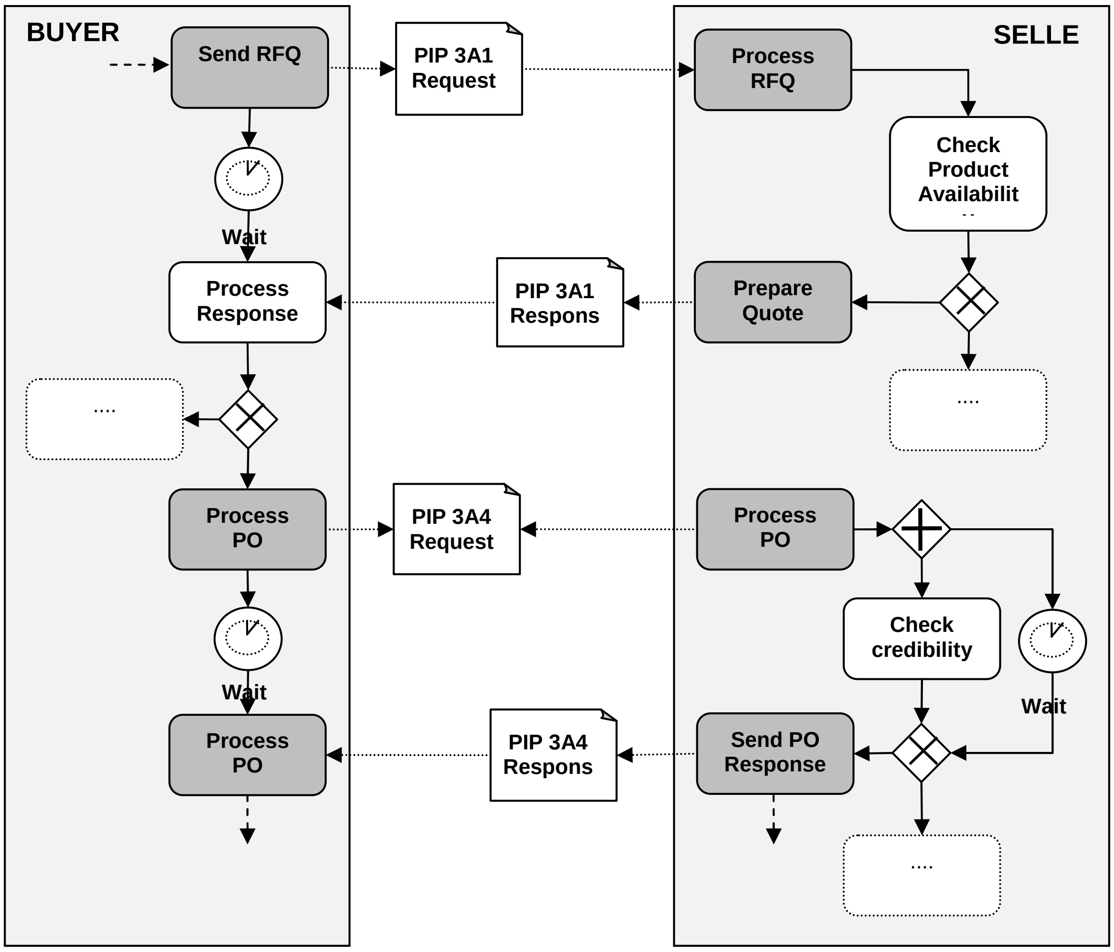


Figure 23. Rosettanet B2B collaboration. White boxes are used to denote internal computational steps, dotted boxes placeholders for possible many omitted internal computational steps and grey boxes represent public behavior according to the PIPs. (Haller et al. 2007, p.1369)

In chapter 2, it was identified that VEs call for standardization. Standardization is seen as an enabler of switching, because it lowers the transaction costs associated to coupling and decoupling production processes (Mowshowitz 2002, p.119). However, adhering to same standard does not alone guarantee proper interoperability between two partners, as standards often offer room for manoeuvre (Preist et al. 2005, p.988). For instance, each message schema within a PIP is defined in RosettaNet, but a number of elements within a given schema are left optional. This leads to a situation, where a buyer may need to interpret new information sent by a new seller. (Haller et al. 2007, p.1369) Moreover, standards tend to be as good as the number of people or companies that leverage them (Linthicum 2004, p.291).

3.2.6 Security

When partially cooperative enterprises engage to B2Bi and electronically share sensitive information, such as product and customer data, as well as automate inter-enterprise business processes, including orders and invoicing, it is quite obvious that security of the business-critical information takes a central role. Leakage of confidential information could have disastrous consequences on the business relationships (Samtani 2002, p.288). The security measures that need to be put into place largely depend on the approach to B2Bi. For example, a POI solution with a Web front-end needs to address different security issues than a SOI solution, which typically involves exchanging SOAP-messages over the Internet. Portal applications are protected by security services, including user profile management, user identify verification, back-end application access control and access policy enforcement (Chandran et al. 2003, p.391). Generally, integration approaches that involve business message exchange over the Internet need to address security concerns, including: (Samtani 2002, p.292; Rosado et al. 2006, pp.521-522)

- ***Authentication*****:** Verifying the identity claimed by the message sender.
- ***Authorization*****:** Determining the set of messages an authenticated party is allowed to send.
- ***Integrity*****:** Ensuring that a message was not altered during transmission - either accidentally or purposely.
- ***Confidentiality:*** Assuring that the privacy of a message prevails during transmission.
- ***Auditing:*** Keeping a record of messages so that problems can be analyzed later on.
- ***Non-repudiation:*** Enabling interacting parties to show legal proof to a third party that the sender did send the message and the recipient received the identical message.
- ***Availability:*** Guaranteeing that critical resources are available at all times.

Of course, the communication end points need to be secured as well - be they back-end applications, databases or integration servers. The amount of technologies that can be used to satisfy the B2Bi security needs is vast and not all can be dissected in this section. Instead, some of the technologies that have gained particular interest in this domain are presented here. These include encryption, digital signatures and certificates, secure socket layer (SSL) and firewalls. Encryption ensures the privacy of messages, by turning them from readable format to non-readable. (Fitzgerald 2001, pp.113-114) The encryption process is done with an encryption algorithm and using private and public keys. In private key encryption, the same unique code, that is, the private key is used both by the sender for encrypting and by the receiver for decrypting messages. In public key encryption, the message sender encrypts the message using a public key, and the message can only be decrypted using the recipient's private key. (Samtani 2002, pp.293-205)

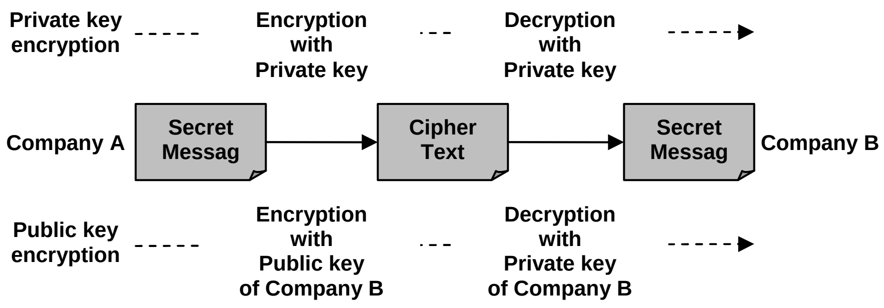

Private key encryption
Encryption with Private key
Decryption with Private key
Company A
Secret Messag
Cipher Text
Secret Messag
Company B
Public key encryption
Encryption with Public key of Company B
Decryption with Private key of Company B

Figure 24. Private and public key encryption (Samtani 2002, pp.294-295, modified).

Public key systems provide the foundation for digital signatures, which are used for authenticating the message sender. Digital signatures can be constructed using a hash function that produces a message digest. A private key is then applied to the digest, resulting a digital signature. The digital signature can then be verified by the recipient using its public key. The question of how to acquire a public key of a specific party is answered by certificate authorities (CAs) that issue digital certificates. A company sends required information, including a public key to the CA, which then verifies the authenticity of the information and then issues a certificate. The information of certificates is used by, for example SSL, which operates on top of the TCP/IP network layer and provides privacy, integrity authentication to protocols, including HTTP, SMTP and FTP. Thus, SSL provides security between communicating systems. (Samtani 2002, pp.296-301)

A security solution is not complete without the added security of a firewall. Firewalls protect a company's internal networks by enforcing an access control policy to and from the network. They can be implemented by both software and hardware. A novel way to leverage firewalls is to put in place a demilitarized zone (DMZ). A DMZ typically includes an external firewall shielding the systems inside the DMZ from external intruders and an internal firewall denying unauthorized access to the company's internal networks from the DMZ. (Samtani 2002, pp.303-312) On the negative side, firewalls do block several communication channels that could be used for interaction between systems and thus represent a special challenge to B2Bi. Some of the issues caused by firewalls can be circumvent through tunneling solutions. (Alonso et al. 2004, pp.114-115)

Security claims a role in trust-building between trading partners, for instance, by ensuring reliable business transactions and data transmission (Ratnasingam & Phan 2003, p.41). Trust is essential to VEs, as discussed in section 2.4.1.1, since VEs typically involve cooperation of parties that are previously unfamiliar with each other. It is quite evident that a failure to conform to security requirements set by the partnering companies may compromise a B2Bi project altogether. On the other hand, configuring security solutions, like opening firewalls, is time consuming. Companies have often developed security policies and procedures that need to be followed when changes are made to the security solutions. VEs require flexibility from these security solutions, policies and procedures, both in formation and reconfiguration, since changes should be made to the VE without incurring excessive transaction costs, including time.

3.3 RISKS IN BUSINESS-TO-BUSINESS INTEGRATION

As discussed earlier in this chapter, multiple different technologies, standards and patterns exist for the B2Bi problem domain. This variety basically means that unless an integration project is thoroughly planned as well as systematically managed and executed, it may become a daunting and risky process. The first thing where things can go wrong is the understanding of the problem domain and the end-to-end inter-organizational business processes (Linthicum 2004, p.370; Lam & Shankararaman 2004, p.41). Failure to understand the complete processes, including how companies handle exceptions, may lead to an integration solution that does not satisfy the needs of the business. To avoid this, all relevant stakeholders of a business process need to be included in the planning. Additionally, the parties involved in a process should clearly agree on the used business semantics. (Lam & Shankararaman 2004, pp.42-43. For example, having not defined the meaning of data on a sufficient level of detail can lead into shipment of wrong items, wrong amount of items or correct items on a wrong date. Moreover, ordering 100000 CPUs on a given date is futile, if the vendor can only supply 10000.

Thorough planning also reveals deficiencies in the systems that are involved in the integration project. In the worst case, these may turn out to be “integration showstoppers”. A key thing is to realize that ICT systems exist to benefit business processes, not the other way around. Hence, a generic guideline is that business processes should not be changed in order to alleviate shortcomings of ICT systems. Planning, including the mapping of business process parts to integration components, uncovers the need to add new system functionality in order to satisfy the process requirements or even adding manual steps to the process. (Lam & Shankararaman 2004, pp.42-43)

Another thing that can make or break an integration project is testing. Overlooking the necessity of testing may result in mishandling of data, including loss of important information or wrong information appearing into applications (Linthicum 2004, p.388). Thus, testing should be a major concern for VEs as well, both during VE formation and reconfiguration. Successful testing process typically involves setting up proper development and test environments within each enterprise, as well as generating test data that resembles real-life data. Furthermore, stress testing should be conducted in order to evaluate system performance and reveal possible bottlenecks. (Lam & Shankararaman 2004, pp.47-48) If timeliness of data is critical to a business, it is clear that the integration solution should deliver consistent response times even under increasing load. This requires designing for performance and scalability from the components of an integration solution. (Linthicum 2004, p.388) Scalability could turn out to be an issue for VEs, if the network grows during operation phase, due to reconfiguration. Additionally, newly added partners' systems may represent possible bottlenecks.

Once the integration solution is properly tested, the next step involves taking it to production. Besides meticulously planning the cut over, the companies should have a rollback plan in case the cut over fails (Lam & Shankararaman 2004, p.48). Lastly, the companies need to have people assigned for maintenance tasks, such as security management, system performance monitoring and problem-solving. Deviations from normal system behavior should be dealt with immediately, as B2Bi involves exchanging crucial business information between mission-critical systems. (Linthicum 2004, p.392). A failure in cut over or neglecting system monitoring is likely to have negative business impact due to, for instance, occurred delays.

The role of planning in B2Bi projects generally cannot be overemphasized. Without proper planning, the implementation of an integration project may become chaotic, with new work items emerging and project completion dates being delayed (Lam & Shankararaman 2004, p.48). Of course, delays have a direct impact on project costs. VEs may be more intolerant to delays since they seek rapid formation and minimized transaction costs. Whether VEs are also more prone to the aforementioned risks is likely to depend on multiple factors, such as, capabilities of individual companies, complexity of both the VE and its business processes and the desired integration solution.

3.4 BUSINESS-TO-BUSINESS INTEGRATION TRENDS AND EMERGING COMPONENTS

In B2Bi domain, adoption and diffusion of new integration approaches or components typically takes time. At times, a new standard or technology is not adopted, until it is demanded by a key business partner. The slow rate of adoption of new technologies is due to issues, including their invasiveness, uncertainty revolving around them and costs associated with their adoption. In this context, invasiveness means changes induced to the existing ICT infrastructure upon the adoption of new technology. Of course, new approaches and components are adopted, if they are valuable to the business. Next, some of the most relevant trends and new components of B2Bi are discussed.

At present, implementations of SOA represent the state-of-the-art in the integration domain (Keen et al. 2004) and the service-oriented approach in general is gaining ground (Linthicum 2004; Rosenberg 2005). According to AMR Research report, IBM alone has approximately 5700 ongoing SOA projects worldwide (IBM 2008). According to Linthicum (2004, p.4), there is a general movement from IOI to SOI. This trend is also interesting from the perspective of VEs. SOI approach promotes loose coupling between components that need to be integrated, standard interfaces, keeping a registry of services and composing services into business processes that span over multiple companies. The aforementioned features should be realizable through open standards and at a relatively low cost. These promises seem to conform very well to the requirements set by VEs. However, the obstacles discussed in sections 3.1.3 and 3.2.2 ensure that SOI approach does not provide an “out-of-the-box” –solution for VE integration.

Another ongoing trend in the B2B domain is convergence. Firstly, as discussed in section 3.2.1, the borders between B2Bi and EAI are becoming blurred, since integration product vendors are including capabilities to their products that are suitable for both domains. Additionally, solutions that build upon SOA are further blurring these borders, since services may reside within a single company or outside the company’s borders. This type of convergence potentially results in designing for integration, rather than designing for internal or external integration. This, in turn, may result in better availability of information, and therefore speed up integration. Secondly, according to Forrester Research Inc, also SOA and BPM are converging (Jax Magazine 2007). From another perspective, the business process –oriented integration is seen as “the ultimate destination of application integration. (Linthicum 2004, p.13)” Hence, the convergence can be seen more of a natural movement. Anyhow, business process -oriented integration remains a hot topic in the B2Bi domain.

Besides the aforementioned trends, there are also several emerging B2Bi components that may gain importance in the near future. In this section, two emerging components, namely semantic Web technologies and software agents, are introduced, since they seem to be particularly interesting for VE integration. In addition to these, there are several new emerging standards, such as, Universal Business Language (UBL) (see UBL 2008; TIEKE 2008) and new technologies to this domain, such as, grid computing (see Foster et al. 2002; Hao & Hao 2007) and P2P technology (see Samtani 2002; Putnik et al. 2005) that will potentially enable new approaches to B2Bi. However, these are not further discussed in this thesis.

B2Bi standards often define meanings for data, but leave room for variation, as discussed in section 3.2.5. For instance, EDIFACT and RosettaNet PIP messages contain many optional elements that can be used differently by separate companies. This opens new possibilities for semantic Web technologies in the B2Bi domain. The semantic Web technologies, including the Resource Description framework (RDF), Resource Description framework (RDF) and Web Ontology Language (OWL), aim to solve the interoperability issues of current Internet technologies (Kotinurmi 2007, p.20). Given this goal, it seems quite natural that semantic Web technologies are being paired with Web services. The idea behind semantic Web services is to place a semantic layer on top of Web services and to extend Web services with explicit representation of meanings (Kotinurmi 2007, p.21). In ideal situation, the use of these technologies would result in higher degree of automation and significantly reduced integration times. Moreover, semantically describing the services a company is offering may potentially ease service discovery and partners selection. (Preist et al. 2005, p.988) Of course, this requires that such registry exists, where the service lookup may occur. Due to the aforementioned features, the value proposition of semantic Web technologies to VE integration is clear (see Zhonghua Yang et al. 2006). However, semantic Web technologies are not yet used in B2Bi, and standards, such as RosettaNet represent the best effort of domain-specific ontologies (Kotinurmi 2007, p.85).

The use of software agents in B2Bi is currently being evaluated. Software agents are basically software components that are programmed to sense the environment they are in, act upon it and adapt to it (Samtani 2002, p.382).

The agents are not designed to operate in isolation (Samtani 2002, p.382), but entail capabilities for interaction, communication and decision making (Shen et al. 2007, p.317). The vision of agents is that they will be able to search and interpret data of vendors and products, make transactions and generally automate coordination activities between business partners (Samtani 2002, p393). The use of agents is also considered in VE context, since a VE is typically formed of partially cooperative, distributed, heterogeneous and autonomous entities, and therefore relatively easily mapped into MAS (Camarinha-Matos & Afsarmanesh 2003, p.343). For now, real-life agent-based B2Bi is non-existent, and only prototypes have been developed in research labs using proprietary agent technologies (Shen et al. 2007, p.317). When the B2Bi community is able to mitigate the complexity of solving traditional B2Bi problems, such advanced technologies may begin to gain ground.

3.5 Integration options and possible constraints for virtual enterprises

In this section, the IOI, SOI, BPOI and POI approaches to B2Bi and the essential B2Bi components have been introduced and discussed in VE context. It has been argued that solutions can be built using the aforementioned approaches and components to conform to the VE requirements for safe transactions, application integration and business process integration. Additionally, consideration has been put on how these conform to the requirements of full VE life cycle support and of support for VE reconfiguration dynamics. The use of Web service technologies, namely UDDI, has been considered as an enabler of automated partners search and selection. However, it has been noted that UDDI registries are restricted to private ones. Still, a service registry could be created to store, for instance, services available to a given VBE. Agent technologies have also been considered for the partners search, but their practical application is still years away in the B2Bi domain.

Reconfiguration dynamics is a characteristic of VEs that poses considerable challenges to B2Bi. These challenges are primarily a result of heterogeneity of business partners that seek B2Bi. Current technologies, including Web services, quite effectively enable flexible integration across diverse platforms. However, differences in application semantics inhibit efforts for integration automation and dynamic integration. The technologies of semantic Web have been considered to solve the issues that arise from semantic heterogeneity. At present, these technologies have not been adopted by companies or trading communities. Hence, the sight is on B2Bi standards that limit variety by providing a ready template, common terminology and structure for the information that needs to be shared among business partners. Some of these standards, such as RosettaNet, also bring value to the integration of inter-company business processes and lead the way for process-oriented solutions, including BPMSs.

It is not that VE reconfiguration dynamics could not be realized through current B2Bi approaches and components. Several middleware solutions, including integration servers, are designed to bridge the semantic gap that exists between companies by providing capabilities for data transformation. However, if differing semantics needs to be tackled case by case when a VE is reconfigured, the transaction costs are likely to rise considerably. Conformance to standards, for instance, within an industry cluster may bring companies semantically closer to each other and thus act as an enabler of dynamic reconfiguration. Hence, one of the key arguments of this thesis is that preparedness of partners is essential to the cheap formation and dynamic reconfiguration of a VE.

4. INTELLIGENT ROUTING SOLUTION TO TRADEXPRESS

One strategy to B2Bi is that a company makes a single connection to a B2B integrator, denoted here as integration service provider (ISP), and lets it “take care of all the complex, behind-the-scenes work necessary to bridge the electronic gap between its partners, vendors and customers quickly and painlessly. (Samtani 2002, p.19)” The fundamental idea is that the customers of an ISP can build on the capabilities of their existing systems without any or with little changes to these systems. Yet, the integration services offered by an ISP make it possible for these companies to sign up on virtually any service that is available electronically.

From the opposite perspective, an ISP faces the formidable task of managing a multitude of connections between its customers and their business partners on the integration server it is running. These connections are often message-based and represent the full spectrum of standards and message types. In this thesis, the integration server of interest is TradeXpress (TX). This chapter includes presenting the design of an intelligent routing solution (IRS) to TX. The IRS addresses an ISP’s need for content-based routing and effectively setting up and managing the message-based connections on TX. Additionally, consideration is put in this chapter on how a company’s preparedness to join a VE is affected in case it decides to leverage the services of an ISP running an integration server as its B2Bi strategy.

4.1 THE NEED AND OBJECTIVES FOR THE INTELLIGENT ROUTING SOLUTION

A fundamental requirement for an integration server is to route incoming messages to correct target systems (recipients), transforming them if required. Incoming messages in this context are messages that are either sent to the integration server or fetched by it. In TX, one possibility to conduct the necessary routing operations for incoming messages is to build static routes. In static routes, messages propagate predetermined paths on the integration server. For instance, all the data sent by one source system is transformed to a fixed format and sent to a fixed target system. When a source system transmits messages intended for multiple recipients or differing messages to a single recipient, additional routing logic is generally needed.

Another alternative is to use dynamic routing. In dynamic routing, routing rules are created into the integration server and based on these rules the integration server makes a routing decision. In TX, these routing rules can be based, for example, on file naming conventions or pre-built content-based routing capabilities of TX routing module (see section 4.3.2). However, there are numerous situations, where messages cannot be routed according to the file names. Additionally, this approach requires that the source system supports a file naming convention and it also restricts the data communication protocols that can be used for the data transfer. As stated previously in section 3.2.4, content-based routing allows integration servers to analyze the incoming messages, and make routing decisions based on their content. TX has pre-built content-based routing capabilities for commonly used standards, such as, EDIFACT, ANSI X.12, VDA and XML. However, the pre-built EDIFACT routing capability for instance is restricted to information in UNG-, UNB- and UNH-segments. If the routing decision needs to be done according to data in other segments in the message, custom programming is needed. There are also limitations in content-based routing of other standards-based messages and content-based routing of messages in any proprietary format always requires custom coding.

As discussed above, an ISP faces a challenging task of effectively configuring as well as managing the message-based connections and routings between its customers and their business partners on the integration server it is running. The challenges result primarily from the amount of connections, as well as the diversity of systems of the parties involved. The diversity of systems requires that a large variety of communication mechanisms and message formats, both standard as well as proprietary, are supported by the integration server. Integration servers are typically designed to cope with this kind of situation and offer scalability, secured transactions and standards compliance (Samtani 2002, p.267). TradeXpress makes no exception to this. However, the issues listed above may lead to a situation, where an ISP has a complex web of dispersed routings that are created in ad hoc -fashion. Instead of using a mixture different routing approaches to separately configure each connection, there is a need for a generic mechanism, which is able to dynamically route all the incoming messages in TX, no matter what the file names, message formats and message types are. The need is most obvious for ISPs serving multiple SMEs on a single TX user environment. However, the need may also exist for user environments created to serve companies with considerable number of partners and multiple internal systems or business units. Basically, this mechanism must be founded on content-based routing. To the aforementioned need, the IRS strives to deliver a solution. It also aims to create more flexible message-based interfaces between ISP's customers and their partners.

The objective of IRS is that customers of an ISP, or their designated partners, can send files containing a number of messages in any given format to TX server, using any communication mechanism supported by TX, and the IRS routes them to the correct target systems based on their content. The routing procedure may include processing, such as, message transformation, or messages may only need to be routed to the correct recipients unprocessed. Given the amount of all possible message formats, the IRS is initially designed to support three common standards-based message formats, EDIFACT, OVT and XML, and additionally, all kinds of text-based proprietary message formats. The idea is that the IRS may be configured to route any type of message that conforms to one of the formats listed above. After the initial setup, the IRS must support adding new functionality. For instance, new message formats, types, processing capabilities and recipients may need to be added to the IRS, as the businesses of ISP's customers evolve or as the ISP acquires new customers. The IRS concept is illustrated in Figure 25 below.

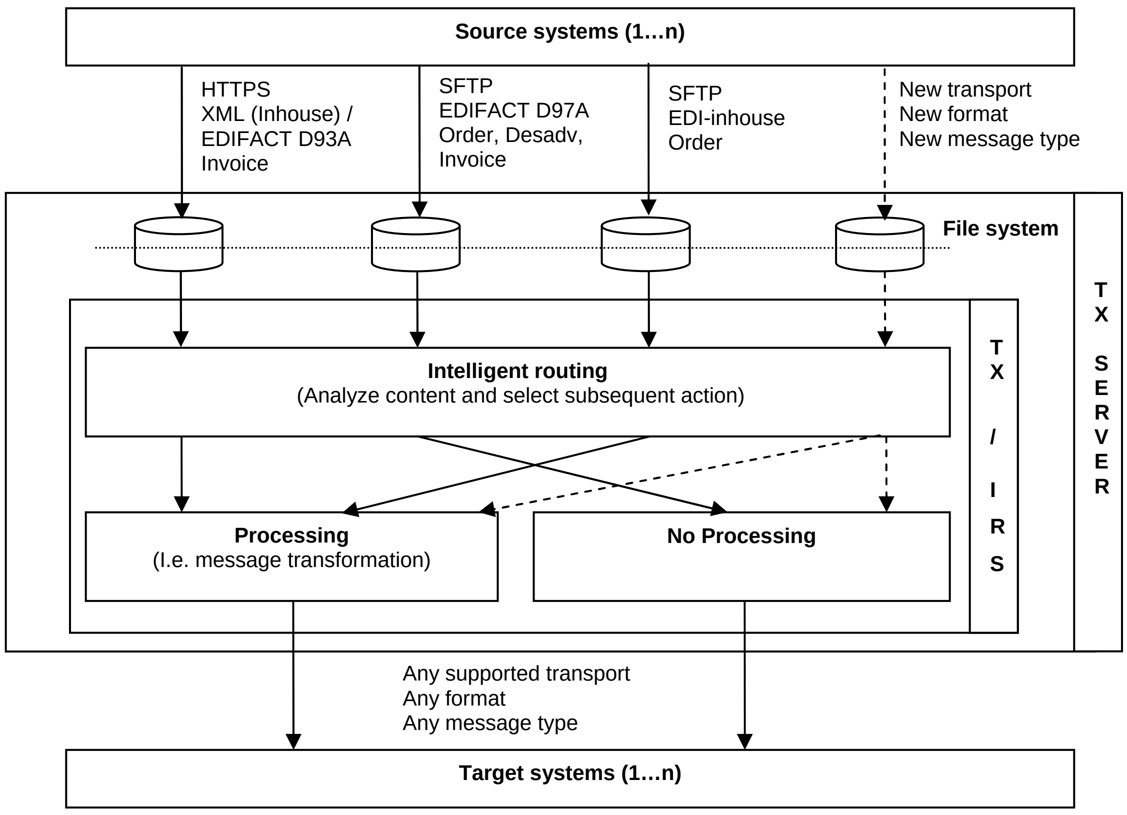


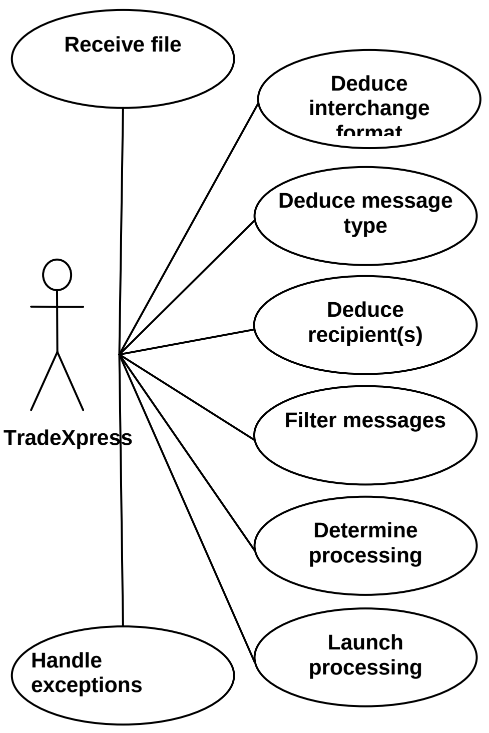
Figure 25. The concept of intelligent routing solution.

4.2 DEFINING THE INTELLIGENT ROUTING SOLUTION

In order to dynamically route messages based on their content, the integration server must understand the format of the incoming messages. Once the message format is known, it is possible to deduce the information needed to determine proper processing for the messages and the correct target systems. This section includes defining the capabilities of the IRS, that is, what the system will and will not do. Additionally, the sequence of actions required to achieve the desired outcome is identified. Hence, the focus is kept in the problem space before moving into the solution domain.

4.2.1 SYSTEM CAPABILITIES

To gain better understanding of what is to be built, it is beneficial to view things at a high level of abstraction, thus hiding the details of the actual implementation. This includes describing the system's capabilities, that is, what the system will and will not do. UML use case diagrams are particularly useful in visualizing the capabilities of a system and they provide a convenient way of prioritizing them later on (Kimmel 2005, p.19) The capabilities of IRS are modeled with two separate use case diagrams, displayed in Figure 26A and Figure 26B, respectively. Figure 26A represents a use case where the IRS conducts the activities required to route messages sent by its clients' and their designated partner's applications. Figure 26B portrays a use case of adding new capability to the IRS. Together these two use cases conform to the objectives presented in section 4.1



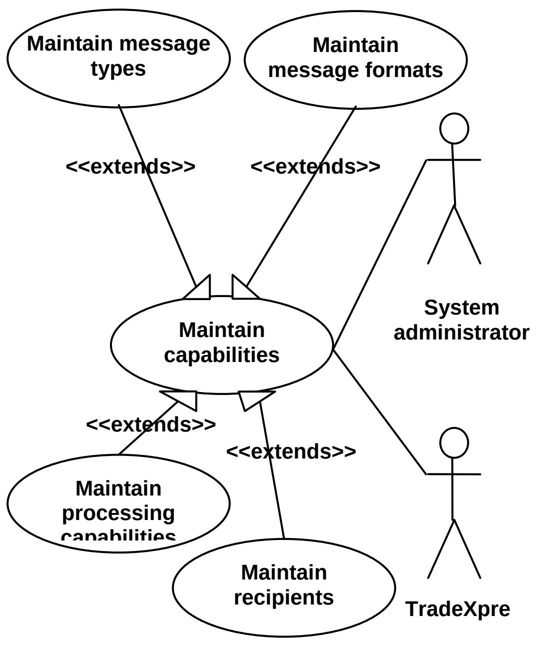


Figure 26A. Use case of routing messages. Figure 18B. Use case of maintaining capabilities.

The first use case (

Figure 26A) starts when a message sent by a source application or fetched by TX is stored to the file system of TX server. The sole <<system>> actor is TX. In order to route messages, the IRS must:

- ***Receive file:*** The IRS needs to be able to receive files for processing. It expects that the messages are stored to the file system of TX server and made available for processing. A file contains an interchange, which in turn may contain several messages, for example, multiple invoice messages for 1 to n recipients. An interchange may not contain messages in multiple formats, for example, both XML and EDIFACT. Neither may it contain multiple types of messages, for example, orders and invoices. Additionally, multiple interchanges in one file are not initially supported. Moreover, the files must be in text format, not binary.

- ***Deduce interchange format:*** A key capability of the IRS is deducing the interchange format. Data representation differs considerably between, for example, XML and EDIFACT. When the format is known, it is possible to extract the information required to select correct processing for messages.
- ***Deduce message type:*** Message type needs to be determined by the IRS in order to select correct processing. Different processing is conducted to messages containing orders than to ones containing dispatch advices.
- ***Deduce message recipient(s):*** A fundamental requirement for the incoming messages is that they must contain a recipient identifier. Additionally, this identifier must be both unique and unambiguous, because it is used to determine the target system the message is sent to. The IRS must extract a recipient identifier from each message and check the validity of each recipient. The identifier can be, for instance in Finland, a business identity code assigned by a public authority or an OVT (Organisaatioiden Välinen Tiedonsiirto) code. Other mechanisms to determine the message recipient can be created later on.
- ***Filter messages:*** An interchange consisting of multiple messages to a number of different recipients needs to be split in separate messages or in recipient-specific interchanges in order to process them correctly. Initially, the IRS is designed to filter messages into files that contain only one message per file.
- ***Determine processing(s):*** Based on the message format, message type and recipient information, the IRS makes a decision of the subsequent processing for each filtered message. The processing may include data transformation, for example, from proprietary format to EDIFACT format and sending message to the target system, or just sending the message. For instance, the message may be routed without processing in case, when the originating system produces data in a representation the target system can understand, but is unable to send messages using the right communication mechanism or the right communication media. In TX, the processing is linked to the sending of messages to the correct systems (see section 4.3.3)
- ***Launch processing:*** Based on the determined processing, the IRS starts the processing for each messages included in the received file.
- ***Handle exceptions:*** In order to be a credible solution, the IRS must handle exceptions as they arise. Exceptions may occur at different stages of the routing procedure. Exceptions and their handling are discussed further in sections 4.2.2 and 4.6.2.

Above, the primary capabilities of the IRS were presented. The IRS is not concerned on what kind of processing is conducted to the messages. The processing is to a great extent connection-specific. For instance, a company may use different message representation for the same information to communicate with different partners, and it is not generally possible to use the same application to transform these messages. Hence, the IRS routes incoming messages to the correct processing and expects that the processing handles the transformation and sending activities. However, the processing is still considered as a part of the system. Additionally, the solution is not concerned on how the communication links between an ISP and its customers as well as between the ISP and its customers' partners are implemented. More precisely, what communication protocols are used, how the connections are configured and what security mechanisms are put in place, is beyond the scope of the IRS. The IRS merely expects that the messages are stored to the file system of TX server and made available for processing.

The other use case presented in Figure 26B involves maintaining the IRS. A new customer to an ISP may require that a new message format is supported by the IRS or an existing customer may want to send new message types to existing or new partners. In the maintenance use case, there are two actors, TX and a system administrator. In order to be maintainable, the IRS must support:

- ***Maintaining message formats:*** It is not feasible to try to support at once every message format that exists, because there simply are so many of them. Initially, the IRS is designed to recognize XML, EDIFACT, OVT and proprietary text-based message formats. Support for new message formats is added as a need arises, for example, when a new customer starts using services of an ISP.
- ***Maintaining message types:*** New message types need to be recognized by the IRS as the trading relationships between ISP's customers and their partners evolve. Companies may first automate order processes and later add invoicing, logistics processes, etcetera. What message types are initially configured to the IRS, depends entirely on the needs of the customers of an ISP.
- ***Maintaining recipients:*** Adding new recipients for the messages routed by the IRS needs to be supported. New recipients are added, for example, when companies start doing business with new partners.
- ***Maintaining processing capabilities:*** New processing capabilities, for example message transformations, need to be added when new customers start sending messages, or when new message formats, message types and recipients are introduced to the IRS.

4.2.2 SYSTEM ACTIVITY

Before going into greater detail of how the system is to be implemented in TX, it must be considered how the system should work on a conceptual level. The focal tasks of the IRS, that is, recognizing the message format, type, recipients and filtering the interchange into individual messages need to be carried out in a specific order, so that correct processing can be selected and launched. The operation logic of the IRS is depicted in Figure 27 below.

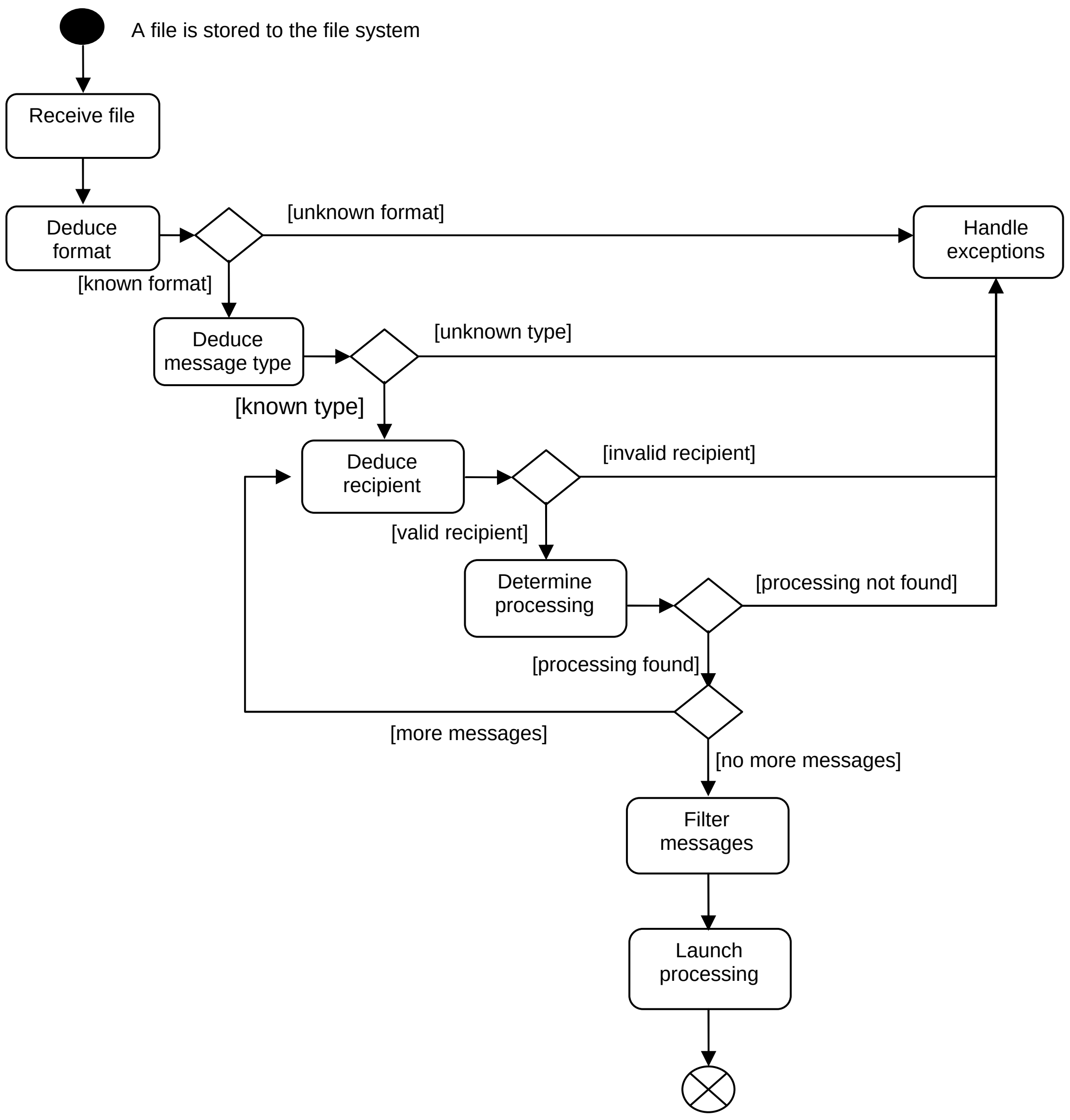


Figure 27. Activity diagram depicting the operation logic of the intelligent routing solution.

First, the files sent to TX or fetched by it need to be retrieved by the IRS. These files may reside virtually anywhere in the file system of the TX server. Hence, the IRS must be able to scan specific directories for incoming files. After the files are retrieved, the IRS needs to deduce the interchange format contained by the retrieved files. In case the format cannot be deduced, proper exception handling procedure needs to be launched. Once the format is known, it is possible to deduce the information inside the messages that is needed to route messages to correct target systems. The first required information is the message type. The IRS must deduce the message format and type to determine the data representation. For instance, if the format is identified as XML and the message type as a custom invoice, the system knows that the message conforms to the custom invoice schema. If the IRS detects an unknown message type, it must start an exception handling procedure.

As the message representation is known by the IRS, it is able to deduct any information, residing anywhere in the interchange. Message recipient information is deduced to select the correct target system for each message contained in a file. The validity of a recipient needs to be checked by the IRS to ensure that a target system configuration exists. In case the recipient is not valid, an exception handling procedure is started. Based on message recipient information, message type and message format, the IRS determines the subsequent processing. If no processing is found, the IRS starts the exception handling procedure. The tasks of deducing recipient information and determining processing are carried out as many times as there are messages in a file. After correct processing is determined for all the messages in a file, the next action is to filter the messages into individual messages. As the aforementioned tasks are all successfully conducted, the IRS starts processing for all the filtered files. The processing then handles the message transformation and sending to the target system, as specified in section 4.2.1.

4.3 BASIC SYSTEM COMPONENTS

TX is an extensive integration server software package. The software is available on multiple platforms, and its most recent major release is version 5.0. However, the intelligent routing solution is designed for the penultimate version 4.4.1, due to its currently wider adoption. However, the software has evolved relatively conservatively, thus migrating the solution to the most recent version should not entail major barriers.

The architecture of TX fundamentally comprises of well-defined user environments, application, as well as data communication interfaces, internal databases, Reliable Transaction Engine (RTE) programming language, event logs and Scanner process for orchestrating TX activity. TX also entails a graphical user interface (UI) for configuring TX user environments and data flows as well as monitoring them. (Influe-Illicom 2007, pp.8-10) Next, the components of TX that form the foundation for the IRS, namely partner management, routing module, edisend module and RTE, are discussed in more detail.

4.3.1 TRADEXPRESS PARTNER MANAGEMENT

The internal database system of TX in version 4.4.1 is proprietary and includes for each user environment a System log (Syslog) database for keeping track of events, a Partner database and two connection databases, namely Edisend and Routing. Additionally, it is possible to build custom databases for various needs. TX databases have a command line user interface and a RTE-based programming interface for different types of administrative purposes. (Illicom 2004a, pp.32-37) Every TX database is also accessible through the graphical UI of TX.

Extracting recipient identifiers from messages and inspecting their validity was mentioned as a core capability of the IRS in section 4.2.1. The information of the parties exchanging messages using TX is by default stored in the dedicated Partner database. It is possible that one physical party, for instance a business entity, has multiple partner profiles on TX, that is, entries in the Partner database. Partner profiles contain a partner identifier, transport specific information, such as transport mechanism, as well as a network address, message syntax used in communication and generic information about the partner (see Figure 28)(Influe-Illicom 2007, p.22). Additionally, each Partner database entry is linked to an info file. The info file can be used to store supplementary free-form information in addition to the predefined information described above. For IRS, the info file offers a way to store data about the processings specific to a certain partner. As a result, the Partner database enables IRS to store all partner metadata in one centralized location.

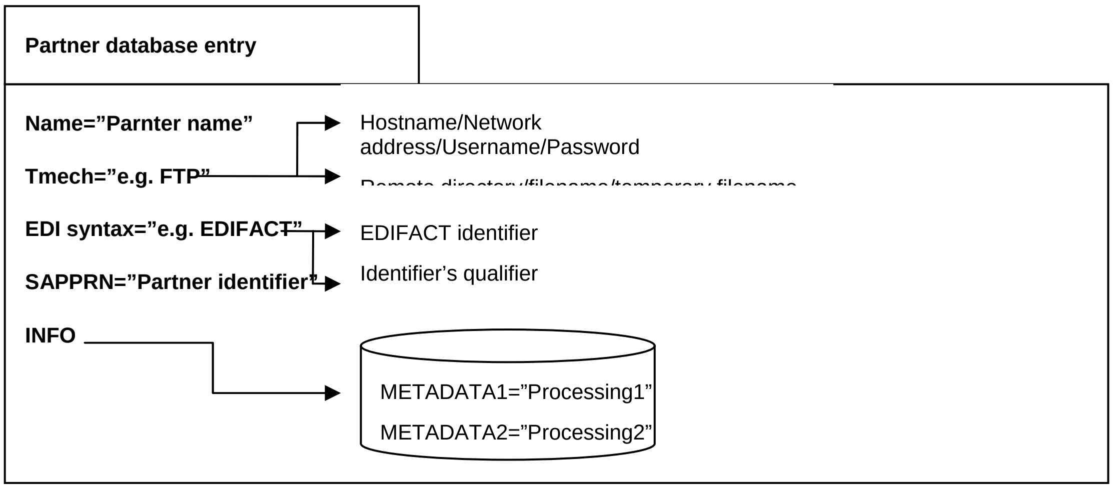


Figure 28. Information related to a Partner database entry.

4.3.2 *TRADEXPRESS ROUTING MODULE*

Because of the pre-built capabilities of the TX routing module, there is no need to start building the routing capabilities of the IRS from scratch. As stated in section 4.1, the routing module includes built-in features for analyzing message content. These are discussed next in this section alongside with the basic operating principles of the module.

TX has been designed to receive, route and processes different kinds of messages and files. However, TX is independent of the underlying data communication software. Instead, it entails interfaces to many common data communication applications that perform network-related tasks, including connection opening and receiving data by means of a network protocol. The interface uses signaling or queues for communication, and through the interfaces TX receives data for processing on an event-driven basis. As TX receives data, the routing module's source and syntax level routers can be used to analyze the data content and make a routing decision based on a combination of pre-built and user-defined routing rules. (Influe-Illicom 2007, pp.31-32) The routing concept of TX and the most common transport mechanisms and standard message formats supported by TX are presented in Figure 29 below.

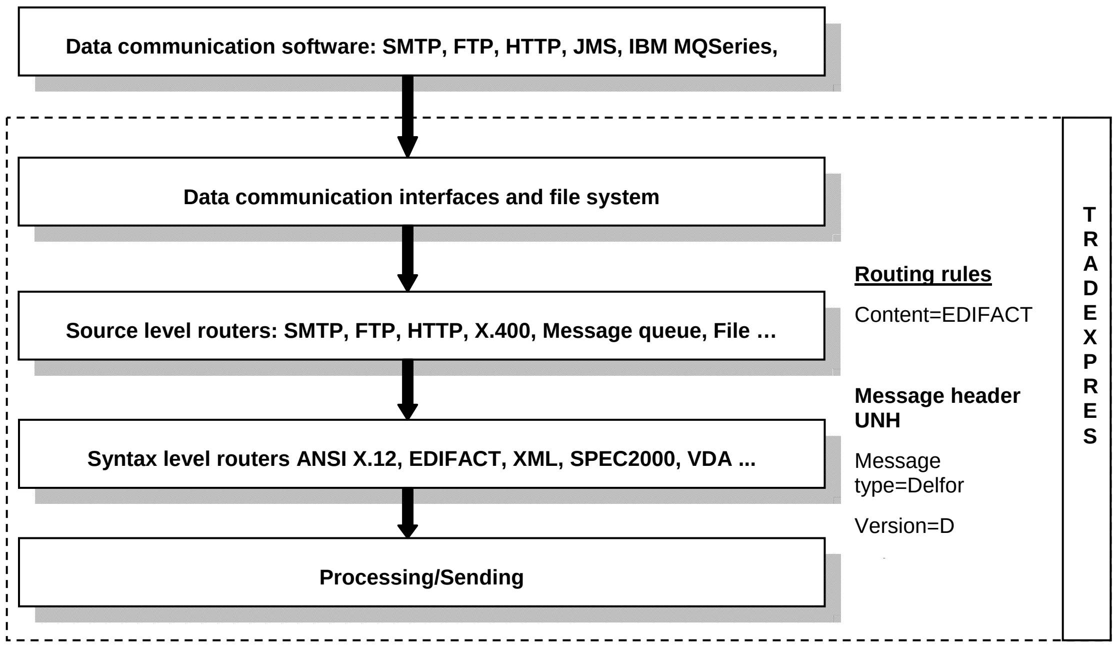

Data communication software: SMTP, FTP, HTTP, JMS, IBM MQSeries,
Data communication interfaces and file system
Source level routers: SMTP, FTP, HTTP, X.400, Message queue, File …
Syntax level routers ANSI X.12, EDIFACT, XML, SPEC2000, VDA ...
Processing/Sending
Routing rules
Content=EDIFACT
Message header UNH
Message type=Delfor
Version=D
TRADEXPRES

Figure 29. TX routing concept (Influe-Illicom 2007, p.31, modified).

As hinted above, the routing process of TX is divided into two layers: Source level and syntax level routing. Source level routers have the capability of identifying certain types of content. This capability includes, that source level routers are able to recognize XML, EDIFACT, ANSI X12, OVT, VDA and other standards-based message formats (Influe-Illicom 2007, p.31). If a source level cannot recognize incoming data to conform to one of these standards, it interprets the incoming data as “text” or “*”. This is the case with proprietary inhouse messages. Additionally, the capability has some limitations. One limitation is the inability to recognize the format of XML messages, if they lack an XML declaration.

Based on content and transport-specific parameters, a source level router determines the subsequent action, which may be syntax level routing, execution of an Edisend sending process or an external command. For instance, the file router can be configured to route EDIFACT messages to EDIFACT syntax level router and XML messages to XML syntax level router. This requires creating two distinct routing rules to the Routing database. Moreover, the file router provides a way for IRS to retrieve files from the file system and deduce the format of messages contained in these files. As TX receives incoming files from ISP’s customers and their designated partners, also other source level routers may be used for receiving purposes. The initial assumption is that these routers are configured to store the received files to the file system for IRS.

Syntax level routers allow more detailed analysis on standards-based messages. For instance, the EDIFACT syntax level router can deduct interchange and header level information from message and compare it to the routing rules specified in the Routing database. In many occasions, a routing decision needs to be based on information found in the message body and thus, the pre-built capabilities need to be extended. For instance, the message recipient identifier may need to be deduced from the message body's NAD segment. Comparable situations may arise for other syntax level routers as well. The needed extension can be based on custom RTE programs, as syntax level routers allow execution of two RTE programs during the syntax level routing process. These two programs are referred to as "translator" and "postprocessor". For example, the syntax level routers can be used to identify just the type of an incoming standards-based message and custom RTE programs can be used to deduce additional information required to make a routing decision. This way, it is possible to alleviate the constraints of TX syntax level routers and route messages to the correct processing and ultimately to the correct target systems.

4.3.3 TRADEXPRESS EDISEND MODULE

The TX sending process configurations are stored in Edisend database. For example, an edisend sending connection may involve sending a message or a file to a partner or merely executing a specific program. One entry in Edisend database contains information of how a sending process is started, what processing is conducted, a default recipient (a Partner database entry) and logging details, that is, what information of a sending process is stored in Syslog. Possible startup mechanisms for an edisend process are file arrival to a certain directory, directory scan and scheduled or external start-up. The scheduling itself is handled by the Scanner process. (Influe-Illicom 2007, pp.21-25)

The four steps, preprocessor, translator, interchange and transport, presented in Figure 30 constitute the data flow of an edisend process. These steps are all optional. Using operating system's standard input and output streams, the output of a previous step becomes input for the next. The preprocessor and translator steps typically involve executing an RTE program, but can also entail executing, for instance, UNIX shell commands. The interchange optionally builds an envelope to, for example, an EDIFACT message, while transport conducts the actual sending of data to the recipient. The correct information for interchange and transport step is retrieved from a partner profile in the Partner database. In case of an exception, for example, when the translator returns a negative exit code, it is possible to leverage TX's built-in Alarm functionality to report the error, take corrective action and possibly restart processing. (Influe-Illicom 2007, pp.25-29) This possibility exists also for TX routing module.

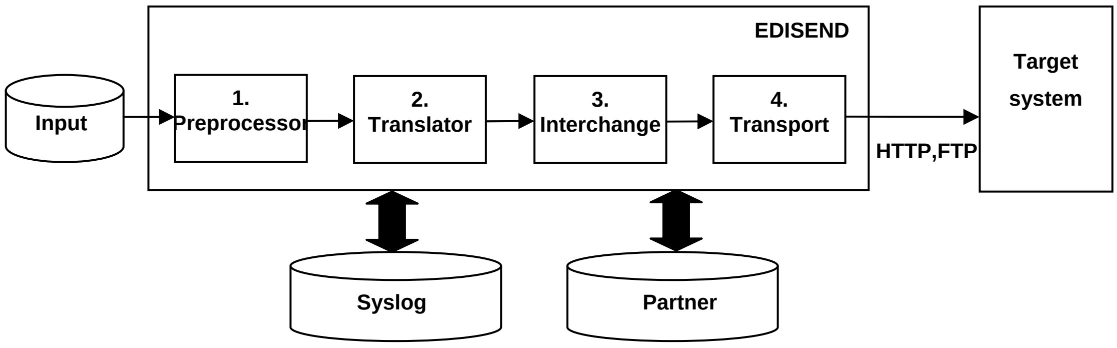
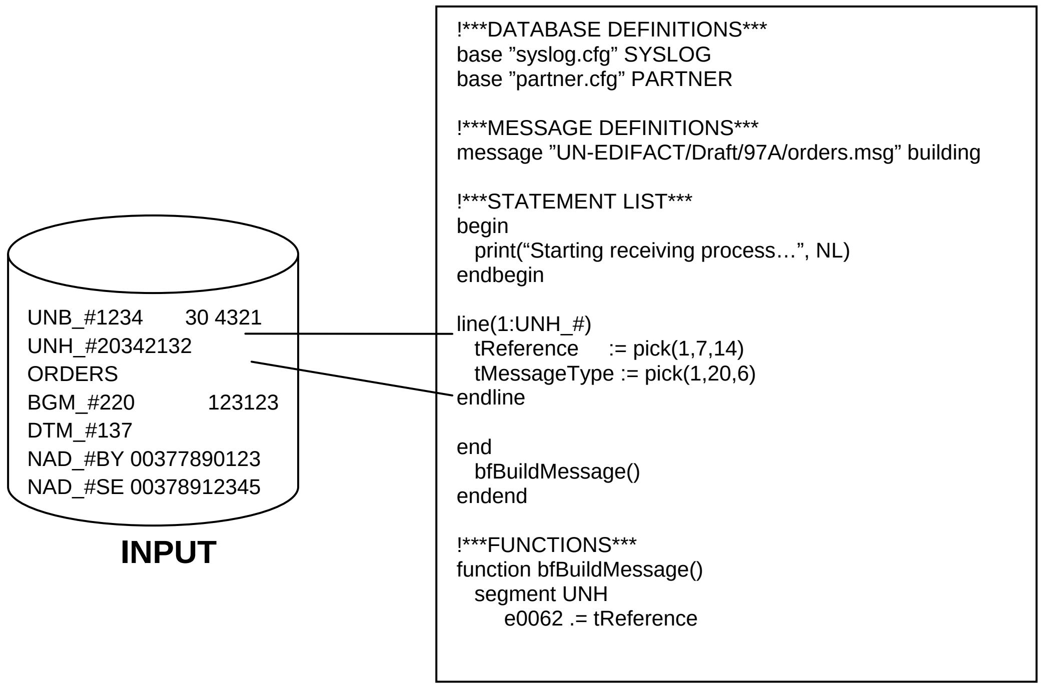


Figure 30. Data flow of an Edisend sending process.

Creating an edisend sending process is the TX standard way to handle the processing and sending of messages. Therefore, the IRS assumes that all message processing and sending activities utilize this facility. This assumption does not limit the operation of TX in any way, since the operation of an Edisend is very flexible, as discussed above.

4.3.4 Reliable transaction engine

TradeXpress includes a high-level programming language, RTE, primarily for creating message translation programs, but it is also suitable for developing a variety of other applications (Illicom 2004b, p.3). All custom coding required by the IRS is done with RTE. The operation of an RTE-program is by default data-driven, that is, program input is taken one line or segment at a time and run through the matching statement list (Illicom 2004b, p.6). This is illustrated in Figure 30. Creation of message transformation programs is based on message definition files that define both the structure of the message received as input and the message created as a result of the translation (Illicom 2004b, pp.3-4). The custom programs of the IRS also benefit from message definition files, since they make referencing to a certain data element in a message easier and thus ease the task of finding the necessary information from messages.



```
!***DATABASE DEFINITIONS***
base ”syslog.cfg” SYSLOG
base ”partner.cfg” PARTNER

!***MESSAGE DEFINITIONS***
message ”UN-EDIFACT/Draft/97A/orders.msg” building

!***STATEMENT LIST***
begin
  print(“Starting receiving process…”, NL)
endbegin

line(1:UNH_#)
  tReference     := pick(1,7,14)
  tMessageType := pick(1,20,6)
endline

end
  bfBuildMessage()
endend

!***FUNCTIONS***
function bfBuildMessage()
  segment UNH
      e0062 .= tReference
```

Figure 31. RTE-program’s basic structure.

The main components of an RTE program are optional database and message definitions, statement lists that include begin, end, line, as well as segment statements, and user-defined functions (see Figure 31). A statement list may contain several statements, for example assignments, control statements and function calls. (Illicom 2004b, pp.40-53) Since RTE is a high-level language, there are a variety of mechanisms that simplify programming. RTE includes proprietary memory management combined with associative memory that allow, for example, variables to be used without declaring them first and using arrays without concern of checking their boundaries. Moreover, the interfaces to TX databases, C-language and file system, add to the functionality of RTE. RTE also entails a host of built-in functions that reduce redundancy and speed up programming. There are ready-made functions for file as well as text handling, process control, database access and other more general purposes. (Illicom 2004b, pp.3-150)

Rte-programs are an integral part of TX processes, as discussed in sections 4.3.2 and 4.3.3. They largely define what is done during the execution of these processes and they can be used to control the execution of these processes through parameters. In IRS, RTE-programs are used in various stages of the routing procedure, for instance, to extend TX routing functionality and for message transformation purposes.

4.4 Designing the intelligent routing solution

In this section, the focus is shifted from the problem domain to the actual design of the IRS. This includes describing the system architecture as well as the building blocks of the system, how they interact and how the actual system is to be implemented in TX. Additionally, consideration is put on issues, such as, system maintenance and monitoring, exception handling, performance and security.

4.4.1 System architecture

The IRS is founded on the capabilities of the TX components introduced in section 4.3. These capabilities need to be supplemented with custom RTE-programs, in order to meet the objectives defined previously in section 4.1. All the necessary information of the involved partners and systems needed to route messages correctly, can be maintained in the Partner database. All the events that take place during IRS routing procedure can be stored in the Syslog database. The processing-nodes that need to be selected and activated by the IRS can be modeled as edisends. Figure 32 gives a coarse-grained view to the architecture of the IRS by depicting its major components and their interaction. The configuration of these components and the activities performed by the IRS are dissected in more detail in sections 4.4.2-4.4.4.

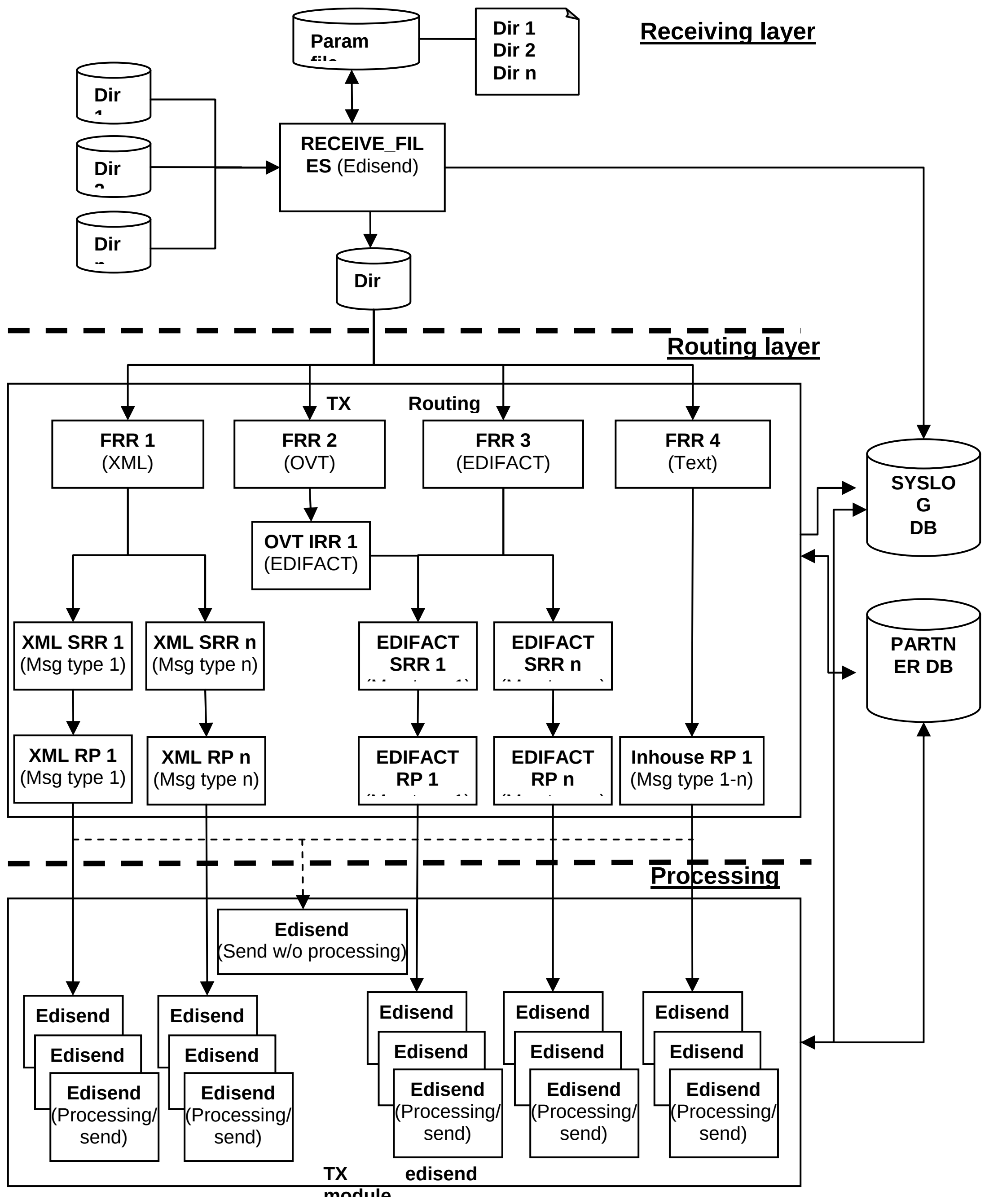


Figure 32. Coarse-grained architecture of the intelligent routing solution.

The architecture of the IRS, presented in Figure 32, is divided into three layers: receiving layer, routing layer and processing layer. At the upmost layer, the files received over a communication media and stored into various directories of TX server's file system are moved by the RECEIVE_FILES node to a single directory. This node is an edisend, which is configured to start periodically at an interval of 1 minute. It executes an RTE-program, which reads a parameter file for the names of the directories that may contain files that need to be routed by the IRS. Without the RECEIVE_FILES node, the amount of needed file routing rules instances would have to be multiplied by the amount of directories. For instance, 4 routing rules times 10 directories would result in 40 routing rules in Routing database instead of mere 4 that are required in case the RECEIVE_FILES-node is used to scan the directories.

The core of the IRS, namely the routing layer, consists of file router (FR) routing rules (FRRs), an intermediary router (IR) routing rule (IRR), syntax router (SR) routing rules (SRRs) and RTE routing programs (RPs). A FRR is created for each message format. Thus, FR is configured to handle the task of identifying the format of incoming messages. All proprietary message formats are regarded by the IRS as text, and thus covered by a single FRR. Hence, to cover XML, OVT, EDIFACT and text-based inhouse messages, four FRRs are needed. Depending on the identified message format, the next step of processing is either an OVT IR or a SR. The IR is needed for handling OVT-framed messages. The IR removes the OVT envelope from the messages, stores the retrieved information from envelope as parameters and hands the message to EDIFACT SR. Initially, only EDIFACT messages are expected to be sent OVT framed.

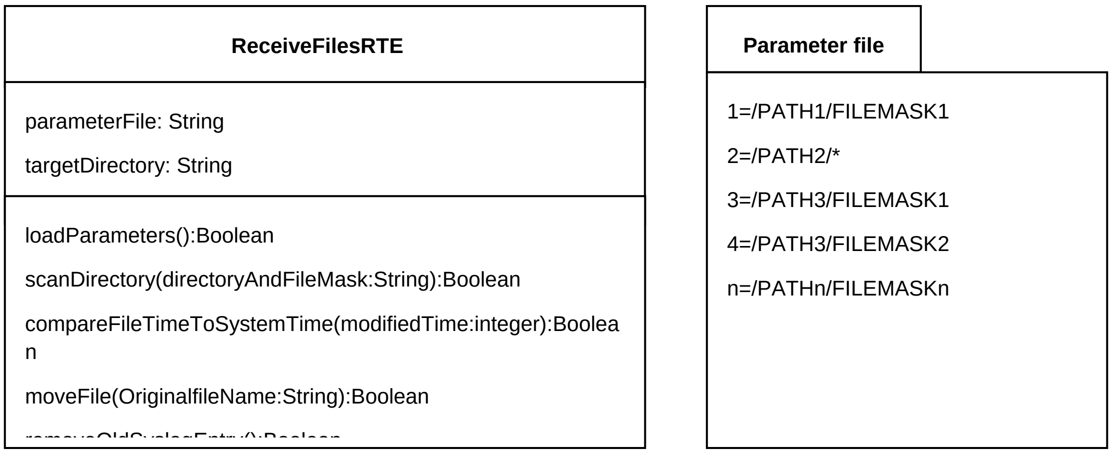
The SRs in the routing layer are configured to handle the task of recognizing message types, except for the proprietary messages. This is done by creating a SRR for each message type. In case of proprietary messages, this task is assigned to an RP, which is directly executed by the FR. Different XML and EDIFACT message types are recognized by the XML and EDIFACT SRs. As the message type of an XML or EDIFACT message is known, the next task is to execute an RP, which deduces recipient information from each message in the interchange, determines subsequent processing, filters messages and launches the correct processing for each message. A separate program needs to be created for every standards-based message type. This way, it is possible to leverage message references within RTE code to deduct the necessary routing information from the messages. With proprietary format messages, the same program is used both to identify the message type and conduct the actions listed above.

The processing layer presented in Figure 32 is somewhat treated as a black box by the IRS. The RPs in the routing layer start an edisend in the processing layer and pass the received file as an input and the deduced information as input parameters. Then, the IRS expects that the processing layer handles message processing and sending to the actual recipient. Hence, the IRS is not concerned on what kind of processing is conducted within the edisends. The RPs must only know the edisend name they need to start. This information is stored in the Partner database. Once the IRS has deduced message format and type and recipient identifier, it searches the corresponding partner from the Partner database and using the message format concatenated with message type as a search key, it retrieves the name of the edisend it should start. The processing itself may include message transformation from XML to EDIFACT format, code conversion, data enrichment or any other processing required by a customer of an ISP. Moreover, the processing may include execution of multiple consecutive edisends, if required. Next, the steps involved in the routing procedure and the components of the IRS are presented in more detail.

4.4.2 RECEIVING LAYER CONFIGURATION

In section 4.2.1, it was specified that the IRS does not directly receive data sent by or fetched from the source systems over a communication media. The IRS expects that data is received by mechanisms discussed in section 4.3.2 and stored to the file system of the TX server and made available for processing after the data transmission is complete. For instance, if two systems owned by different companies transmit files over SFTP, the files are put into separate directories in the TX server due to security reasons. Hence, the first task of the IRS is to retrieve these files for routing from these separate directories. In the previous section 4.4.1, the task of scanning defined directories for files was assigned to the RECEIVE_FILES node. This node is essentially an edisend, which is scheduled to start at 1 minute interval. It executes an RTE program, which reads a parameter file including a list of directories that need to be scanned. If the program finds files, it moves them to a single directory, for instance $HOME/incoming/. For each file moved to the aforementioned directory, the Scanner starts a process for the

FR. Figure 33 shows the key elements of the RTE program and the structure of its parameter file.


ReceiveFilesRTE
parameterFile: String
targetDirectory: String
loadParameters():Boolean
scanDirectory(directoryAndFileMask:String):Boolean
compareFileTimeToSystemTime(modifiedTime:integer):Boolean
moveFile(OriginalfileName:String):Boolean
Parameter file

```
1=/PATH1/FILEMASK1
2=/PATH2/*
3=/PATH3/FILEMASK1
4=/PATH3/FILEMASK2
n=/PATHn/FILEMASKn
```

Figure 33. ReceiveFiles class and the structure of the parameter file containing the directories that need to be scanned.

The functionality that was described above is gained by implementing the class ReceiveFilesRTE. This class entails the operations loadParameters, scanDirectory, compareFileTimeToSystemTime, moveFile and removeOldSyslogEntry, as well as the attributes parameterFile, and targetDirectory. A description of the class operations is given below on

Table 2. The attributes contain the parameter file’s full name and the directory name, where the incoming files need to be moved. The parameter file is constructed using consecutive numbers followed by an equals sign and the path name and the file mask of incoming files that need to be routed. An asterisk is used to denote that all files arriving to a directory are moved to the target directory.

Table 2. The operations of the class ReceiveFilesRTE.

<u>Operation</u>	<u>Description</u>
loadParameters	Loads the contents of the parameter file into a text array. Returns the Boolean value TRUE if the load was successful; otherwise FALSE.
scanDirectory	Receives a string containing a directory name and a file mask as an input parameter, and scans the given directory for files that match the file mask. Returns the Boolean value TRUE if the scan was successful; otherwise FALSE.
compareFileTimeToSystemTime	Receives the modified time of a file as an input parameter and compares it to the system time subtracted by one minute. Returns the Boolean value TRUE if the system time subtracted by one minute was greater than the modified time; otherwise FALSE. This operation ensures that no incomplete files are moved.
moveFile	Receives the original file name as an input parameter and moves the original file to the target directory given as a class attribute. Returns the Boolean value TRUE if the move was successful; otherwise FALSE. Preserves the original file name, but adds the Syslog index of the edisend process that started it to the file name when the file is moved.
removeOldSyslogEntry()	Removes the Syslog entry created by the previous run of RECEIVE_FILES edisend, if its field USERTEXT32 contains value 0. This indicates that 0 files were retrieved during previous execution of the RECEIVE_FILES edisend. Returns the Boolean value TRUE if the removal was successful; otherwise FALSE.

In addition to implementing the defined operations, a key issue regarding the program implementation is logging. The program must write enough information to Syslog, in order to ensure proper system monitoring and traceability of the messages that propagate through TX. The minimum requirement is that the full file names of the retrieved files are written to the log-file of Syslog database (see Figure 34). Additionally, the amount of retrieved files needs to be written into the USERTEXT32 field of the Syslog. This information is used by the removeOldSyslogEntry operation, which removes unnecessary entries from Syslog. Moreover, file naming is essential. The operation moveFile should add the Syslog index of the edisend process that started the program to the original file name. For instance, a file named ORDER_20080805_1234 should be renamed to 90102_ORDER_20080805_1234, where 90102 is the Syslog index of the edisend process. This way, the next step in the IRS routing procedure has the information about the edisend process that retrieved the files and backwards tracing is possible.

```
*********************************************

* Execution begins @ 2008.09.10 1400

*********************************************

/stx/users/production/incoming/http/1234567

/stx/users/production/incoming/http/1234568

/home/companyX/ORDER_20080805_1234

/home/companyY/XML_ORDER_COMPANY_45673687

------------------------------------------

Total: 4 files retrieved

*********************************************

* Execution complete @ 2008.09.10 1400

*********************************************
```

Figure 34. Example log file of a RECEIVE_FILES entry in Syslog.

4.4.3 ROUTING LAYER

As described previously in this chapter, the IRS deduces interchange format, message type and recipient information from the received files and determines message routes based on the deduced information. The basic components used to implement this activity were presented in section 4.4.1. In this section, the configuration and operation of these components is presented in more detail.

4.4.3.1 Source level router configuration

In section 4.4.1, the task of deducing interchange format was assigned to FR. TX source-level routers, including FR, have the capability of recognizing certain standard formats based on built-in rules, as discussed previously in this chapter. This capability has its shortcomings, including the inability to recognize XML format without XML declaration, but these can be overcome by the IRS. In order to recognize XML, OVT, EDIFACT and proprietary inhouse messages, four FRRs need to be created to the Routing database for the IRS. Table 3 shows the key parameters of these FR entries. When a new standards-based message format needs to be added to the IRS, a new FRR is created. If TX does not have a pre-built capability to recognize the new message format, these messages are handled as proprietary messages.

Table 3. Key parameters of the file router routing rules.

Name	Priority	Selection criteria filename	Content	Command	Command parameters
FILE_XML	10	$HOME/incoming/*	XML	XML	
FILE_OVT	20	$HOME/incoming/*	OVT	OVT	
FILE_EDIFACT	30	$HOME/incoming/*	EDIFACT	EDIFACT	
FILE_INHOUSE	98	$HOME/incoming/*	TEXT	COMMAND	$HOME/control/InhouseRouterRTE _FILE_ _INDEX_

Table 4. Key parameters of the intermediary router routing rules.

Name	Priority	Content	Command	Command parameters
OVT_EDIFACT	10	EDIFACT	EDIFACT	

The FRRs need to have a priority associated with them. The priority determines the sequence in which the routing rules are matched by the FR (see Figure 35). A priority number 1 means higher priority than priority number 2. The FILE_INHOUSE rule must have the lowest priority, since content parameter “TEXT” would match to every other message format, in addition to the proprietary inhouse formats it is aimed for. The selection criteria filename parameter is set to “$HOME/incoming/*” for every routing rule. This parameter is automatically added to the user environment’s Scanner configuration file, so that the Scanner process is able to monitor file arrival on the given directory and start a process for the FR for each file in the given directory. The asterisks in the filename parameters ensure that the rules match to every file that is moved to the given folder by the RECEIVE_FILES-node. Thus, the task of determining the message format is independent of the file names.

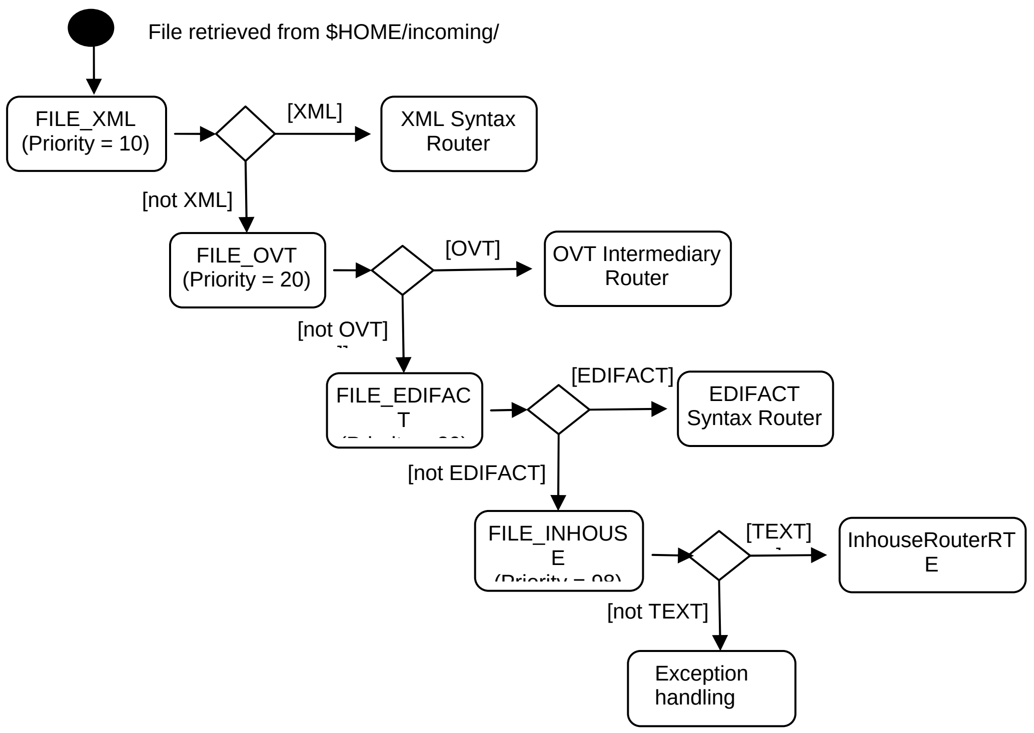


Figure 35. Activity diagram presenting the logic of file router operation and routing rule matching.

The decision of the subsequent action is done based on the value on the routing rule's content parameter. The FR analyses the content of the retrieved file and matches it to the values found from the content parameters of the routing rules (see Figure 35). If a match is found, the action specified in the rule's command parameter is executed. For FILE_XML, this action is XML SR, for FILE_OVT it is OVT IR, for FILE_EDIFACT it is EDIFACT SR and for FILE_INHOUSE it is an InhouseRouterRTE RTE-program. If the FILE_OVT rule is matched, then the processing is handed to the OVT IR, which analyses the data content within the OVT frames and makes a routing decision based on the content. Initially, only EDIFACT messages are expected within OVT frames (see Table 4), as mentioned in section 4.4.1. In case the FR doesn't find a match for the retrieved file in the routing rules, it triggers an exception handling procedure. Exception handling is also triggered if the OVT IR does not find a match or the RTE-program returns a negative exit code to the FR that executed it. The handling of exceptions is discussed in more detail in section 4.6.2.

4.4.3.2 Syntax level router configuration

As the FR or the OVT IR has matched a routing rule and the format of interchange in a file is known, the next step is to deduce the type of messages. The retrieved file is passed either to a SR or in case of proprietary message format, to the InhouseRouter RTE-program. The operation of the InhouseRouter is presented in the following section 4.4.3.3. For the syntax routers' part, the operation is quite similar to the operation of the FR, described in the previous section 4.4.3.1. The main principle of how the SRs are used to deduct the message type from the interchange is presented in this section.

The EDIFACT SR extracts information from the UNG, UNB, and UNH segments (Influe-Illicom 2007, p.143) of EDIFACT messages and matches this information to the specified routing rules in a priority order. The UNB-segment carries information about the interchange sender and recipient. However, this recipient information is often insufficient and routing needs to be based on recipient information found from, for instance, NAD-segment in the message body. Therefore, the IRS uses EDIFACT SR only to identify the message type and a separate RTE program to deduce information about recipients in the interchange. This way, it is possible to create more complex routing rules that are based on information residing anywhere in a given interchange. Table 5 shows how the message type details in the UNH-segment are set as routing parameters for an order message. A similar routing rule needs to be created for each message type. In the IRS, different EDIFACT message versions and releases, for instance, ORDERS version D release 93A and ORDERS version D release 97A require separate routing rules.

Table 5. The routing parameters found from EDIFACT message's UNH-segment (Influe-Illicom 2007, p.144).

Parameter	Description / example value
Message	Indicates the message type, such as ORDERS, INVOIC, DESADV, or other EDIFACT message type names. Extracted from EDIFACT element S009.0065. **EXAMPLE:** ORDERS
Version	Indicates the message version of the message type. Extracted from EDIFACT element S009.0052. **EXAMPLE:** D

Release	Indicates the release number within the current message version. Extracted from EDIFACT element S009.0054. **EXAMPLE:** 97A

The operation of the XML SR differs from the operation of the EDIFACT SR, because the XML syntax is different than the EDIFACT syntax. In EDIFACT, the segment UNH, which carries message type information, must exist in every message. In case of XML, there is no such element or attribute that specifies the message type for every XML message. Instead, the XML message type needs to be identified individually for each different message. However, XML SR is more flexible than EDIFACT SR, and allows routing information to be deduced from any element or attribute of an XML message. The XML SR can route messages using a pair consisting of an element name and element value (Influe-Illicom 2007, p.173) or attribute name and attribute value. The IRS uses this functionality to identify the XML message types. Hence, each different XML message type routed by the IRS needs to have a separate routing rule that uniquely identifies one incoming message type. For instance, the rule Envelope/Message/Type=ORDER would match to the following XML message:

```
<?XML VERSION="1.0" encoding="UTF8" ?>
<Envelope>
        <Message>
                <Reference>123456798</Reference>
                <Type>ORDER</Type>
.
.
        </Message>
</Envelope>
```

As with the EDIFACT SR, a separate RTE routing program is used to deduce the interchange recipients. For each routing rule, the name of this message type -specific program needs to be stored in the routing rule's translator parameter. When a SR finds a matching routing rule for a message, the specified program gets executed. In case a SR does not find a match for a message in routing rules, an exception handling procedure is launched. As previously mentioned, exception handling is discussed in more detail in section 4.6.2.

4.4.3.3 RTE routing program specification

The tasks of deducing message recipient identifiers from interchange, filtering the interchange to individual messages, determining correct edisend process for each message, and launching the edisend process for each message were assigned to custom RPs in section 4.4.1. In case of standards-based messages, a separate RP is needed for each message type. With proprietary formats, a single RP is sufficient for all message types. However, this RP also needs to perform the task of deducing message type. As the amount of different message types can potentially grow large, it is necessary to avoid rewriting code. Hence, the operations common to all of the RPs need to be carefully identified and only implemented once. The logic and operations that need to be implemented by each RP, that is, are dependent on the message structure also need to be identified. Figure 36 shows the main routing program classes.

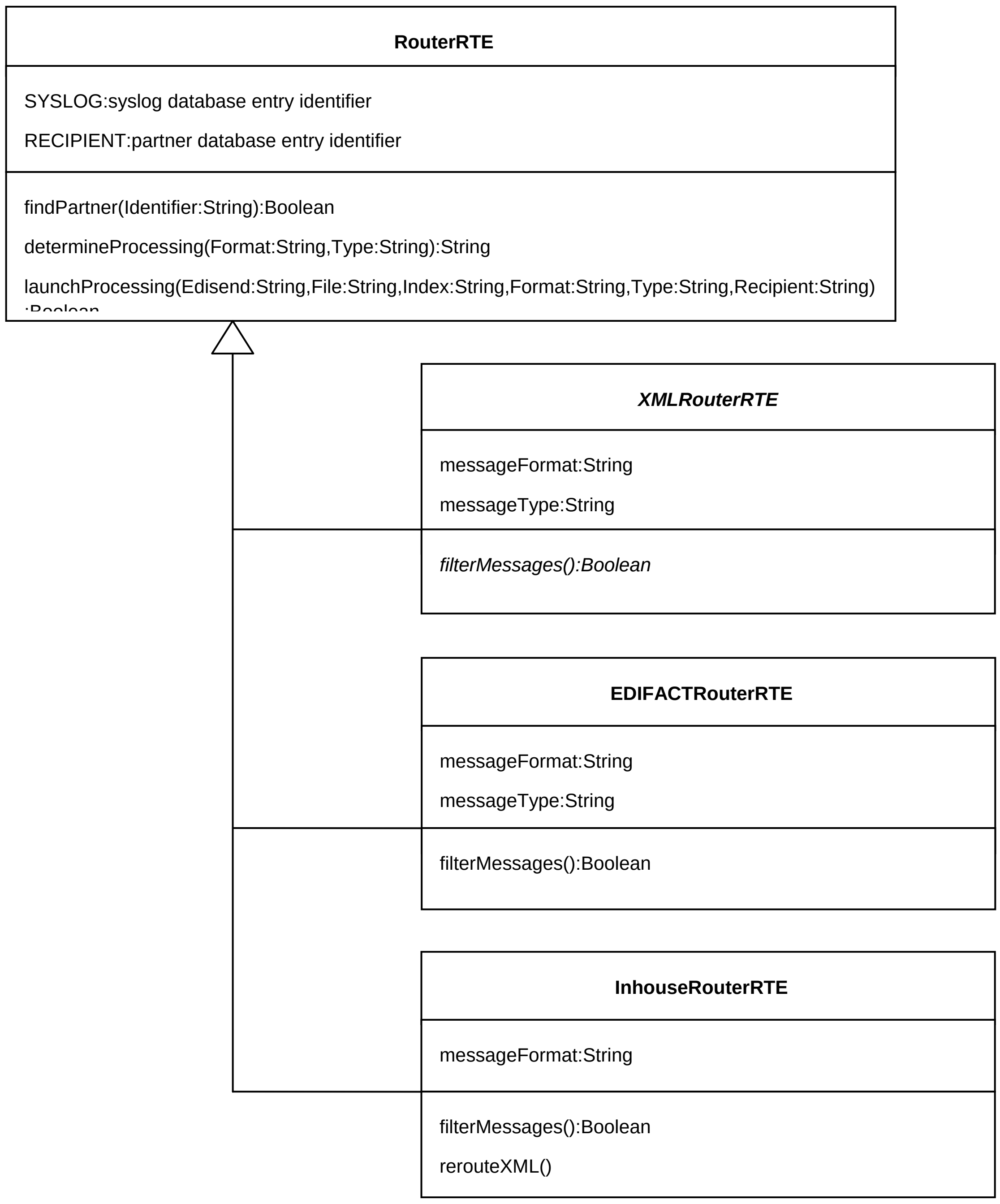

RouterRTE
SYSLOG:syslog database entry identifier
RECIPIENT:partner database entry identifier
findPartner(Identifier:String):Boolean
determineProcessing(Format:String,Type:String):String
launchProcessing(Edisend:String,File:String,Index:String,Format:String,Type:String,Recipient:String)
XMLRouterRTE
messageFormat:String
messageType:String
filterMessages():Boolean
EDIFACTRouterRTE
messageFormat:String
messageType:String
filterMessages():Boolean
InhouseRouterRTE
messageFormat:String
filterMessages():Boolean
rerouteXML()

Figure 36. Class diagram of the RTE router programs.

The carrying idea of the RPs is that the operations common to all message formats are implemented in one super class, RouterRTE. The classes that are specific to each different format inherit the identical functionality of the super class and add new functionality of their own. Table 6 gives more detailed description of the operations of these classes. In RTE's terms, the super class is implemented as an include file. An RP is an instance of one of the classes InhouseRouterRTE, *XMLRouterRTE* or EDIFACTRouterRTE. All three aforementioned class instances are implemented as standard RTE programs. The *XMLRouterRTE* is declared as an abstract class and operation *filterMessages* as an abstract operation since each instance of the class need to implement its own version of the operation. As explained previously in section 4.4.1, a separate RP needs to be implemented for each different XML and EDIFACT message type. One instance of class InhouseRouterRTE is sufficient for every proprietary message type. Suggestions for the RouterRTE and the RP implementation are given in following section 4.4.3.4.

Table 6. Descriptions of the operations of classes RouterRTE, *XMLRouterRTE*, EDIFACTRouterRTE and InhouseRouterRTE.

Operation	Description
findPartner	Receives the partner identifier as an input parameter, and finds the corresponding partner entry from the Partner database. Returns the Boolean value TRUE if the operation was successful; otherwise FALSE. Operation is implemented in class **RouterRTE**.
determineProcessing	Receives the interchange format and type as input parameters and determines based on these the correct processing. Each Partner database entry involved in the IRS contains information about processings in its info file (see section 4.4.3.5 for details about partner metadata). Returns a string containing the name of the edisend process; or an empty string if no processing is found from partner metadata. Operation is implemented in class **RouterRTE**.
launchProcessing	Receives the edisend process name, input file full name, Syslog index of the routing process instance, interchange format, type and recipient as input parameters. Based on input parameters, starts the edisend and gives it the file as input and message format, type and recipient as parameters. Returns the Boolean value TRUE if the operation was successful; otherwise FALSE. Operation is implemented in class **RouterRTE**.
filterMessages	Reads the input file, and writes each message in the file to a separate file. Stores the file names of the split files in a table. Preserves the original file. The behavior of this operation is largely dependent on the interchange format and message type. Hence, it is implemented in instances of classes ***XMLRouterRTE***, **EDIFACTRouterRTE** and **InhouseRouterRTE**.
rerouteXML	Writes an XML declaration to the beginning of the original file, moves the new file to the directory $HOME/incoming/ and writes the information REROUTED to Syslog's USERTEXT32 field. The new file name contains the original file name and the index of the routing process that rerouted the file. Returns the exit code 0 to the process that started the program. Operation is implemented in the sole instance of class **InhouseRouterRTE**.

4.4.3.4 RTE routing program implementation

In addition to the operations defined in previous section 4.4.3.4, there are also other issues regarding the RP implementation. Firstly, the instance of class InhouseRouterRTE must perform the task of deducing the message type found in an incoming file. The received file is traversed through line by line and deducing the message type is done with line matching rules. A sample rule for a certain type EDI-inhouse could be:

```
line(1:”UNH_#”)
        tMessageType := pick(1, 20, 6)
        if tMessageType <> EMPTY then
                bMessageType := TRUE
        endif
endline
```

Thus, a line matching rule is written for each message type the program handles. Line matching rules are also used to deduce message recipient identifiers. Additionally, a line matching rule is added for every XML message type that lacks an XML declaration. A line matching condition, where the operation rerouteXML is called, could be line(1:”<ROOT>”), where “ROOT” is the root element of the XML message.

Secondly, with standards-based messages, message recipient identifiers are deduced from the messages using segment statements. The incoming message is traversed through segment by segment and matched to the segment statements in RTE routing programs. A sample rule for an EDIFACT Delfor D 97A message could be:

```
segment NAD g2
        if e3035=”SE” then
                tIdentifier:=eC082.3039
        endif
endsegment
```

Each instance of EDIFACTRouterRTE and *XMLRouterRTE* must include a message definition, which defines the message reference it applies to. XML message references can either be schemas or Document Type Definitions (DTDs). EDIFACT message references can be taken from the TX EDIFACT message library.

Thirdly, each RP needs a control logic that controls the order, in which the required operations are performed. This logic is dependent on both interchange format and message type. Thus, the logic differs to some extent with each RP. Appendix B shows a suggested implementation of class EDIFACTRouterRTE for EDIFACT Orders D 97A messages. Appendix A contains a suggested implementation of class RouterRTE. In the example found from appendix B, the message recipient information may be found either from the UNB segment of the interchange header or from the NAD segments of the message body. The program first tries to find a Partner database entry based on the interchange recipient and determine processing based on message format and type. If a match is found, the program launches the determined processing and gives the original message as an input to the edisend. If no match is found, the program continues searching for recipient identifiers from the NAD segments. For each found identifier, the program finds a partner, determines processing and records the processing name in a table. The identifiers are also stored in a table. Next, the program filters interchange into files that contain only one message and stores the file names into a table. Lastly, the program goes through the table that contains the processing names and starts an edisend process for each table record, giving the correct split file as an input for each edisend process.

The described type of logic can easily be replicated to other RPs aimed for different EDIFACT message types and versions. For other formats, the program logic may differ, but generic principles can be derived regardless of format:

1. Deduce message recipient identifier from message.
2. Find Partner database entry based on the recipient identifier.
3. Determine processing from partner’s info file and store processing into a table.
4. Repeat steps 1-3 for as many times there are messages in an interchange.
5. Filter messages and store resulting file names to a table in the same order as the processing names.
6. Start processing for each filtered message.

Fourthly, logging is needed also at this stage for proper system monitoring and message tracking. The logging functions were deliberately excluded from Appendices A and B to keep the focus purely on the program logic. At minimum, each routing process should write the index of the RECEIVE_FILES process to the Syslog database’s USERTEXT8-field. This index is received in the input file name. It is advisable to implement common logging functions in one include file, which the RPs may then utilize. Additionally, the RPs should provide enough logging of their operations to enable proper problem-solving in case an exception occurs during a routing process.

Lastly, exceptions that occur during RP execution need to be handled by the IRS. If a key operation fails, an RP must signal exception to the routing process with negative exit code. However, this signaling should be conducted delayedly, that is, all messages should be analyzed before signaling the exception. For instance, an interchange may contain an unknown partner identifier in addition to several known ones. In this case, valid messages should be routed and erroneous ones should not be routed. This type of functionality is achieved by maintaining error counters, as suggested in Appendix B. Error handling is further discussed in section 4.6.2.

4.4.3.5 Partner metadata

For each different recipient identifier found in the incoming messages, an entry must be created into the Partner database. By default, an entry is created for each customer of an ISP, even if they do not receive any messages. Above all, each Partner database entry is associated with one identifier. This identifier is stored in the SAPPRN field of the Partner database. The identifier must be unique, as discussed in section 4.2.1, since it is used to determine the target system the message is sent to and also used as a search key to find the correct entry in the Partner database. It is possible that the identifier found in a message only identifies the business party the message is intended to, and not the party the message is actually sent to. For instance, an identifier may identify a particular business unit of a company, and all messages intended to that company are sent to an ISP. An entry must also be created for that ISP and a reference that points to the “ISP partner” created for each “business unit” partner. This reference can be stored in a partner-specific info file of the Partner database. However, RPs always make a routing decision based on the identifier found in a message. Determining the actual recipient of the message and possible identifier conversion issues are handled in the edisends of the processing layer that control the sending of the messages. Key Partner database entry information used by the IRS is shown in Figure 37.

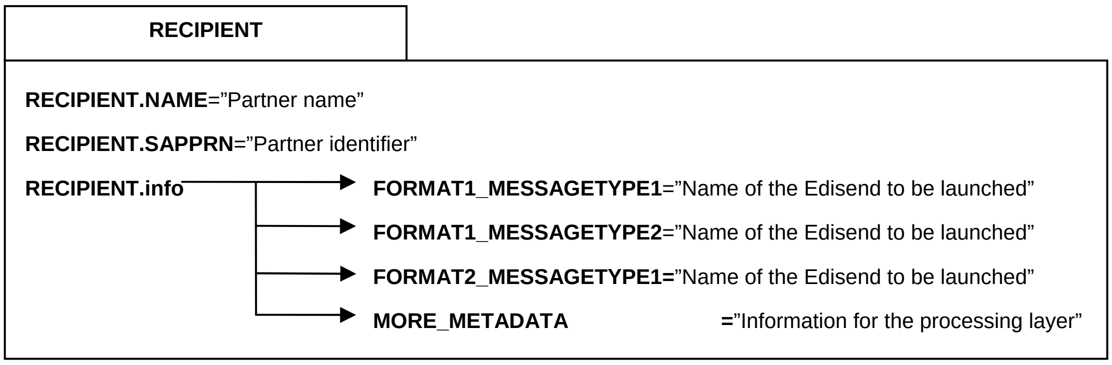

RECIPIENT
RECIPIENT.NAME="Partner name"
RECIPIENT.SAPPRN="Partner identifier"
RECIPIENT.info
FORMAT1_MESSAGETYPE1="Name of the Edisend to be launched"
FORMAT1_MESSAGETYPE2="Name of the Edisend to be launched"
FORMAT2_MESSAGETYPE1="Name of the Edisend to be launched"
MORE_METADATA ="Information for the processing layer"

Figure 37. Partner database entry information used by the IRS.

The information of which edisend process to start for each incoming message, is also stored in the partner-specific info file of the Partner database. For example, if a customer of an ISP would send EDIFACT Delfor D 97A messages to its business partner A, the corresponding partner A entry's info file could contain the following line: EDIFACT_DELFORD97A=SEND_MESSAGE. On the left side of the equals sign is interchange format, a "_" -sign and message type, and on the right side is the name of the edisend process that needs to be started. The RTE operation to retrieve the name of the edisend process is straightforward. As an RP has deduced the message recipient and found the corresponding Partner database entry, it loads the contents of the entry's info file into a table. Using interchange format, a "_" -sign and message type, it constructs a search key and retrieves the name of the edisend from the table. This way, an RP is able to start the correct edisend process.

4.4.4 PROCESSING LAYER CONFIGURATION

The processing layer was specified in section 4.4.1 to handle the transformation, as well as other processing, of the incoming messages and sending them to the correct recipients, that is, target systems. Although the IRS is not concerned on what processing is conducted within the edisends of the processing layer, the interaction between the routing layer and the processing layer must be defined. Additionally, each edisend of the processing layer must implement a logging operation in order to ensure proper monitoring of the overall system. Edisend naming convention must also be defined, because the routing layer is dependent on the edisend names. Moreover, the start-up type of each edisend launched by a routing program must be “external”.

The interaction between the routing layer and the processing layer is conducted by passing parameters. The RPs of the routing layer start an edisend process for each filtered message file and pass the Syslog index of the routing process, input file name, the deduced interchange format, message type and recipient identifier as input parameters. These parameters are usable in an RTE program executed by an edisend process. An example command executed by an RP could be:

```
edisend IRSINDEX=91000 IRSFORMAT=EDIFACT IRSTYPE=DELFORD97A
IRSRECIPIENT=123456789  DELFORD97A_send
/stx/users/editest/outgoing/1_DELFOR_20080929_123456
```

All edisend processes are executed asynchronously, that is, the routing programs do not wait for their completion. The parameter names passed to edisend sending processes are IRSINDEX, IRSFORMAT, IRSTYPE and IRSRECIPIENT. An edisend process may retrieve the values of these parameters in RTE code, using “p” and the parameter name, for example, pIRSINDEX. The parameter values can be used to control the sending process and for logging purposes.

In section 4.1, it was specified that the incoming messages are routed to the target systems either with or without processing. The suggested way to send messages without processing is to create a single edisend that handles the sending of all kinds of messages, and a simple RTE program that controls the edisend sending parameters. The idea is that the RTE program finds the correct entry from the Partner database based on the input parameters given by the routing process and sets the partner as the recipient of the edisend process. The edisend process then executes the sending based on the entry’s transport-specific parameters. It must be noted that the partner the message is sent to may be different than the one given as an input parameter, as discussed in previous section 4.4.3.5. Additionally, the RTE program needs to implement an operation that writes the Syslog index of the routing process to the Syslog database’s field USERTEXT8. The index is received in parameter IRSINDEX. The same operation must be implemented by each edisend of the processing layer for system monitoring purposes.

As noted above, there is a dependency between the routing layer and the names of the edisends of the processing layer. The edisend name is stored in the Partner database for each entry that is associated to the edisend. Hence, an edisend name cannot be changed without making changes to all associated Partner database entries. If the edisend name is changed without changing the references found in the partner info files, an RP would try to start a non-existent edisend process. Additionally, the name of an edisend must be unique. Having two or more edisends with the same name would result in a situation, where an RP cannot determine which edisend instance to execute. New edisends may be added to the processing layer, as long as their names are correctly stored to the Partner database. Above all, there should be a procedure developed for the maintenance of both edisends and partners that are involved in the IRS. A well-defined procedure ensures that the edisend names and partner metadata stay synchronized.

4.5 System maintenance

Given the amount of standards-based as well as proprietary message formats and types, it’s not feasible to design the IRS to support all formats and types at once, as discussed in section 4.1. However, after the initial setup, the IRS must support adding new message formats, types, recipients and processing capabilities as they are requested by the customers of an ISP. The system maintenance capabilities were defined in section 4.2.1. Adding new message formats and types is handled according to the procedures defined in sections 4.4.3.1-4.4.3.4. Adding new message recipients and processing capabilities can be done by following the rules specified in sections 4.4.3.5 and 4.4.4. New folders to be scanned for incoming files are added to the RECEIVE_FILES node’s configuration file in the format defined in section 4.4.2. Removing obsolete routes is also supported by the system, without breaking it.

There are five main change cases that need to be covered by the IRS. Figure 38 lists these change cases and their impact on the system. The impact of the change cases is largely dependent on how a change case affects the interface between the ISP and its customers and between ISP and its customers' partners. In this context, an interface is equal to the message formats and types used in the message-based communication. A change case may affect all three layers of the IRS, if it entails modifications to the interface, as comes out from Figure 38. The impact of changes to data communication interfaces and the procedure of how the changes are requested by the customers are beyond the scope of IRS design.

CHANGE CASE 1: New partner for a customer using an existing interface

DESCRIPTION: A customer of an ISP needs to start sending messages to a new business partner and/or receive messages from that same partner, using a message format and message types that are defined to the IRS.

IMPACT: New entry is created to the Partner database, if the new partner's identifier does not already exist for the purposes of another customer. Partner entry's metadata is added, or updated if the partner exists. In case the new partner sends messages to the customer, the RECEIVE_FILES-node's configuration file is updated.

--**CHANGE CASE 2:** New partner for a customer with a new interface

DESCRIPTION: A customer of an ISP needs to start sending messages to a new business partner and/or receive messages from that same partner, and the new partner requires a message format and/or message types that are defined to the IRS.

IMPACT: New entry is created to the Partner database, if the new partner's identifier does not already exist for the purposes of another customer. Partner entry metadata is added, or updated if the partner exists. In case the new partner sends messages to the customer, the RECEIVE_FILES-node's configuration file is updated. New source level routing rule is added, or InhouseRouterRTE is updated, if the partner sends messages to the customer in a format not known by the IRS. New SRRs and new instances of RPs are created, or InhouseRouterRTE is updated, if the message types used in communication are not defined to the IRS. New processing capabilities are added to the processing layer, if the messages sent to or sent by the new partner need to be translated to a new format or from a new format or if a processing capability for a new message type does not exist.

--**CHANGE CASE 3:** New message type sent to or sent by an existing partner

DESCRIPTION: A customer of an ISP needs to start sending new types of messages to an existing business partner or receive new types of messages from that same partner.

IMPACT: Partner entry's metadata is updated. New SRR and new instance of an RP is created, or InhouseRouterRTE is updated, if the message type used in communication is not defined to the IRS. New processing capability is added to the processing layer, if a processing capability for the new message type does not exist. If the partner has not previously sent any messages, the RECEIVE_FILES-node's configuration file is updated.

--**CHANGE CASE 4:** New customer for an ISP

DESCRIPTION: An ISP acquires a new customer that has a number of partners. The new customer or its partners may need to send messages of new format or type. This change case also covers the possibility that a new business unit of an existing customer starts sending and receiving messages.

IMPACT: New Partner entry is created for the new customer and for its partners, if the corresponding identifiers do not already exist for the purposes of another customer. The metadata of each partner is added, or updated if a partner exists. The RECEIVE_FILES-node's configuration file is updated. New source level routing rule is added, or InhouseRouterRTE is updated, if a partner sends messages to the customer in a format not known by the IRS. New SRRs and new instances of RPs are created, or InhouseRouterRTE is updated, if the message types used in communication are not defined to the IRS. New processing capabilities are added to the processing layer, if the messages sent to or sent by the new partner need to be translated to a new format or from a new format or if a processing capability for a new message type does not exist.

--**CHANGE CASE 5:** Interface between customer and ISP or between customer's partner and ISP changes

DESCRIPTION: A customer or its partner changes an internal system and the format of messages used in communication changes. These changes are often transparent, since the message format is often not directly produced by the internal system.

IMPACT: The metadata of each partner is updated. New source level routing rule is added, or InhouseRouterRTE is updated, if the partner sends messages to the customer in a format not known by the IRS. New SRRs and new instances of RPs are created, or InhouseRouterRTE is updated, if the message types used in communication are not defined to the IRS. New processing capabilities are added to the processing layer, if the messages sent to or sent by the new partner need to be translated to a new format or from a new

Figure 38. Main change cases and their impact to the IRS.

The FRRs and SRRs, as well as partners, and edisends are maintained through the graphical UI of TX, namely TX Modeler tool. The suggested way of creating the RPs needed in the routing layer is with TX Designer. The use of TX Designer is likely to hasten the programming process by, for instance, providing graphical access to message references. The maintenance of the configuration file used by the RECEIVE_FILES node can be done with any text editor.

4.6 System monitoring, exception handling, performance and security

In the previous sections 4.4.1-4.4.4, the design of the IRS was presented. In order to be a complete solution, it must be defined how the system is monitored and how possible exceptions are detected and handled as they arise. Also system performance and security are issues that need to be considered. In the following sections 4.6.1-4.6.4, the issues of system monitoring, exception handling, performance and security are dissected.

4.6.1 System monitoring

The system is monitored through the Syslog database. Each event during the routing process creates an entry to the Syslog or updates an existing entry in order to ensure proper traceability of the system. The first step of the IRS, which is the RECEIVE_FILES-node, creates an entry to Syslog each time it is executed. To avoid unnecessary logging, RECEIVE_FILES-node removes the previous entry created by the RECEIVE_FILES-node from the Syslog, if it did not retrieve any files. As the message propagates from the FR to the SR and to a RP, another entry is created to Syslog. Only one entry is also created if the FR executes directly the InhouseRouterRTE program, or if the next step after the file router is the OVT IR. By default, each edisend in the processing layer also creates one entry to the Syslog. Table 7 shows example entries the IRS should produce.

Table 7. Example entries, which the IRS should produce to Syslog.

INDEX	CREATED	CONNECTION	DIRECTION	STATUS	TEXT	MORE TEXT
91001	20080805 16:40	RECEIVE_FILES	S	OK		4
91010	20080805 16:41	INHOUSE_RECV	R	OK	91001	
91011	20080805 16:41	INHOUSE_RECV	R	OK	91001	
91012	20080805 16:41	XML_INVOIC_RECV	R	OK	91001	
91013	20080805 16:41	DELFORD97A_RECV	R	OK	91001	
91014	20080805 16:41	INH2ORDERSD93A	S	Sending..	91010	
91015	20080805 16:41	INH2ORDERSD93A	S	OK	91010	
91016	20080805 16:41	INH2INVOIC00A	S	OK	91011	
91017	20080805 16:41	INV2FINVOICE	S	ERROR	91012	
91018	20080805 16:41	SEND_MESSAGE	S	OK	91013	DELFORD97A
91019	20080805 16:42	RECEIVE_FILES	S	OK		7

One of the main objectives of system monitoring is that the route of messages can be traced from the last step of processing to the very first one, that is, from the last edisend to the receiving of files. In order to achieve this traceability, there must always be a reference from the latter entry to the previous one. In the IRS, this reference is Syslog index. RECEIVE_FILES-node adds its Syslog index to the original file name. An RTE routing program receives the index from the file name and writes it to the USERTEXT8 field of its own Syslog entry. It then removes the index from the file name and passes its own Syslog index in parameter IRSINDEX to each edisend it will execute. An edisend reads the parameter value and writes it to the USERTEXT8 field of its own Syslog entry. With the described type of referencing, it is possible to, for instance, verify that each of the messages filtered by an RTE routing program was correctly processed and sent by an edisend of the processing layer. Verifying that every incoming message was processed correctly is another key issue of system monitoring.

The original file name is preserved throughout the routing procedure. With Syslog indexes, the system administrator is able to trace the RECEIVE_FILES entry and with the file name the original file full name that triggered the routing procedure. Of course, overall system monitoring requires more information to be written into Syslog. The name of the TX partner is written to Syslog by each edisend that handles message sending. Additionally, an Edisend may write to Syslog a reference that identifies a particular message or an interchange. With partner names and message references, it is relatively easy to trace individual messages from Syslog. Message sender information is not required by the IRS to make a routing decision. In some messages, this information may not even be present. Of course, the initiator of data transmission is authenticated. However, an ISP must develop a procedure to flawlessly identify the message sender. For instance, if message translation fails in the processing layer due to defective data, a system administrator may need to contact the message sender. This identification may be based, for example, on partner identifiers or some unique piece of data in a message that is sent only by one particular partner. Initially, the sender identification procedure is assumed to be conducted manually, but if a need arises, the IRS may be customized to include also the recognition of message senders.

4.6.2 EXCEPTION HANDLING

The exception handling of IRS is designed to utilize the Alarm facility of TX. The fundamental idea is that when an operation during the routing procedure fails, the corresponding routing or edisend process records an ERROR status to Syslog. The status indicates that a special sending facility, which uses the partner “ALARM”, is to be executed. The “ALARM” partner can be used to transport an error message to or to execute an operating system command (Illicom 2004a, p.105). The operation of the Alarm facility is configured per TX user environment, and is tailored to the needs of the company running TX. Common operation is that the Alarm facility is used to notify a system administrator for the erroneous Syslog entry. For IRS, key is that each step of the IRS routing procedure records an ERROR status, if an exception occurs during its execution. A failure in built-in functions, such as in message sending during an edisend process, automatically results in recording an ERROR status. An RTE program executed by an edisend or routing process must communicate an error using negative exit code.

The IRS routing procedure entails multiple steps, and an exception may occur in any of these steps. A list of main expected exceptions of the IRS is provided in Table 8. Generally, these exceptions are handled in two ways. If an exception occurs due to faulty data, for instance, missing or incorrect recipient identifier or missing mandatory data elements, the correct way to handle the exception is to request the sender to resend the data. If the exception occurs due to other error, for example, fault in the IRS configuration or temporary network failure during message transfer, a system administrator may take corrective action and try to reprocess the data. This is possible, since the original message is always kept in the Syslog database. Additionally, unexpected errors may occur, for instance, due to a system failure. However, handling these unexpected exceptions is outside the scope of IRS design.

Table 8. The main expected exceptions of IRS and their description.

Exception	**Location**	**Description**
Operation scanDirectory fails	ReceiveFilesRTE	ReceiveFilesRTE-program cannot read a directory due to, for instance, insufficient rights. ReceiveFilesRTE returns negative exit code, when all files have been handled. RECEIVE_FILES edisend records ERROR to Syslog.
Operation moveFile fails	ReceiveFilesRTE	Move-command cannot be performed successfully. ReceiveFilesRTE returns negative exit code, when all files have been handled. RECEIVE_FILES Edisend records ERROR to Syslog.
No route for an incoming EDIFACT message	EDIFACT syntax router	FR recognizes format as EDIFACT, but no rule matches to the given message type. EDIFACT SR records ERROR to Syslog.
No route for an incoming XML message	XML syntax router	FR recognizes format as XML, but no rule matches to the given message type. XML SR records ERROR to Syslog.
No route for an incoming OVT message	OVT intermediary router	FR recognizes format as OVT, but IR does not recognize content as EDIFACT. IR records ERROR to Syslog.
No route for an incoming message	File router or InhouseRouterRTE	An incoming message cannot be recognized. InhouseRouterRTE is executed, since its rule “TEXT” matches every text-based message. The incoming message does not match to a single line statement of the InhouseRouterRTE program. Negative exit code is returned to the FR process that called this program. FR also fails, if the incoming message cannot be interpreted as text. FR records ERROR to Syslog.
Recipient identifier not present in message / not found in Partner db	RTE routing program	Message recipient identifier is missing from incoming message or a partner cannot be found in Partner database with the given identifier. Valid messages are routed to processing and invalid left unprocessed. RP returns negative exit code and routing process records ERROR to Syslog.
Processing not found for an incoming message	RTE routing program	Processing cannot be found from partner metadata for an incoming message. Valid messages are routed to processing and invalid left unprocessed. RP returns negative exit code and

		routing process records ERROR to Syslog.
Message processing fails	Processing layer edisend	For example, message translation fails. The processing RTE program returns negative exit code and edisend records ERROR to Syslog.
Message sending fails	Processing layer edisend	Message sending to the target system fails. Edisend process records ERROR to Syslog.

4.6.3 SYSTEM PERFORMANCE

TX performance depends on several factors, for instance, on the underlying hardware and operating system. TX has been proven in real life to handle large message volumes per hour, including message receiving, sending and processing operations. Given that the load caused by the IRS likely generates only a limited portion of total system load, performance issues resulting from heavy load, for instance, Scanner dropping tasks should not arise due to the operation of the IRS. Thus, it is more relevant to analyze the IRS performance through parameters, such as data latency.

The IRS is designed to conform to near-time rather than real-time data requirements. Near-time data generally means that information is not updated instantaneously, but on a reasonable delay or at set intervals (Linthicum 2004, p.48). The IRS introduces two kinds of delays: waiting delays and processing delays. The first waiting delay occurs at RECEIVE_FILES node, which periodically scans directories for files that are “older” than 1 minute. Another waiting delay occurs when the RECEIVE_FILES node has moved the incoming files to the $HOME/incoming/ directory. This waiting delay is similar to the previous one, with the difference that the Scanner process notifies FR as a file arrives to the $HOME/incoming/ directory.

Processing delays of the IRS are caused both by receiving layer’s RPs and by edisends in the processing layer. RPs entail reading the incoming message line by line or segment by segment and matching the input to the line or segment statements. An incoming file may be read through twice, as the routing information is deduced and as messages are filtered. On reasonably sized files, these delays are negligible compared to the waiting delays. The same holds true with processing layer’s edisends. However, on considerably large files the processing delays gain significance. In processing layer, the delay is dependent on factors, including message format and what kind of processing is conducted.

Regardless of waiting and processing delays of IRS, the greatest delays may actually occur at sources, if data is extracted periodically from the source systems rather than in response to an event. This is typically the case when an interchange contains multiple messages. Additionally, considerable delays occur if data is pulled from the source systems periodically rather than source systems push data when they have data to send. For instance, if data is extracted or fetched hourly, the delays of IRS generally lose their significance.

4.6.4 SYSTEM SECURITY

The largest security concerns of the IRS are; how to eliminate the threat of data loss and the threat of messages being routed to incorrect processing and thereby to incorrect target systems. The former threat may occur at RPs, where the original message is filtered to individual messages and processing is started for these. The filtering operation entails reading the original input file line by line, creating an individual output file for each message found in the original file and writing the input lines to these individual output files. Due to a coding error, individual lines or whole messages may not be written to the output files. A similar threat may arise at processing layer’s edisends, where messages are processed. However, this threat is eliminated with proper testing, which should reveal errors in program code. Additionally, the original file is kept in the Syslog database, so it is possible to recover from the error, as long as the abnormal behavior is detected. Otherwise, the system is almost totally immune to data loss.

The latter threat may arise because of faulty partner metadata or a fault in RP logic. For instance, an error in partner’s info file may result in starting a wrong edisend process. Problems may also arise if two partner instances are created with the same identifier in the Partner database. The threats caused by faulty partner metadata should be handled by creating standards for maintenance procedures. A fault in RP logic may result, in turn, passing the wrong file as an input to the sending process. Again, a threat concerning RPs should be eliminated with proper testing.

Security issues related to data communication should also be identified, and handled by mechanisms provided by TX, operating system, data communication software and firewalls. It should be noted that these issues may pose severe threats to the overall service provided to the customers of the service provider. However, handling these security issues is outside the scope of the IRS design.

4.7 Implications on virtual enterprise integration

By implementing the designed IRS to TX, it is possible to route incoming text-based messages to the correct target systems in a generic way, regardless of the message format and type, processing the messages if required. The IRS is likely to simplify the interface between an ISP and its customers, as well as the interface between the ISP and its customers' partners, since it allows these parties to "just" transmit pre-agreed types of messages to TX using any data transmission supported by TX and rely on TX to route them in the right format to the correct target systems. The downside is that each different message type routed by the IRS requires programming a custom RP. However, the amount of needed programming was reduced by identifying the operations common to all RPs and implementing these just once. This amount can most likely to be further reduced by fine-tuning the RP implementation suggested in appendices A and B.

An alternative approach to the problem described in section 4.1 could have been to design a single program that analyzes the content of every incoming message and deduces the necessary information for making a routing decision. However, the size and complexity of this program would grow very large with increasing number of message types, making maintenance of the system difficult. It would also represent a single point of failure. Moreover, this type of solution would neglect the built-in capabilities of TX and forbid the use of message references. Therefore, it can be argued that the solution presented in section 4.4 better conforms to the objectives set in section 4.1 and to the capabilities set in section 4.2.1. More importantly, it can be argued that the designed IRS conforms to the requirements of a generic routing solution.

How do companies benefit from utilizing the services of an ISP, then? How does this strategy potentially affect a company’s preparedness to join a VE? Moreover, what are the implications of the IRS on VE integration? As noted in the beginning of this chapter, a company may purchase integration capabilities from an ISP as a service. In this type of strategy, a company connects to the ISP and lets it take care of the integration issues with its partners transparently. This way, a company may alleviate the technological anxieties related to B2Bi. Through an ISP, new technologies or standards can be more readily adopted by non-IT companies, since the companies need not develop a thorough understanding of these to leverage them. Based on this assumption, a company’s readiness for VE set ups may be improved. However, a company needs to consider how dependent on the services of an ISP it becomes, in case it desires to switch its ISP or to a completely different integration solution.

Another essential benefit of the ISP strategy is that a company does not have to deal with the large initial hardware, software and consulting investments involved in the set up of an integration technology solution, such as integration server. Neither does it have to directly worry about the operating costs that incur when running and administering the integration technology solution. In case of SMEs, the initial investments or running costs often are considerably large compared to the utilization. Additionally, the companies leveraging ISP services need not possess all the required integration expertise in-house. Hence, also SMEs may gain more variety to their B2Bi options through services of an ISP and improve their preparedness for VE set ups.

Of course, outsourcing the B2Bi services involves costs, while the magnitude of these costs depends, among other things, on what services are purchased and on what service level. For instance, an ISP may offer 24/7 service for its customers. This may be valuable for a VE, if its partners conduct activities on multiple time zones. The costs typically consist of elements, such as an initial set up fee, data traffic fees and consulting fees. Moreover, new costs are induced each time the services need to be changed, for instance when customers of an ISP need to integrate with new partners. New waiting delays may also emerge, as there is an intermediary actor between the parties that seek integration. The costs also spiral upwards when the number of partners increases, since the amount of exchanged data grows. Compared to an Integration solution leveraging Internet technologies and hosted by the company itself, the ISP-oriented solution may prove costly, if changes need to be made frequently to the portfolio of purchased services. Thus, the ISP-oriented approach is likely to improve a company's ability to take part in VE set ups, as it has reach to the integration capabilities and expertise of an ISP, but it may also lead to increased transaction costs and thereby inhibit VE reconfiguration dynamics.

The designed IRS reveals some of the basic technical tasks of building an integration solution. These include configuring the data communication interfaces, implementing routing logic and programming message translators. Earlier in chapter 3, it was noted that information needs to be exchanged between back-end applications, between services, as well as between business process execution engine and applications. More specifically, information needs to be exchanged using correct transmission mechanisms and it needs to be sent to the correct systems and in the correct format. Regardless of the integration approach, these are the main tasks that VEs need to optimize to achieve low-cost integration and reduced transaction costs induced by switching.

In the IRS, the change case with minimum reconfiguration time is the one where an ISP can use an existing interface to route messages to a new partner of a customer. In that change case, only minimum changes to the integration server were required. Depending on the required configuration of data communication and security mechanisms, testing procedures and the interface of the receiving party, such connection can be set up quickly. However, this requires compatible interfaces from the partnering companies - standards-based interfaces are not sufficient. On the other extreme, having to specify mappings from a data representation to another, program the required message translators and thoroughly test the resulting solution may require time from several weeks to even months. From the perspective of a VE, this is not acceptable. The aforementioned issues further support the suggestion for the need of pre-formation collaborative work of potential VE participants. By having compatible, as well as ready-made, interfaces, VEs may achieve effective, low-cost integration that supports dynamic reconfiguration.

5. CONCLUDING REMARKS

In this thesis, virtual enterprise has been identified as a distinct collaborative organizational form. A VE is dynamic network consisting of independent companies, linked by modern ICT to exploit a limited business opportunity. It can be seen as an evolutionary step from dynamic network, and its most recent instance is agile virtual enterprise. The incentives for creating or joining a VE have been grouped to internal and external. Some of the main internal incentives are gaining access to partners markets and increasing flexibility. A key external motivation is, in turn, coping with the turbulent markets and shortened product life cycles. Moreover, the incentives are seen valid for both SMEs and larger companies.

A VE is typically characterized by cheap, as well as opportunistic, formation, limited existence, building upon core competences, extensive use of ICT, reconfiguration dynamics and flexible company interfaces. Additionally, a VE generally undergoes a distinct life cycle from creation and configuration, to operation and eventual dissolution. The main operations of the first stage include partners search and selection, contract negotiations, setting VE strategy and governance principles, designing cross-company business processes and configuring the VE ICT infrastructure. To ease partners search and enable fast formation of a VE, the necessity of external entities has been evaluated. External entities, such as VBEs, alleviate the formation issues that arise from lack of preparedness, as well as heterogeneity, of partners, by enforcing preliminary cooperation arrangements. These arrangements include developing basis for common interoperable infrastructures and ontologies. During operation stage, a VE executes the designed business processes in order to meet its business objectives. As reconfiguration dynamics characteristic suggests, partners can be added and removed during operation. However, a VE must compare the potential benefits of reconfiguration to the incurring transaction costs. Finally, a VE is dissolved when the business objectives are met or when it becomes clear that the objectives cannot be met. The dissolution may also be only partial, if the VE business venture detects permanent market potential or if some of the companies need to stay committed to the VE in order to take care of the after sales service. Therefore, it has been suggested that dissolution is only a special case of dynamic reconfiguration.

As stressed in this thesis, the use of modern ICT is essential for VEs. ICT provides the necessary link between VE companies, for instance, by enabling companies to exchange technical and business information. In other words, it enables integration between VE partners on a technological level. VE ICT infrastructure should be flexible in order to support shareability, as well as reuse, of components and adaptation to new business processes. Moreover, it should support dynamic reconfiguration, by allowing individual components be added and removed without breaking the system. The challenge of VE ICT infrastructure configuration is formidable. This is a result of the requirements that VE operation sets on infrastructures, but also of heterogeneity of partners, and time constraints. Additionally, the lack of a widely accepted reference infrastructure in this domain is forcing VEs to invest resources in developing their infrastructures from scratch. Given the requirements that VEs set on their infrastructures, it is not surprising that many VE research projects have sought a solution to the VE ICT infrastructure configuration issues from state-of-the-art technologies, including semantic Web services and agent technologies. However, companies' insufficient willingness and readiness to adopt such technologies partly inhibit the efforts of including these technologies as a part of VE ICT infrastructures.

VEs have also been identified in this thesis to set peculiar requirements for B2Bi. Fast and cheap formation, frequent partner network reconfiguration and limited existence are not issues that a typical B2Bi project runs into. In case of VEs, coping with considerable heterogeneity is also likely to be required. Moreover, VEs may require inter-company integration in four different levels: safe transactions, applications, business processes and professional teams, of which the first three levels fall into the domain of B2Bi. A solution aimed to enable VE integration should also provide support for the entire VE life cycle, including partners search and selection. To enable dynamic reconfiguration of a VE, changing business processes and partners should be supported. Additionally, the solution should be self-annealing. To further complicate things, the VE creation and changes should be made with minimum transaction costs. However, a VE may only require integration in a specific context, representing only a few of all the possible processes, which mitigates the complexity of VE integration.

B2Bi aims to integrate applications, systems and business processes between different companies. It fundamentally tries to solve the issues that arise from heterogeneity of partner's application semantics, information content and platforms. Several B2Bi approaches, including IOI, SOI, BPOI and POI have been developed to achieve proper integration solutions between business partners. Moreover, the integration solutions consist of multiple components, such as EAI, Web services, BPMS, middleware, standards and security solutions. In this thesis, consideration has been put on how these correspond to the requirements set by VEs. By using IOI, SOI, and BPOI, or a combination of these, it is possible to build integration solutions that conform to the first three levels of integration required by VEs. Additionally, a VE may leverage POI to build a customer interface. However, this approach does not allow end-to-end automation, which has been identified as a key driver for B2Bi.

EAI was seen to influence companies' readiness to join VEs, since cross-organizational business processes may require information simultaneously from multiple systems. Therefore, it is necessary to have an unconstrained information flow across a company's internal systems. Another B2Bi component, namely Web services, was considered particularly interesting for VEs, since Web services promote low-cost, as well as loosely coupled, composition of distributed applications and accelerated integration. Moreover, SOA involves recording information of available services in a registry. The possibility to leverage service registries to benefit VE partner search and selection was also contemplated. However, public registries are not currently available. Moreover, it was questioned whether necessary capabilities can be represented as services and described by, for instance WSDL.

BPMSs were identified as an important component of BPOI. They allow graphical modeling of business processes and connecting the systems involved in the process model. In VE context, the lack of widely accepted business process modeling standards was seen as possible inhibitor of VE BPOI. Standards, in turn, generally provide a foundation to build B2Bi solutions. Fundamentally, standards mitigate the issues that arise from heterogeneity of partners' information semantics. Some standards, including RosettaNet, also aim to interface certain business processes. RosettaNet, as well as other similar vertical standards, may prove valuable for VE integration, if companies within some specific industry already share the same standardized information and process interfaces. The problem with B2Bi standards is that they leave room for variation and generally are only as good as the number of companies using them. A B2Bi component typically needed to bridge the heterogeneity between partners' systems is middleware. In this thesis, integration servers were suggested to improve companies' preparedness to join VEs, since they offer capabilities for different types of data communication, intelligent routing, data transformation and standards conformance. However, data transformation often requires creating a translator program, which causes delays for integration projects. Moreover, configuration of security solutions was seen as a possible cause for delays in VE integration and, thus cause for transaction costs. Delays may also occur as a result of insufficient integration planning and improper testing, which were identified as risks of B2Bi.

An evident trend in the B2Bi domain is movement towards services, SOA and SOI. From the perspective of VEs, this trend is interesting, since SOI approach is founded on open standards, it promotes loose coupling between components, standard service interfaces, maintaining a registry of services and composing services into distributed business processes. Additionally, the solutions based on SOI should be realizable at a relatively low cost. As such, services do not solve the issues that result from heterogeneity of application semantics. This is one of the main reasons why services are, and need to be paired with B2Bi standards or semantic Web technologies. While current integration approaches and technologies enable integration between companies, semantic Web technologies are eventually expected to enable dynamic integration. At present, semantic Web technologies do not have a role in B2Bi.

One approach to B2Bi is to outsource the integration services to an ISP. In this type of an approach, a company makes a single logical connection to the ISP and allows it to handle the B2Bi issues transparently with its partners. This way, a company does not have to bear the considerable costs involved in the set up of an integration technology solution. Neither does it have to possess all the required integration expertise in-house. In this thesis, it has been suggested that this type of strategy may enhance a company's preparedness for VEs, by making it more susceptible to new standards, as well as technologies, required by VE operation and by having access to the integration capabilities and expertise of an ISP. However, the ISP-oriented approach may introduce considerable transaction costs, if the portfolio of purchased services needs to be changed frequently. Additionally, the running costs need to be evaluated when choosing this type of approach to B2Bi.

So far, the requirements that VEs set on B2Bi and how the B2Bi approaches, as well as B2Bi components, conform to these requirements have been analyzed in this thesis. Moreover, the ISP-oriented approach to B2Bi has been discussed in light of VEs. But, what is the effect of VEs on B2Bi, then? Modern, dynamic inter-company collaborations, such as VEs, clearly require flexibility from B2Bi solutions. As discussed, B2Bi fundamentally aims at integrating business partners' applications and systems, as well as automating the business processes occurring between them. The efforts to overcome the three barriers that arise from the heterogeneity of partners' application semantics, information content and platforms have been so formidable that less attention has been given to the costs involved in achieving this integration and automation. It has been expected that B2Bi relationships remain stable, that B2Bi costs are distributed on a longer time span and that benefits of automation supersede the incurring costs. VEs seek for rapid and low-cost formation that includes integrating the partners' ICT infrastructures, with the intent of changing partners during operation and dissolving after the business objectives are met. Currently, there is no such B2Bi approach or technology that could provide an out-of-the-box solution for VE integration. All in all, VEs imply that B2Bi's focus needs to be shifted from integration to dynamic integration. This most evidently is the effect of virtual enterprises to business-to-business integration.

6. BIBLIOGRAPHY

Aerts, A.T.M., Szirbik, N.B. & Goossenaerts, J.B.M., 2002. A flexible, agent-based ICT architecture for virtual enterprises. *Computers in Industry* [online], 49(3), pp.311-327. Available from: http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B6V2D-475RHFF-1&_user=952938&_rdoc=1&_fmt=&_orig=search&_sort=d&view=c&_acct=C000049220&_version=1&_urlVersion=0&_userid=952938&md5=75f645dfe13852d9821bb40abbac537b [Accessed 19 Nov 2008].

Alonso, G., Casati, F., Kuno, H. & Machiraju, V., 2004. *Web Services: Concepts, Architectures and Applications.* Berlin: Springer-Verlag.

Baeyens, T. &Fricke, P, 2006. *Bringing BPM to the Mass Market* [online]. (Updated 20 Nov 2008). Available from: http://www.alignjournal.com/index.cfm?section=article&aid=264 [Accessed 20 Nov 2008].

Broadbent, M., Weill, P. & Neo, B.S., 1999. Strategic context and patterns of IT infrastructure capability. *The Journal of Strategic Information Systems* [online], 8(2), pp. 157-187. Available from: http://web.ebscohost.com/ehost/detail?vid=1&hid=114&sid=09e8463c-085a-4791-9fe2-6841645b40d4%40sessionmgr103&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#db=buh&AN=9601071460 [Accessed 19 Nov 2008].

Bussler, C., 2002a. The Application of Workflow Technology in Semantic B2B Integration. *Distributed and Parallel Databases* [online], 12(2-3), pp. 163-191. Available from: http://www.springerlink.com/content/u548x4l9w4348550/ [Accessed 19 Nov 2008].

Bussler, C., 2002b. P2P in B2BI. *System Sciences, 2002, HICSS, Proceedings of the 35th Annual Hawaii International Conference* [online], pp.3915- 3924. Available from: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=994528 [Accessed 19 Nov 2008].

Byrne, J.A., 1993. The Virtual Corporation. *International Business Week*, 8 February, pp. 36-41.

Camarinha-Matos, L.M. & Afsarmanesh, H., 2003. Elements of a base VE infrastructure. *Computers in Industry* [online], 51(2), pp. 139-163. Available from: http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B6V2D-48V84GV-3&_user=952938&_coverDate=06%2F30%2F2003&_rdoc=1&_fmt=&_orig=search&_sort=d&view=c&_acct=C000049220&_version=1&_urlVersion=0&_userid=952938&md5=d5baa39d6038b0408ecf1cfbeac9f685 [Accessed 19 Nov 2008].

Camarinha-Matos, L.M. & Afsarmanesh, H., 2001. Virtual Enterprise Modeling and Support Infrastructures: Applying Multi-agent System Approaches. *In:* Luck, M., Marik, V., Trappl, R., eds. *Multi-Agent Systems and Applications* [online]. Heidelberg / Berlin: Springer, pp.335-364. Available from:

http://www.springerlink.com/content/e42e4xhcvyvy2x2h [Accessed 19 Nov 2008].

Camarinha-Matos, L. & Afsarmanesh, H., 2007. A framework for virtual organization creation in a breeding environment. *Annual Reviews in Control* [online], 31(1), pp. 119-135. Available from: http://www.sciencedirect.com/science/article/B6V0H-4NH6NGN-1/2/4881790bd3c08b41b3948d0587cbb89f [Accessed 19 Nov 2008].

Cardoso, H.L. & Oliveira, E., 2005. Virtual Enterprise Normative Framework Within Electronic Institutions. . *In:* Gleizes, M-P., Omicini, A. & Zambonelli, F., eds. *Engineering Societies in the Agents World V* [online]. Heidelberg / Berlin: Springer, pp.14-32. Available from: http://www.springerlink.com/content/9xq634uypnra88fn [Accessed 19 Nov 2008].

Carvalho, J.D.A., Moreira, N.A. & Pires, L.C.M., 2005. Autonomous Production Systems in virtual enterprises. *International Journal of Computer Integrated Manufacturing* [online], 18(5), pp. 357-366. Available from: http://search.ebscohost.com/login.aspx?direct=true&db=buh&AN=18636061&site=ehost-live [Accessed 19 Nov 2008]

Chalmeta, R. & Grangel, R., 2003. ARDIN extension for virtual enterprise integration. *Journal of Systems and Software* [online], 67(3), pp. 141-152. Available from: http://www.sciencedirect.com/science/article/B6V0N-48GVXGR-2/2/db9a6c09a3d8d0d9dc03bfaa7072c15b [Accessed 19 Nov 2008].

Chandran, A., Moutayakine, D., Ulmer, T., Pal, M., Dodani, M., Craig, G., Cutlip, R., Rowe, R. & Smith, D.R.H., 2003. *Architecting Portal Solutions* [online]. IBM Redbooks. Available from: http://www.redbooks.ibm.com/abstracts/sg247011.html?Open [Accessed 20 Nov 2008].

Charlesworth, I. & Jones, T., 2003. The Web Services and EAI Report. *eAI Journal* [online], *19* March, pp.11-18. Available from: http://www.alignjournal.com/index.cfm?section=article&aid=182# [Accessed 20 Nov 2008].

Chesbrough, H.W. & Teece, D.J., 2002. Organizing for Innovation: When is Virtual Virtuous. *Harward Business Review,* 80(8), pp. 127-134.

Clarke, I. & Flaherty, T.B., 2003. Web-based B2B portals. *Industrial Marketing Management* [online], 32(1), pp. 15-23. Available from: http://www.sciencedirect.com/science/article/B6V69-47N56W5-3/2/2ca62dbfe487d115e1d8653a7f490e47 [Accessed 20 Nov 2008].

Cunha, M.M. and Putnik, G., 2006. *Agile Virtual Enterprises: Implementation and Management Support* [online]. Hershey, PA: Idea Group Publishing. Available from: http://site.ebrary.com/lib/otaniemi/Top?channelName=otaniemi&cpage=1&docID=10130337&f00=text&frm=smp.x&hitsPerPage=20&layout=document&p00=agile+virtual+enterprise&sch=sch&sch.x=0&sch.y=0&sortBy=score&sortOrder=desc [Accessed 19 Nov 2008].

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N. & Weerawarana, S., 2002. Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. *Internet Computing, IEEE* [online], 6(2), pp.86-93. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=991449&isnumber=21386 [Accessed 20 Nov 2008].

Davenport, T.H., 1998. Putting the Enterprise into the Enterprise System. *Harvard Business Review* [online], 76(4), pp. 121-131. Available from: http://web.ebscohost.com/ehost/detail?vid=1&hid=114&sid=76977af8-3017-42c1-a41a-683e2180bf02%40sessionmgr108&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#db=buh&AN=780261 [Accessed 20 Nov 2008].

Duncan, N.B., 1995. Capturing Flexibility of Information Technology Infrastructure: A Study of Resource Characteristics and their Measure. *Journal of Management Information Systems* [online], 12(2), pp. 37-57. Available from: http://web.ebscohost.com/ehost/detail?vid=1&hid=114&sid=09e8463c-085a-4791-9fe2-6841645b40d4%40sessionmgr103&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#db=buh&AN=9601071460 [Accessed 19 Nov 2008].

Erasala, N., Yen, D.C. & Rajkumar, T.M., 2003. Enterprise Application Integration in the electronic commerce world. *Computer Standards & Interfaces* [online], 25(2), pp. 69-82. Available from: http://www.sciencedirect.com/science/article/B6TYV-47602C2-1/2/b58989dcfc11b10d6f4633de9e5356cb [Accessed 20 Nov 2008].

Fitzgerald, M., 2001. *Building B2B Applications with XML: A resource guide* [online]. New York: John Wiley & Sons, Inc. Available from: http://site.ebrary.com/lib/otaniemi/Top?channelName=otaniemi&cpage=2&docID=10001744&f00=text&frm=smp.x&hitsPerPage=10&layout=document&p00=b2b+integration+security&sch=sch&sch.x=0&sch.y=0&sortBy=score&sortOrder=desc [Accessed 20 Nov 2008].

Foster, I., Kesselman, C., Nick, J.M. & Tuecke, S., 2002. Grid services for distributed system integration**.** *Computer* [online], 35(6), pp.37-46. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1009167&isnumber=21753 [Accessed 21 Nov 2008].

Franke, U.J., ed, 2002. *Managing Virtual Web Organizations in the 21st Century: Issues and Challenges* [online]. Hershey, PA: Idea Group Publishing. Available from: http://site.ebrary.com/lib/otaniemi/Top?channelName=otaniemi&cpage=1&docID=10019263&f00=text&frm=smp.x&hitsPerPage=20&layout=document&p00=franke&sch=sch&sch.x=0&sch.y=0&sortBy=score&sortOrder=desc [Accessed 19 Nov 2008].

Gang, C., Zhang, J.B., Low, C.P.,Yang, Z. & Zhuang, L., 2007. Collaborative Virtual Enterprise Integration via Semantic Web Service Composition. *Industrial Electronics and Applications, 2007, ICIEA 2007* [online], pp.1409-1414. Available from: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4318638 [Accessed 19 Nov 2008].

Goranson, H.T., 2003. Architectural support for the advanced virtual enterprise. *Computers in Industry* [online], 51(2), pp. 113-125. Available from: http://www.sciencedirect.com/science/article/B6V2D-48PDGPN-2/2/7a9fcf52e83e08e687143300fd0630da [Accessed 19 Nov 2008].

Goranson, H.T., 2005. Semantic Distance, the Next Step? *In:* Putnik, G., Cunha, M.M., eds. *Virtual Enterprise Integration: Technological and Organizational Perspectives* [online]. Hershey, PA: Idea Group Publishing, pp.283-296. Available from: http://site.ebrary.com/lib/otaniemi/Top?channelName=otaniemi&cpage=1&docID=10077328&f00=text&frm=smp.x&hitsPerPage=20&layout=document&p00=Virtual+enterprise+integration+&sch=%A0%A0%A0%A0%A0Search%A0%A0%A0%A0%A0&sortBy=score&sortOrder=desc [Accessed 19 Nov 2008].

Gottschalk, K., Graham, S.H. & Snell, J., 2002. Introduction to Web services architecture. *IBM Systems Journal* [online], 41(2), pp. 170-177. Available from: http://researchweb.watson.ibm.com/journal/sj/412/gottschalk.pdf [Accessed 20 Nov 2008].

Gou, H., Huang, B., Liu, W. & Li, X., 2003. A framework for virtual enterprise operation management. *Computers in Industry* [online], 50(3), pp. 333-352. Available from: http://www.sciencedirect.com/science/article/B6V2D-48GNX19-1/2/3dc58d581e59172ec4ad1a903bdbc175 [Accessed 19 Nov 2008].

Grönroos, M., 2003. *Mahdollisuuden aika - kohti virtuaalista organisaatiota.* Tampere: Transatlanta Oy.

Gruden, A. & Strannegard, P., 2003. Business Process Integration: The Next Wave. *eAI Journal* [online], pp. 8-12. Available from: http://www.bijonline.com/index.cfm?section=article&aid=664# [Accessed 14 Jan 2008].

Gulati, R., Nohria, N. & Zaheer, A. 2000. Strategic Networks. *Strategic Management Journal* [online], 21(3), pp. 203-215. Available from: http://www3.interscience.wiley.com/cgi-bin/fulltext/71001342/PDFSTART [Accessed 19 Nov 2008].

Gulledge, T., 2006. What is integration? *Industrial Management & Data Systems* [online], 106(1), pp.5-20. Available from: http://www.emeraldinsight.com/10.1108/02635570610640979 [Accessed 20 Nov 2008].

Haas, L., 2007. Beauty and the Beast: The Theory and Practice of Information Integration. *In:* Schwentick, T. & Suciu, D. eds. *Database Theory – ICDT 2007* [online]. Heidelberg / Berlin: Springer, pp. 28-43. Available from: http://www.springerlink.com/content/e42e4xhcvyvy2x2h [Accessed 19 Nov 2008].

Haas, L.M., Lin, E.T. & Roth, M.A., 2002. Data integration through database federation. *IBM Systems Journal* [online], 41(4), pp. 578-596. Available from: http://web.ebscohost.com/ehost/detail?vid=1&hid=113&sid=3fb0cd9b-de34-424f-8154-5aebc9f9c080%40sessionmgr104&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#db=buh&AN=7928970 [Accessed 19 Nov 2008].

Haller, A., Kotinurmi, P., Vitvar, T. & Eyal, O., 2007. Handling heterogeneity in RosettaNet messages. *Symposium on Applied Computing: Proceedings of the 2007 ACM symposium on Applied computing* [online], pp.1368 - 1374. Available from: http://portal.acm.org/citation.cfm?id=1244297&coll=GUIDE&dl=GUIDE&CFID=9062365&CFTOKEN=19925734&ret=1#Fulltext [Accessed 20 Nov 2008].

Hardwick, M., Spooner, D.L., Rando, T. & Morris, K.C., 1996. Sharing Manufacturing Information in Virtual Enterprises. *Communications of the ACM* [online], 39(2), pp.46-54. Available from: http://doi.acm.org/10.1145/230798.230803 [Accessed 19 Nov 2008]

Hardwick, M. & Bolton, R. 1997. The industrial virtual enterprise. *Communications of the ACM* [online], 40(9), pp. 59-60. Available from: http://portal.acm.org/citation.cfm?id=260750.260770 [Accessed 19 Nov 2008].

Hao, G. & Hao, F., 2007. Application of Grid Technology in eBusiness. *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on 21-25 Sept. 2007* [online], pp.3442-3445. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4340626&isnumber=4339775 [Accessed 21 Nov 2008].

IBM, 2008. *Report: Worldwide, IBM Is Well Ahead of Traditional Competitors in SOA Consulting Services* [online]. (Updated 21 Nov 2008). Available from: http://www-03.ibm.com/press/us/en/pressrelease/23265.wss [Accessed 21 Nov 2008].

Illicom, 2004a. *TradeXpress Administrator's Guide for UNIX*.

Illicom, 2004b. *TradeXpress RTE reference*.

Influe-Illicom, 2007. *TradeXpress Enterprise User's Guide*.

Jax Magazine, 2007. *Fresh Brew: SOA and BPM Converging, Says Forrester*. (Updated 21 Nov 2008). Available from: http://www.jaxmag.com/itr/news/psecom,id,33038,nodeid,146.html [Accessed 21 Nov 2008].

Jhingran, A.D., Mattos, N. & Pirahesh, H., 2002. Information integration: A research agenda. *IBM Systems Journal* [online], 41(4), pp. 555-562. Available from: http://web.ebscohost.com/ehost/detail?vid=1&hid=7&sid=e97577e9-e4a0-4a43-887c-eff32eb9f8dd%40sessionmgr2&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#db=buh&AN=7928968 [Accessed 19 Nov 2008].

Kanet, J.J., Faisst, W. & Mertens, P., 1999. Application of information technology to a virtual enterprise broker: The case of Bill Epstein. *International Journal of Production Economics* [online], 62(1-2), pp. 23-32. Available from: http://www.sciencedirect.com/science/article/B6VF8-3WG359G-3/2/2822da65f2466f5048e9bca2fc814dd9 [Accessed 19 Nov 2008].

Katzy, B. & Löh, H. 2003. Virtual Enterprise Research State of the Art and Ways Forward. *CeTIM* [online]. Available from:

http://portal.cetim.org/file/1/63/104_Katzy_Loehprint.pdf [Accessed 19 Nov 2008].

Keen, M., Acharya, A., Bishop, S., Hopkins, A., Milinski, S., Nott, C., Robinson, R., Adams, J. & Verschueren, P., 2004. *Patterns: Implementing an SOA using an Enterprise Service Bus* [online]. IBM Redbooks. Available from: http://www.redbooks.ibm.com/abstracts/sg246346.html?Open [Accessed 21 Nov 2008].

Khoshafian, S., 2002. *Web Services and Virtual Enterprises* [online]. Tect. Available from: http://www.webservicesarchitect.com/content/articles/WSAVEKHOSHAFIAN220502.pdf [Accessed 20 Nov 2008].

Kimmel, P., 2005. *UML Demystified* [online]. Emeryville, CA: McGraw-Hill Osborne. Available from: http://site.ebrary.com/lib/otaniemi/Top?channelName=otaniemi&cpage=1&docID=10156015&f00=text&frm=smp.x&hitsPerPage=20&layout=document&p00=paul+kimmel&sch=sch&sch.x=0&sch.y=0&sortBy=score&sortOrder=desc [Accessed 21 Nov 2008].

Kotinurmi, P., 2007. *E-Business Framework Enabled B2B Integration* [online]. Dissertation (D. Tech.). Available from: http://lib.tkk.fi/Diss/2007/isbn9789512289929/ [Accessed 20 Nov 2008].

Lam, W. & Shankararaman, V., 2004. An enterprise integration methodology. *IT Professional* [online], 6(2), pp.40-48. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1278864&isnumber=28563 [Accessed 19 Nov 2008].

Lee, J., Siau, K. & Hong, S., 2003. Enterprise integration with ERP and EAI. *Communications of the ACM* [online], 46(2), pp. 54-60. Available from: http://doi.acm.org/10.1145/606272.606273 [Accessed 20 Nov 2008].

Leymann, F., Roller, D. & Schmidt, M., 2002. Web services and business process management. *IBM Systems Journal* [online], 41(2), pp. 198-211. Available from: http://search.ebscohost.com/login.aspx?direct=true&db=buh&AN=6580311&site=ehost-live [Accessed 20 Nov 2008].

Linthicum, D.S., 2004. *Next Generation Application Integration: From Simple Information to Web Services.* Boston, MA: Pearson Education, Inc.

Mahmoud, Q.H., 2005. *Service-Oriented Architecture (SOA) and Web Services: The Road to Enterprise Application Integration (EAI)* [online]. Available from: http://java.sun.com/developer/technicalArticles/WebServices/soa/ [Accessed 20 Nov 2008].

Martin, D., Burstein, M., McDermott, D., McIlraith, S., Paolucci, M., Sycara, K., McGuinness, D.L., Sirin, E. & Srinivasan, N., 2007. Bringing Semantics to Web Services with OWL-S. *World Wide Web [online],* 10(3), pp. 243-277. Available from: http://www.springerlink.com/content/wp8q2133g5725340/ [Accessed 20 Nov 2008].

Meadors, K., 2005. Secure electronic data interchange over the Internet. *Internet Computing, IEEE* [online], 9(3), pp. 82-89. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1438309&isnumber=30969 [Accessed 20 Nov 2008].

Medjahed, B., Benatallah, B., Bouguettaya, A., Ngu, A.H.H. & Elmagarmid, A.K., 2003. Business-to-business interactions: issues and enabling technologies. *The VLDB Journal* [online], 12(1), pp. 59-85. Available from: http://www.springerlink.com/content/659ll5tlgretcpe1/ [Accessed 20 Nov 2008].

Miles, R.E. & Snow, C.C., 1986. Organizations: New Concepts for New Form. *California Management Review* [online], 28(3), pp.62-73. Available from: http://web.ebscohost.com/ehost/detail?vid=1&hid=109&sid=68484736-6261-44e5-81a8-9e7225e5b507%40sessionmgr102&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#db=buh&AN=4762576 [Accessed 19 Nov 2008].

Mowshowitz, A., 2002. *Virtual Organization: Toward a Theory of Societal Transformation Stimulated by Information Technology.* Westport, CT: Quorum Books.

Nayak, N., Bhaskaran, K. & Das, R., 2001. Virtual enterprises-building blocks for dynamic e-business. *Australian Computer Science Communications* [online], 23(6), 80-87. Available from: http://doi.acm.org/10.1145/545617.545630 [Accessed 19 Nov 2008].

OASIS, 2008. *OASIS Universal Business Language (UBL) TC* [online]. (Updated 21 Nov 2008). Available from: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ubl [Accessed 21 Nov 2008].

O'Connel, M. & Nixon, P., 2000. Next Generation Business-to-Business E-Commerce. *In:* Bauknecht, K.,Madria, S.K. & Pernul, G., eds. *Electronic commerce and Web Technologies* [online]. Heidelberg / Berlin: Springer, pp.452-465. Available from: http://www.springerlink.com/content/vgxka13ne6q9b2le/ [Accessed 19 Nov 2008].

Papazoglou, M.P., 2003. Service-oriented computing: concepts, characteristics and directions. *Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference* [online], pp.3-12. Available from: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1254461 [Accessed 20 Nov 2008]

Papazoglou, M.P. & Van den Heuvel, W-J., 2007. Service oriented architectures: approaches, technologies and research issues. *The VLDB Journal* [online], 16(3), pp. 389-415. Available from: http://www.springerlink.com/content/d2228066725332u2/ [Accessed 20 Nov 2008].

Papazoglou, M.P., Traverso, P., Dustdar, S. & Leymann, F., 2007. Service-Oriented Computing: State of the Art and Research Challenges. *Computer* [online], 40(11), pp.38-45. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4385255&isnumber=4385238 [Accessed 19 Nov 2008]

Pereira Klen, A. A., Rabelo, R.J., Campos Ferreira, A. & Spinosa, L.M., 2001. Managing distributed business processes in the virtual enterprise. *Journal of intelligent manufacturing* [online], 12(2), pp. 185-197. Available from: http://proquest.umi.com/pqdlink?Ver=1&Exp=03-03-2013&FMT=7&DID=352547721&RQT=309 [Accessed 19 Nov 2008].

Preist, C., Esplugas-Cuadrado, J., Battle, S.A., Grimm, S. & Williams, S.K., 2005. Automated Business-to-Business Integration of a Logistics Supply Chain Using Semantic Web Services Technology. *In: Gil, Y., Motta, E., Benjamins, R.J. & Musen, M.A., eds. The Semantic Web –ISWC 2005* [online]. Heidelberg / Berlin: Springer, pp.987-1001. Available from: http://www.springerlink.com/content/p0gg6464464q8060/ [Accessed 20 Nov 2008].

Putnik, G., Cunha, M.M., Sousa R. & Avila, P., 2005. Virtual Enterprise Integration: Challenges of a New Paradigm. *In:* Putnik, G., Cunha, M.M., eds. *Virtual Enterprise Integration: Technological and Organizational Perspectives* [online]. Hershey, PA: Idea Group Publishing, pp. 1-30. Available from: http://site.ebrary.com/lib/otaniemi/Top?channelName=otaniemi&cpage=1&docID=10077328&f00=text&frm=smp.x&hitsPerPage=20&layout=document&p00=Virtual+enterprise+integration+&sch=%A0%A0%A0%A0%A0Search%A0%A0%A0%A0%A0&sortBy=score&sortOrder=desc [Accessed 19 Nov 2008].

Ratnasingam, P. & Phan, D.D., 2003. Trading Partner Trust in B2b E-Commerce: a Case Study. *Information Systems Management* [online], 20(3), pp. 39-50. Available from: http://search.ebscohost.com/login.aspx?direct=true&db=buh&AN=9809965&site=ehost-live [Accessed 20 Nov 2008].

Rocha, A.P., Cardoso, H.L. & Oliveira, E., 2005. Contributions to an Electronic Institution Supporting Virtual Enterprises' Life Cycle. *In:* Putnik, G., Cunha, M.M., eds. *Virtual Enterprise Integration: Technological and Organizational Perspectives* [online]. Hershey, PA: Idea Group Publishing, pp.229-246. Available from: http://site.ebrary.com/lib/otaniemi/Top?channelName=otaniemi&cpage=1&docID=10077328&f00=text&frm=smp.x&hitsPerPage=20&layout=document&p00=Virtual+enterprise+integration+&sch=%A0%A0%A0%A0%A0Search%A0%A0%A0%A0%A0&sortBy=score&sortOrder=desc [Accessed 19 Nov 2008].

Rosado, D.G., Gutierrez, C., Fernandez-Medina, E. & Piattini, M., 2006. Security patterns and requirements for Internet-based applications. *Internet Research* [online], 16(5), pp. 519-536. Available from: [Accessed 20 Nov 2008].

Rosenberg, F. 2005. Business rules integration in BPEL - a service-oriented approach*. E-Commerce Technology, 2005. CEC 2005. Seventh IEEE International Conference on 19-22 July 2005* [online], pp.476-479. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1524091&isnumber=32584 [Accessed 20 Nov 2008].

RosettaNet, 2008. *RosettaNet standards* [online]. (Updated 20 Nov 2008). Available from: http://www.rosettanet.org/cms/sites/RosettaNet/Standards/RStandards/index.html [Accessed 20 Nov 2008].

Roy, J. & Ramanujan, A., 2001. Understanding Web services. *IT Professional* [online], 3(6), pp.69-73. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=977775&isnumber=21082 [Accessed 20 Nov 2008].

Salam, A.F. & Stevens, J.R., eds, 2007. *Semantic Web Technologies and E-Business: Toward the Integrated Virtual Organization and Business Process Automation* [online], Hershey, PA: Idea Group Publishing.

Samtani, G., 2002. *B2B Integration: A Practical Guide to Collaborative E-Commerce. London:* Imperial College Press.

SAP, 2008*. SAP Exchange Infrastructure* [online]. (Updated 16 Apr 2008). Available from: http://help.sap.com/saphelp_nw04/helpdata/en/14/80243b4a66ae0ce10000000a11402f/frameset.htm [Accessed 20 Nov 2008].

Schmidt, M-T., Hutchison, B., Lambros, P. & Phippen, R., 2005. The Enterprise Service Bus: Making service-oriented architecture real. *IBM Systems Journal* [online], 44(4), pp. 781-797. Available from: http://search.ebscohost.com/login.aspx?direct=true&db=buh&AN=19086838&site=ehost-live [Accessed 20 Nov 2008].

Shen, W., Hao, Q., Wang, S., Li, Y. & Ghenniwa, H. 2007. An agent-based service-oriented integration architecture for collaborative intelligent manufacturing. *Robotics and Computer-Integrated Manufacturing* [online], 23(3), pp. 315-325. Available from: http://www.sciencedirect.com/science/article/B6V4P-4JKRTNP-2/2/cab637378a3d76a25bef8f07c6013f3f [Accessed 19 Nov 2008].

Stonebraker, M., 2002. Too much middleware. ACM *SIGMOD Records* [online], 31(1), pp. 97-106. Available from: http://doi.acm.org/10.1145/507338.507362 [Accessed 20 Nov 2008].

Tatsiopoulos, I.P., Ponis, S.T., Hadzilias, E.A. & Panayiotou, N.A., 2002. Realization of the Virtual Enterprise Paradigm in the Clothing Industry through E-Business Technology. *Production & Operations Management* [online], 11(4), pp. 516-530. Available from: http://web.ebscohost.com/ehost/detail?vid=1&hid=104&sid=7c13b587-0c56-4b86-9cd4-f145b7537631%40sessionmgr102&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#db=buh&AN=15753565 [Accessed 19 Nov 2008]

TIEKE, 2008. *UBL-Sanomaprojektit* [online]. (Updated 21 Nov 2008). Available from: http://www.tieke.fi/kehityshankkeet/ubl-sanomaprojektit/ [Accessed 21 Nov 2008].

Tsai, T-M., Huang, W-S., Chang, C-C., Wu, F-T. & Chou, S-C.T., 2006. eXFlow: A Web Services-compliant system for supporting B2B process integration. *Information Systems and E-Business Management* [online], 5(1), pp. 47-64. Available from: http://www.springerlink.com/content/x77416634762866n/ [Accessed 20 Nov 2008].

UDDI Consortium, 2001. *UDDI Executive White Paper* [online]. (Updated 2 Nov 2006). Available from: http://uddi.org/pubs/ [Accessed 20 Nov 2008].

Umar, A. & Missier, P., 1999. A framework for analyzing virtual enterprise infrastructure. *Research Issues on Data Engineering: Information Technology for Virtual Enterprise, RIDE-VE '99, Proceedings* [online], pp. 4-11. Available from: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=758584 [Accessed 19 Nov 2008].

Vallejos, R.V., Lima, C.P. & Varvakis, G., 2007. Towards the development of a framework to create a virtual organisation breeding environment in the mould and die sector. *Journal of Intelligent Manufacturing* [online], 18(5), pp. 587-597. Available from: http://www.springerlink.com/content/tl8n372vkl67880v [Accessed 19 Nov 2008].

Van de Putte, G., Jana, J., Keen, M., Kondepudi, S., Mascarenhas, R., Ogirala, S., Rudrof, D., Sullivan, K. & Swithinbank, P., 2004. *Using Web services for business integration* [online], IBM Redbooks. Available from: http://www.redbooks.ibm.com/abstracts/sg246583.html?Open [Accessed 20 Nov 2008].

W3C, 2004. *Web Services Architecture Requirements* [online]. (Updated 11 Feb 2004). Available from: http://www.w3.org/TR/wsa-reqs/ [Accessed 20 Nov 2008].

Weisenfeld, U., Fisscher, O., Pearson, A. & Brockhoff, K. 2001. Managing technology as a virtual enterprise. *R&D Management* [online]. 31(3), pp. 323-334. Available from: http://web.ebscohost.com/ehost/detail?vid=1&hid=114&sid=b503a4d0-d95f-40b6-83b0-6fb8846f09b8%40sessionmgr102&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#db=buh&AN=4917886 [Accessed 19 Nov 2008]

Wu, N. & Su, P., 2005. Selection of partners in virtual enterprise paradigm. *Robotics and Computer-Integrated Manufacturing* [online], 21(2), pp. 119-131. Available from: http://www.sciencedirect.com/science/article/B6V4P-4D4D1GP-2/2/ef5b5f277e3dc17aadba73be632c2771 [Accessed 19 Nov 2008].

Yang, Z., Zhangx, J-B., & Low, C.P., 2006. Towards Dynamic Integration of Collaborative Virtual Enterprise Using Semantic Web Services. *Industrial Informatics, 2006 IEEE International Conference on 16-18 Aug. 2006* [online], pp.102-107. Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4053370&isnumber=4053336 [Accessed 21 Nov 2008].

APPENDIX A

```
!*********************************************************
! Class RouterRTE
!*********************************************************
base "syslog.cfg" SYSLOG
base "partner.cfg" RECIPIENT

function bfFindPartner(tIdentifier)
    bRet := FALSE
    if tIdentifier <> EMPTY then
        RECIPIENT := find(sPARTNER,SAPPRN=tIdentifier)
        if valid(RECIPIENT) then
            bRet := TRUE
        endif
    endif
    return bRet
endfunction

function tfDetermineProcessing(tFormat,tType)
    tRet := EMPTY
    tProcess := build(tFormat,"_",tType)
    load(RECIPIENT.info,taInfo,"=")
    tRet := taInfo[tProcess]
    return tRet
endfunction

function bfLaunchProcessing(tEdisend,tFile,tIndex,tFormat,tType,tRecipient)
    bRet := FALSE
    fFile := tFile
    if fFile.EXIST and tEdisend <> EMPTY and tIndex <> EMPTY and tFormat <> EMPTY\
    and tType <> EMPTY and tRecipient <> EMPTY then
        nCmd := system(build("edisend ","IRSINDEX=",tIndex," ",“IRSFORMAT=",tFormat,"","IRSTYPE=",tType," ",\
                    "IRSRECIPIENT=",tRecipient," ",tEdisend," ",tFile))
        if nCmd = 0 then
            bRet := TRUE
            remove(tFile)
        endif
    endif
    return bRet
endfunction
```

APPENDIX B

```
!*******************************************************
! Class ORDERSD97ARouterRTE implements EDIFACTRouterRTE
!*******************************************************
message "UN-EDIFACT/97/draft/A/orders.msg" receiving

#include "../inc/RouterRTE.inc"

begin
    tMessageFormat := "EDIFACT"
    tMessageType   := "ORDERSD97A"
    nMessageCount  := 0
    nErrors        := 0
    tInputFile     := pGW.ORIGINAL.NAME
    split(tInputFile,taTemp,"_")
    tOriginalFile  := peel(tInputFile,build(taTemp[1],"_"))
    SYSLOG := find(sSYSLOG,INDEX=number(pINDEX))
    if not valid(SYSLOG) then
        exit(1)
    endif
    tRecipientUNB  := pUNB.S003.0010
    if bfFindPartner(tRecipientUNB) then
        tProcess := tfDetermineProcessing(tMessageFormat,tMessageType)
        if tProcess <> EMPTY then
            tTargetFile := build(sHOME,"/outgoing/IRS/1_",tOriginalFile)
            copy(SYSLOG.x,tTargetFile)
            if bfLaunchProcessing(tProcess,tTargetFile,pINDEX,tMessageFormat,tMessageType,tRecipientUNB)
then
                exit(0)
            endif
        endif
    endif
endbegin

segment UNH
    nMessageCount++
endsegment

segment NAD g2
    if e3035 = "SE" then
        taRecipient[nMessageCount] := eC082.3039
        if bfFindPartner(taRecipient[nMessageCount]) then
            taProcess[nMessageCount] := tfDetermineProcessing(tMessageFormat,tMessageType)
        else
            nErrors++
        endif
    endif
endsegment

end
    bfFilterMessages()
    nIndex := 1
    while nIndex < (nMessageCount+1) do
        if taProcess[nIndex] <> EMPTY then
            if not
bfLaunchProcessing(taProcess[nIndex],taSplitFile[nIndex],pINDEX,tMessageFormat,tMessageType,\
            taRecipient[nIndex]) then
                nErrors++
            endif
        else
            nErrors++
        endif
        nIndex++
    endwhile
    if nErrors > 0 or nMessageCount = 0 then
        exit(1)
    endif
endend

function bfFilterMessages()
```

```
    nUNHCount := 0
    tInput:= SYSLOG.e
    tLine := read(tInput)
    while tLine <> EOF do
        tTAG := substr(tLine,1,3)
        switch tTAG
            case "UNH":
                nUNHCount++
                if nUNHCount > 1 then
                    close(taSplitFile[nUNHCount-1])
                    if taProcess[nUNHCount-1] = EMPTY then
                        remove(taSplitFile[nUNHCount-1])
                    endif
                endif
                taSplitFile[nUNHCount] := build(sHOME,"/outgoing/IRS/",nUNHCount,"_",tOriginalFile)
                system(build("touch ",taSplitFile[nUNHCount]))
                print(tLine) >> taSplitFile[nUNHCount]
            default:
                print(tLine) >> taSplitFile[nUNHCount]
        endswitch
        tLine := read(tInput)
    endwhile
    if nUNHCount > 0 then
        close(taSplitFile[nUNHCount])
        if taProcess[nUNHCount] = EMPTY then
            remove(taSplitFile[nUNHCount])
        endif
    endif
endfunction
```