HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Electrical and Communications Engineering

Antti Kiviluoto

Application Integration in Intranets and Extranets

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Engineering

Helsinki, March 1st, 2004

Supervisor Professor Heikki Hämmäinen

Author:	Antti Kiviluoto
Name of the Thesis:	Application Integration in Intranets and Extranets
Date:	1.3.2004
Number of pages:	94
Department:	Department of Electrical and Communications Engineering
Professorship:	S-38 Telecommunications Technology/ TU-91
Supervisor:	Professor Heikki Hämmäinen
Instructor:	M. Sc. Sami Tähtinen

This thesis examines the integration of applications within and between companies. The integration of companies' internal applications is known as Enterprise Application Integration (EAI). Term Business-to-Business integration is used when integrating applications and processes between companies. The idea of this thesis is to present the technology used in application integration and to analyze the opportunities and challenges that application integration provides.

The theoretical part examines the technologies, standards and methodologies behind EAI and B2Bi. It also presents the general information systems used in companies, which set the foundation for application integration.

The analysis part analyzes the integration market with products and, most of all, the opportunities and challenges in application integration for companies. Analysis is carried out by presenting different factors related to application integration that display the opportunities and/or challenges for companies. The focus of the thesis is on the opportunities and challenges delivered by EAI and B2Bi products.

Keywords: Enterprise Application Integration, EAI, Business-to-Business Integration, B2Bi

Tekijä:	Antti Kiviluoto
Työn nimi:	Sovellusintegraatio sisä- ja ulkoverkoissa
Päivämäärä:	1.3.2004
Sivumäärä:	94
Osasto:	Sähkö- ja tietoliikennetekniikan osasto
Professuuri:	S-38 Teletekniikka/ TU-91
Työn valvoja:	Professori Heikki Hämmäinen
Työn ohjaaja:	DI Sami Tähtinen

Työ käsittelee yritysten sisäisten sovellusten ja yritysten välisten sovellusten integraatiota. Yritysten sisäisten sovellusten integraatiosta käytetään nimitystä Enterprise Application Integration (EAI) ja vastaavasti yritysten välisten sovellusten integraatiosta käytetään nimitystä Business-to-Business integration (B2Bi). Työn tarkoituksena on esitellä sovellusintegraatioon liittyvät tekniikat sekä analysoida niiden tarjoamat mahdollisuudet ja haasteet yrityksille.

Työn teoriaosuudessa esitellään mitä käsitteet EAI ja B2Bi tarkoittavat sekä niihin liittyvät tekniikat, standardit ja menetelmät. Tämän lisäksi teoriaosuudessa käydään läpi yritysten yleisimpiä tietojärjestelmiä, joiden integraatiosta EAI:ssa sekä B2Bi:ssä lopulta on kysymys.

Työn analyysiosio tutkii integraatiomarkkinoita, niihin liittyviä tuotteita sekä olennaisimpana osana sovellusintegraatioon liittyviä mahdollisuuksia ja haasteita yrityksille. Analyysi tapahtuu tuomalla esiin erilaisia sovellusintegraatioon liittyviä tekijöitä, jotka tarjoavat joko mahdollisuuksia tai haasteita yrityksille. Pääpaino analyysissä on sovellusintegraatiotuotteiden tarjoamissa mahdollisuuksissa sekä haasteissa.

Avainsanat: järjestelmäintegraatio, sovellusintegraatio, yritysten välinen integraatio, EAI, B2Bi

PREFACE

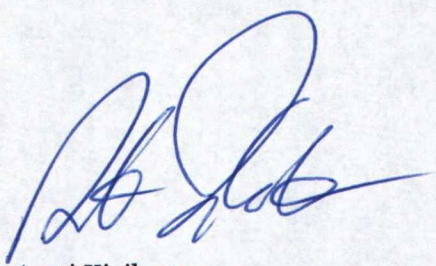
This master's thesis was made for FRENDS Technology Inc. in Finland. I wish to thank the company for giving me the opportunity to carry out the study. I also wish to thank all my colleagues for their valuable support.

I want to express special thanks to my instructor Sami Tähtinen and to supervisor Heikki Hämmäinen for their guidance throughout the procedure.

I am grateful for my family for the support and help with my studies.

Finally, very special thanks to my wife Miia and to our new-born baby girl Julia (27.12.2003). They set me a good timetable for graduation and provided their endless support till the end.

Helsinki, Finland, March 2004



Antti Kiviluoto

INDEX

LIST OF ABBREVIATIONS

ANSI	American National Standards Institute
API	Application Programming Interface
ARTS	Association for Retail Technology Standards
B2B	Business-to-Business
B2Bi	Business-to-Business integration
BAM	Business Activity Monitoring
BPM	Business Process Management
BPMI	Business Process Management Initiative
BPSS	Business Process Specification Schema
CDMA	Code Division Multiple Access
COM	Component Object Model
CORBA	Common Object Request Broker Architecture
CRM	Customer Relationship Management
DBMS	DataBase Management System
DCOM	Distributed Component Object Model
DTD	Document Type Definition
EAI	Enterprise Application Integration
EDI	Electronic Data Interchange
EDIFACT	Electronic Data Interchange for Administration, Commerce and Transport
ERP	Enterprise Resource Planning
FTP	File Transfer Protocol
GSM	Global System for Mobile Communications
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
JDBC	Java Database Connectivity
JMS	Java Messaging Service
MOM	Message Oriented Middleware
OBI	Open Buying on the Internet
ODBC	Open Database Connectivity
OMG	Object Management Group
PDM	Product Data Management
PGP	Pretty Good Privacy
PIP	Partner Interface Processes
ROI	Return of Investment
RPC	Remote Procedure Call
SCM	Supply Chain Management
SMS	Short Message Service
SMTP	Simple Mail Transfer Protocol

SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSL	Secure Sockets Layer
TCO	Total Cost of Ownership
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VAN	Value Added Network
WAP	Wireless Application Protocol
WSDL	Web Services Description Language
WWW	World Wide Web
XML	eXtensible Markup Language
XSLT	eXtensible Stylesheet Language Transformations

TRADEMARKS

BizTalk is a trademark of Microsoft Corporation.

FRENDS is a registered trademark of FRENDS Technology, Inc. and FRENDS Technology USA, Inc.

RosettaNet is a trademark of RosettaNet organization.

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

Windows and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

All other mentioned products are trademarks or registered trademarks of their respective owners.

1. INTRODUCTION

1.1. MOTIVATION FOR THE STUDY

Companies have been investing to their applications for decades to serve their business operations. The application environment has become extremely heterogeneous and the amount of applications inside companies is still growing. The need of sharing and controlling data, information and internal processes inside the company, meaning also between the applications, is evident. Enterprise Application Integration (EAI) can provide the solution to this requirement.

Automating and controlling internal processes and information sharing is important for companies. The next challenge for companies is to do the same thing to external processes between business partners. Business-to-Business integration (B2Bi) can provide the solution to this challenge. This challenge is more difficult because standardization is the key to success and it is still in the very early stages.

These two approaches to application integration are tightly bonded to each other. The process between business partners is a combination of two processes – the public and private process. Companies must have their internal private processes automated and controlled before they can start building the external public processes that enable the business-to-business integration.

The purpose of this thesis is to find out what are the challenges and opportunities in the area of application integration within and between companies. The literature study will explain what are the technologies, methodologies and standards for integrating applications. The analysis part will deliver the opportunities and challenges by means of technology and business perspective.

1.2. RESEARCH PROBLEM

The research problem of the present study is:

What are the technologies, methodologies and standards for integrating applications within and between companies? What are the challenges and opportunities for companies in application integration by means of technology and business perspective?

1.3. OBJECTIVES OF THE STUDY

- The first objective is to do a literature survey of EAI and B2Bi.

- The second objective is to deliver market and product analysis of the application integration segment.
- The third objective is to analyze the challenges and opportunities for companies in application integration from technology and business perspective.

1.4. SCOPE OF THE STUDY

The scope of this study is limited to corporate and business-to-business level of application integration. There is no analysis of consumer side application integration or consumer e-business area.

1.5. RESEARCH METHODS

Used research methods are:

- Literary research
- Interviews

In the literary research the material is gathered from books, research papers, articles, and the Internet. There are only limited amount of books written about the research area, as the research area is quite new. Therefore the main source of information will be articles and web sites covering the area, and the material published by the organizations developing related software or standards as well as providing services in the area.

1.6. USED TERMS

This thesis is about EAI and B2Bi. The application integration can be divided in two main categories. EAI is integration of company's internal systems whereas B2Bi is integration of systems between companies. From the process perspective EAI means the integration of companies private processes and B2Bi the integration of public processes between companies.

The terms EAI and B2Bi are rather new, however, application integration has existed since the first applications. The term EAI became commonly used in the middle of nineties and B2Bi at the end of nineties. This is why there are quite a variety of names for solutions, products and architectures in this segment. In the area of application integration within and between companies following kind of terms exists: Middleware, Message Broker, Integration Broker, Integration Broker Suite, Adapter Suite, Application Server, Platform middleware, Online Transaction Processing Application Server, Portal Product, Application Platform Suite, Business-To-Business Server, etc.

In this thesis EAI is used in the context of internal application integration and private process implementation within companies, and B2Bi is used in the context of external application integration and public process implementation between companies.

2. ENTERPRISE APPLICATION INTEGRATION

2.1. INTRODUCTION

For the past few decades companies have been developing and implementing several information systems that serve companies' businesses. The technical environment inside the company is rather heterogeneous. There are applications written in several programming languages and running at different operating system platforms. Business is requiring more interoperability among the applications, and the communications between applications should be done online. Applications together should also support the business processes as well.

This chapter will introduce the technologies, standards and methods for integrating companies' applications internally. It starts from the most common information systems in company, goes over the technology issues in EAI and ends up with the process perspective of EAI. The following chapters introduce the four layers of EAI that are shown in the figure 1.

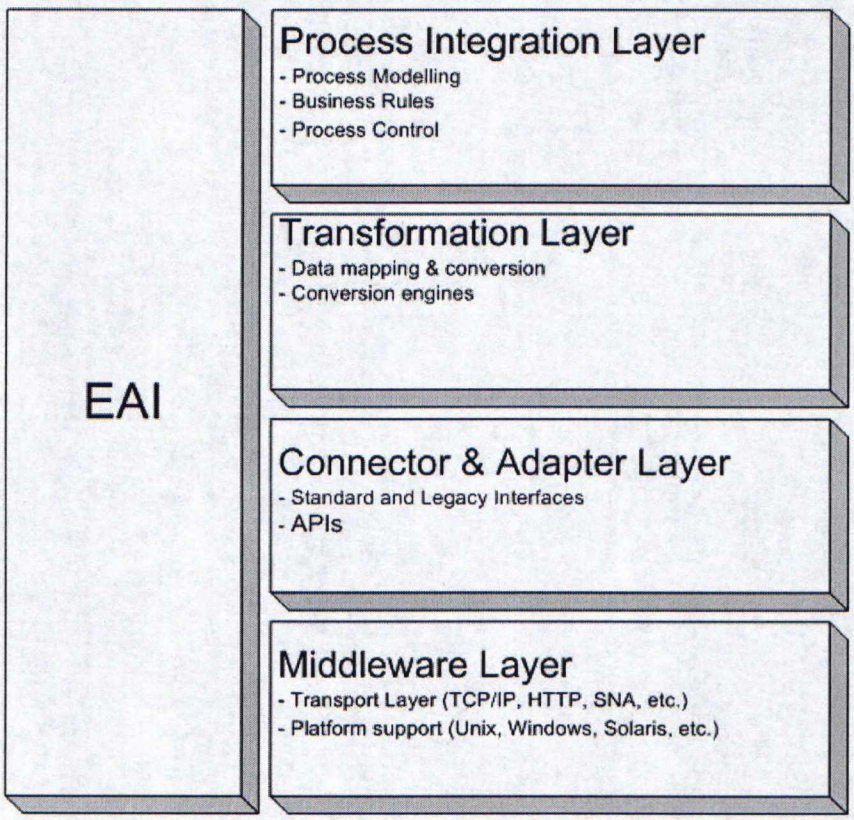


Figure 1. *The Layers of EAI.*

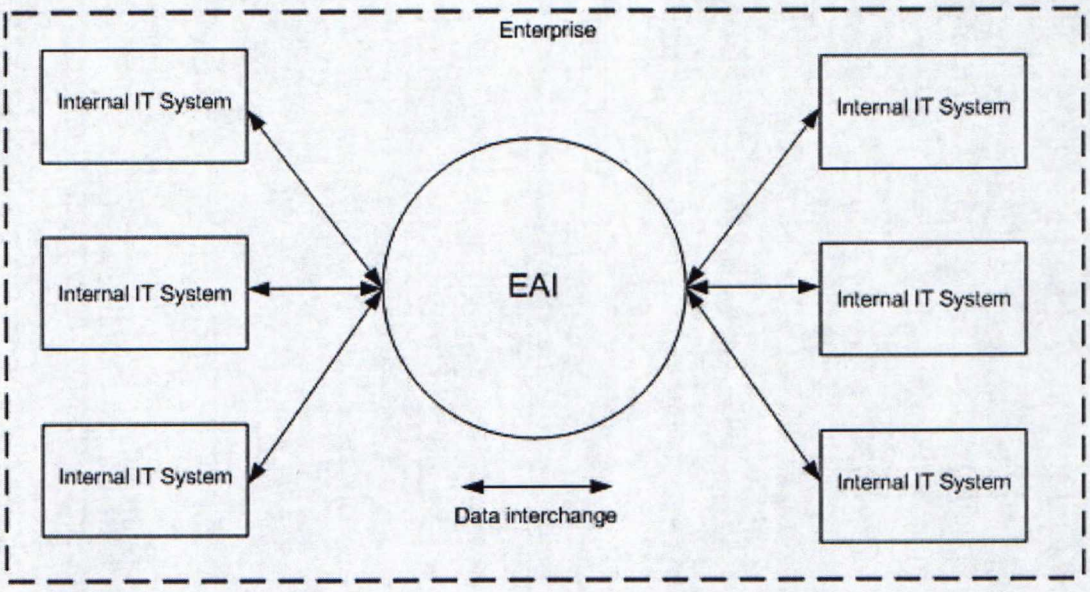


Figure 2. *Relation of EAI and Internal Information Systems.*

2.2. INFORMATION SYSTEMS

Information systems are a vital part of companies' businesses. Most of the companies' key operations have their own information system and this creates a very heterogeneous system environment inside companies. Companies have invested a fortune to these systems and their goal is to make the most out of these applications.

This chapter introduces the most important information systems for companies. At the same time it shows that companies can have a number of different information systems. The intercommunication of these systems is vital for the business operations. This leads to the need of application integration.

2.2.1. ENTERPRISE RESOURCE PLANNING

Enterprise resource planning (ERP) attempts to integrate most departments and functions across a company onto a single computer system that can serve all those different departments' particular needs. ERP automates the tasks involved in performing a business process—such as order fulfilment, which involves taking an order from a customer, shipping it and billing for it. It tries to serve the needs of people in finance as well as it does the people in human resources and in the manufacturing generally. Each of those departments typically has its own computer system optimized for the particular ways that the department does its work. But ERP combines them all together into a single, integrated software program that runs a single database so that the various departments can more easily share information and communicate with each other. ERP is a comprehensive transaction management system that

integrates many kinds of information processing abilities and places data into single database [SHA01, p.31].

ERP vanquishes the old standalone computer systems in finance, human resources, manufacturing and the warehouse, and replaces them with a single unified software program divided into software modules that roughly approximate the old standalone systems. Finance, manufacturing and the warehouse all still get their own software, except now the software is linked together so that someone in finance can look into the warehouse software to see if an order has been shipped. Most vendors' ERP software is flexible enough that you can install some modules without buying the whole package. Many companies, for example, will just install an ERP finance or Human Resource module.

ERP has an integrated approach to enterprise's different systems and functions. In reality the case is that enterprises still have separated systems for their different needs and ERP offers only part of this. As mentioned earlier ERP only roughly approximates the old standalone systems. Companies still purchase separate software packages that best serve their business needs. ERP software vendors have also understood this reality and they have started offering integration modules with their software to ease the integration to other business applications.

Most known ERP software products are SAP, Baan and J.D. Edwards.

2.2.2. CUSTOMER RELATIONSHIP MANAGEMENT

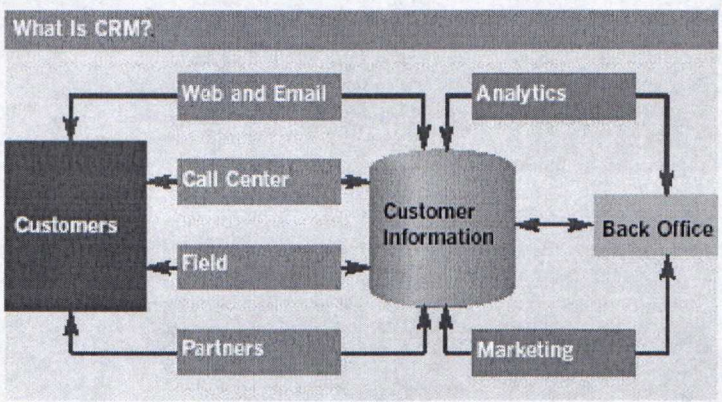


Figure 3. CRM information flows

Customer Relationship Management (CRM) helps businesses use technology and human resources to gain insight into the behaviour of customers and the value of those customers. It is a strategy used to learn more about customers' needs and behaviours in order to develop stronger relationships with them. After all, good customer relationships are at the heart of business success. If CRM works as hoped, a business can provide better customer service,

make call centers more efficient, help sales staff close deals faster, simplify marketing and sales processes, discover new customers and increase customer revenues.

CRM software offers a powerful tool to manage all customer information via one system. It is also evident that the information must be delivered from various different systems. This opens also an interesting business case for application integration while the vital customer related information lies in the heterogeneous application environment.

Most known CRM software products are Siebel, Peoplesoft and J. D. Edwards.

2.2.3. SUPPLY CHAIN MANAGEMENT

Supply Chain Management (SCM) systems are back-end applications designed to link suppliers, manufacturers, distributors and retailers in a cohesive production and distribution network. SCM system improves the way company finds the raw components it needs to make a product or service manufacture that product or service and deliver it to the customers. Primary goals of SCM systems are:

- Decreasing inventory costs by matching production to demand,
- Reducing overall production costs by streamlining the flow of goods through the production process,
- Improving information flow between different parties and
- Improving customer satisfaction by offering increased speed and adaptability.

SCM tools are often divided into two different modules: Supply chain planning software and supply chain execution software. Supply chain planning software uses complex math algorithms to improve the flow and efficiency of the supply chain and reduce inventory. Planning is entirely dependent upon information for its accuracy. It needs accurate and up-to-date information of for example customer orders, sales data, manufacturing capacity and delivery capability. The execution software is intended to automate the different steps of the supply chain. This could be as simple as electronically routing orders from manufacturing plants to suppliers.

Most known SCM software products are i2, J.D. Edwards and SAP.

2.2.4. PRODUCT DATA MANAGEMENT

Product Data Management (PDM) system is used to organise, access, and control the needed information on products and services that companies provide. The product information provided by PDM varies from CAD pictures to the detail of sales data. PDM is mostly used by the product development, product planning and quality management teams.

There are many synonyms for PDM, such as Engineering Document Management, Product Information Management, Technical Document Management, etc.

Most known PDM software products are MatrixOne, Agile Software and ERP software providers.

2.2.5. OTHERS

There are also several other packaged information systems for companies such as finance and storage systems. Companies have also several proprietary and legacy information systems that can't be categorized easily. These systems have been developed and tailored just for companies special business operations.

All these systems play a major role in companies' operations and contain important data to be shared among other systems and employees. This is why these systems are also vital part of the application integration scene. Usually the proprietary systems are most challenging to integrate while they might not have any standardized communication interfaces or the documentation of these systems is incomplete.

2.3. MIDDLEWARE LAYER

Middleware is any type of software that facilitates communications between two or more information systems. It is able to hide the complexities of the source and target systems, freeing developers from focusing on low-level Application Programming Interfaces (APIs) and network protocols, allowing them to concentrate on sharing information [LIN01, p.127]. For example, a single middleware API can be used across many different types of application development products and many different platforms. This use of a common API hides the complexities of both the entities being connected and the platforms they reside on.

On the following sub-chapters different models of middleware is introduced.

2.3.1. POINT-TO-POINT MODEL

Point-to-point middleware utilizes a simple pipe to allow one application to link to one other application. When application A seeks to communicate with application B it can use the existing middleware pipe by using a Remote Procedure Call (RPC) or message.

Point-to-point middleware is limited by its inability to bind together more than two applications, although it is not impossible to bind several applications together. It requires point-to-point links between all the applications that are involved, which means exponential

growth of connections between applications. The equation in figure 6 shows that the amount of connections rises exponentially. When there are 10 point-to-point connected applications, the amount of connections can be as much as 45. It is also just a "dummy" pipe that doesn't involve any understanding for application logic.

Point-to-point middleware solution is at its best when there are only few applications to be interconnected. It is used especially between most important applications that need transactional support and high reliability. The great advantage of point-to-point middleware is the simplicity.

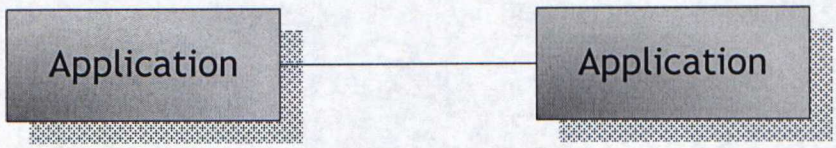


Figure 4. *Point-to-point model, two applications.*

On the other hand, when there are several systems to be interconnected, point-to-point model creates a number of connections between applications. This kind of environment is expensive and difficult to maintain, develop and change.

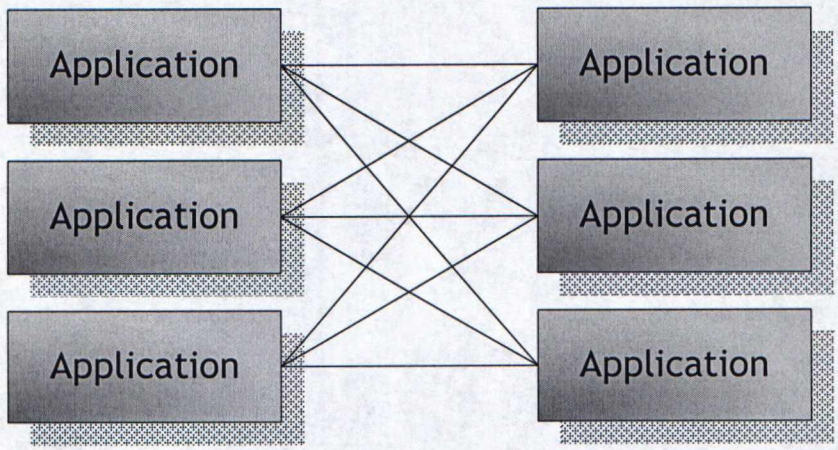


Figure 5. *Point-to-point model, several applications*

amount of connections $= n(n-1)/2$,

where n = *number of applications* (1)

Figure 6. *Exponential growth of connections in point-to-point model*

2.3.2. MANY-TO-MANY MODEL

Many-to-many middleware links many applications to many other applications. Any type of middleware that can deal with more than two sources or target applications at the same time is considered to support this model. These kinds of middleware are message brokers, application servers and transaction processing monitors. Most of the companies, excluding the small ones, have several applications to be integrated that make this option the most popular middleware model.

This model minimises the amount of connections needed between different applications. It reminds the architecture of telephone network where subscribers' terminals are like applications and the switchboard is like many-to-many model middleware. It means that the amount of connections equals the amount of applications.

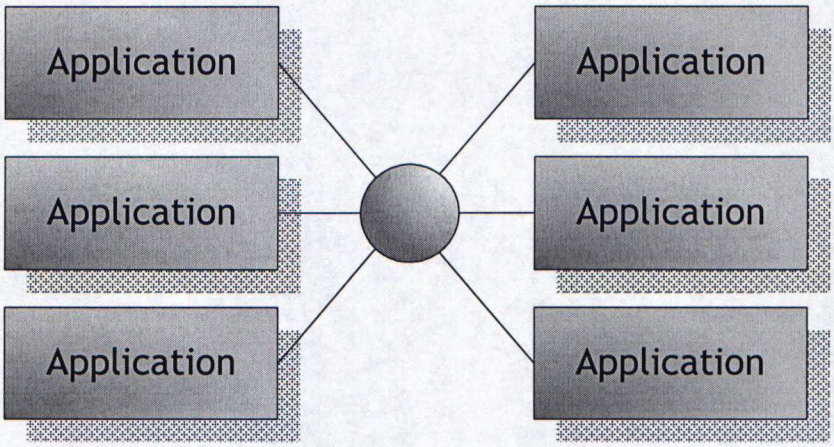


Figure 7. Many-to-many model

2.3.3. COMMUNICATION METHODS

There are two basic models to communicate and exchange messages between applications: synchronous and asynchronous communications. Synchronous communications is connection-oriented communications model, which means that two parties (processes) establish connection, exchange data and then disconnect. Asynchronous communications is connectionless communications model that does not establish any connection with the target process. Target application just acts on the request and responses only if it is required.

Asynchronous and Synchronous communications

Asynchronous middleware moves information between applications in asynchronous mode. This means that applications are not dependent on other connected applications for processing. Application is not required to respond to the received message right away; it can

send a response while it is ready or neglect the response totally. The process that allows this to occur has the application(s) placing a message in a message queue and then resuming with its own tasks, waiting for the response at later time from the application(s). The middleware does not block the application for processing, in fact, applications can continue processing regardless of the state of the other applications.

Synchronous middleware moves information between applications in synchronous mode. Synchronous middleware is tightly coupled to applications and visa versa. The calling application must stop other processing in order to wait for the remote application to respond. This is called blocking type of middleware [LIT01, p.132]. Synchronous model is good for online-transaction processing where the transaction is evolved with several systems such as payment card online credit check. The disadvantage of the synchronous model is the coupling of the application to the middleware and the remote application. Problems with middleware, like network or remote server problems, stop the application from processing. In addition, synchronous middleware eats bandwidth while the synchronous process requires longer function call across the network.

Queued Communications

Queued communications generally require a queue manager to place a message in a queue. The remote application then retrieves the message – either shortly after it has been sent or at any time in the future (barring time-out restrictions). If the calling application requires a response (such as a verification message or data), the information flows back through the queuing mechanism. Most MOM products use queued communications. The advantage of the queuing communications model is that the remote program does not need to active for the calling program to send a message to it. Queuing communications middleware typically does not block either the calling program or the remote program from proceeding with processing. [LIN01, p.133]

Request and reply

Request and reply (R/R) is almost an attempt to make asynchronous messaging look synchronous – it can be used to force a particular sequence of business events [SAI01].
The R/R model is exactly what its name implies. A request is made to an application using R/R middleware, and it responds to the request. Examples of R/R middleware include any middleware that can facilitate a response from a request between applications, such as message brokers or application servers [LIN01, p.134].

Fire and Forget

Fire and Forget (F/F) represents the truly asynchronous model of sending message and then continuing to follow the sub-process currently being executed. The message sent will probably

trigger some other sub-processes but these can run in parallel. The other implication of F/F is that delivery of the message is guaranteed, once only. The alternative is to have extensive coding to deal with the risk of missing or duplicate message [SAI01].
The F/F model allows the middleware user to "fire off" a message and then "forget" about it, without worrying about who receives it or even if the message is received. This is another example of an asynchronous approach. The purpose of F/F is to allow a source or target application to broadcast specific types of messages to multiple recipients, bypassing auditing and response features [LIN01, p.136].

Publish and Subscribe

Publish and subscribe (P/S) model is a very different form of messaging than R/R and F/F. In this model components can subscribe to a particular type of message or to some particular content field value. When an application generates a message it publishes the message, thereby making it available to the EAI software to handle it. The message and/or contents are matched to the list of subscribers and the message is then sent to these subscribers only [SAI01].

P/S model frees an application from the need to understand anything about the target application. All it has to do is send the information it wants to share to a destination within the P/S broker. The broker then redistributes the information the information to any interested applications. For example, if a financial application wants to make all accounts receivable information available to other applications that want to see it, it informs the P/S broker. The broker then makes it known that this information is available, and any application can subscribe to that topic in order to beginning receiving account receivable information. In this scenario, the publisher is the provider of the information. Publishers supply information about the topic without needing to understand anything about the applications that are interested in the information. The subscriber is the recipient, or consumer, of the information. The publisher specifies a topic when it publishes the information. The subscriber specifies a topic that it's interested on. In this scenario the subscriber receives only the information it's interested in [LIN01, p.134].

2.3.4. Message Oriented Middleware

Message Oriented Middleware communication is based on messaging between applications. MOM is able to guarantee that messages will reach their destination – even when the destination is not available at the time the messages are sent. MOM provides a standard API across hardware, operating system platforms, and networks [LIT01, p.170].

MOM utilizes one of two "macro-messaging" models: process-to-process or message queuing. In the process-to-process model, to exchange messages, both the sending and receiving processes must be active. In the queuing model, only one process must be active, because

messages can be stored in a message queue. The queuing model is the best when communication is taking place between computers that are not always up and running, over networks that are not always dependable, or when there is a limitation of bandwidth. The queuing model does not require the platform to be up and running for an application to request services. If the server is down, the request remains in a queue. As soon as the server comes back online, the request is processed.

Unlike RPCs, which block the processing until the procedure call returns, MOM is asynchronous and consequently allows the application to process when the middleware API is invoked. MOM message functions can return immediately, even thought the request has not been completed. This allows the application to continue processing, assured that it will know when the request is completed. In addition to these queuing features, MOM provides concurrent execution features, allowing the processing of more than one request at a time.

MOM is a good choice for store-and-forward communications or when dealing with applications that are not expected to be reachable at the same time. MOM is a good choice for "defensive communications" – communications between applications when networks frequently fail. MOM is also a good choice when communications between processes need to be logged.

2.3.5. DISTRIBUTED OBJECTS

Distributed object and component technology allow objects and components, distributed throughout servers within and between enterprises, to interoperate and share functionality. Client objects and components invoke public methods of server objects and components providing business method level integration.
An object is a self-contained software entity that consist of three distinct parts [SAM02, p.231]:

- Private Data – information or attributes, generally of a persistent nature, which define an object
- Private Methods – internal procedures for accessing or updating the private data
- Public Interface – public methods for communicating with other objects.

An object contains both data and business logic in a single software entity or package. It is designed to carry out the same functions as those in real world.

Distributed objects are classified as middleware because they facilitate inter-application communications [LIT01, p.140]. They are also mechanisms for application development, providing enabling technology for enterprise- or trading community-wide method sharing. In

fact, distributed objects are really small application programs that utilize standard interfaces and protocols to communicate with one another.

Distributed objects must work within a common framework, using commonly distributed protocol. There are currently three competing distributed architectures:

- Common Object Request Broker Architecture (CORBA) by Object Management Group (OMG)
- Java 2 Enterprise Edition Enterprise JavaBeans (J2EE EJB) by Sun Microsystems
- Windows .NET by Microsoft

Each protocol defines specification for object interoperability, interfaces, communication and distribution. These standards will be discussed in more detail in the chapter 2.7 "Standards".

2.4. Adapter Layer

Middleware is software that is able to connect to several different software components that can be geographically distributed and may operate on different software and hardware platforms. Middleware is different than communication protocol; instead of giving an access to a protocol stack, middleware usually gives an access to an API. Middleware itself can utilize one or several communication protocols. Different middleware solutions include distributed object-oriented architectures such as CORBA or DCOM and queuing systems such as IBM's MQSeries or Microsoft's Message Queuing.

Middleware approach is not adequate alone as it defines only infrastructure for intercommunication. Middleware can manage the communication between different components of the system, but it is not able to handle possibly complex processes such as conversion and routing that the overall functionality of the system requires. EAI software has grown on the top of the middleware layer. Typical features for EAI software include connectors and adapters to different middleware, software and communication components as well as business process engines being able to process, filter and route information intelligently between different components in the integration domain.

Two distinct adapter architectures are emerging: distributed and centralized. As the name suggests, centralized adapters run with the EAI software. Generally, these are thin adapters that only bind the EAI's API to the API of the source or target application. Just as centralized adapters are thin adapters, distributed adapters are thick adapters that exist on the EAI software as well as on the source or target application. Running an adapter on the application being integrated allows many processes of the source or target application to be better determined, such as capturing events, monitoring states, or even restarting the application as

required. Because the adapter is in two parts, it is better able to coordinate the transfer of information between the EAI software and the source or target application.

Basic reasons for the need of adapters:

- The need of reusable set of software services that can extract and publish information to source or target systems. The need of individual interfaces in different integration projects is diminished.
- The need of common interfaces into source or target systems that provide a consistent set of services
- The need of management visibility into connections between source and target systems. This is needed for reliability, as source and target systems can fail and adapters can help manage those outages.

2.4.1. THIN ADAPTER

The most popular EAI softwares offer thin adapters today. In most cases, they are simply API wrappers, or binders, that map the interface of the source or target system to a common interface supported by the EAI software. In other words, they simply perform an API-binding trick, binding one API to an other. Thin adapters don't provide sophisticated layers of software between the source or target systems and the EAI software, and are just simple abstractions on top of existing APIs.

Thin adapters have the advantage of being simple to implement. With no additional, "thick" layer of software between sources and target applications, there is greater control. Thin adapters have a number of disadvantages, however. Since using thin adapters accomplishes roughly nothing more than trading one interface for another. Thin adapters impact performance without increasing functionality. Fair amount of programming is still required, however, this functionality can be implemented centrally to the EAI software. Complicating matter is the fact that the common APIs that are being mapped are almost always proprietary. This is more of an opportunity for thin adapters while no thick adapters exist for proprietary systems either. In this case the use of simple thin adapters and building the integration functionality to the EAI software in the middle with its efficient tools is attempting choice.

Other examples of thin adapters are wrapping application interfaces using open interface standards, such as CORBA or COM. Here again, one interface is being traded for another. However, in this case, providing a common, open interface is an advantage. Most ERP vendors are seeking to create open interfaces to their applications, for both data- and process-oriented integration, making integration easier to implement with traditional tools while

reducing the risk of proprietary interfaces. For example, SAP is seeking to provide interfaces based on CORBA and Java. [LIT01, p.250-251].

2.4.2. THICK ADAPTERS

Unlike thin adapters, thick adapters provide significant amount of software and functionality between the EAI software infrastructure and the source or target applications. The thick adapter's layer of abstraction makes managing the movement of information or invoking processes painless. Because the abstraction layer and the manager negotiate the differences between all the applications requiring integration, almost no programming is needed.

The layer of sophisticated software that hides the complexity of the source and target application interfaces from the EAI software user allows thick interfaces to accomplish this. The user sees only a businesslike representation of the process and the metadata information as managed by the abstraction layer and the adapter. In many cases, the user connects many systems through this abstraction layer and the graphical user interface, without even having to resort to hand coding.

Repositories are major players in the thick adapter scenario. As we noted, the repository is able to understand much of the information about the source and target applications and is able to use that information as a mechanism to interact with those applications on behalf of the EAI software. In addition, several abstraction layers may be created around the type of applications to be integrated. For example, there may be an abstraction for common middleware services such as distributed objects and MOM. There may also be an abstraction layer for packaged applications and other layer that is able to address the integration of relational and non-relational databases. This structure hides from the end user the complexities of the interface that each entity (middleware, packaged application, database, etc.) employs.

With the many advantages and conveniences of thick adapters, it is quite obvious that EAI software vendors are moving toward them. Their progress is slowed by the fact that thick adapters require tremendous amount of time to develop, as much as six times of a thin adapter. Right now, this time investment deters some vendors. However, problem is that large "packaged" information systems such as ERPs are still quite customized, which makes the development of thick adapters extremely problematic.

Enterprises will continue to look for more sophisticated solutions, solutions that require no programming and provide an easy, businesslike method to view the integration of the enterprise [LIT01, p.251-253].

2.5. TRANSFORMATION LAYER

The internal representation of numeric and character data is not uniform on computer systems from different or even the same vendors. The representation generally depends on the utilized hardware and software (operating system, programming language, I/O library). For this reason, data that is to be used on more than one computer system or is to be moved between systems normally has to be converted into the proper format before use.

One of the most time consuming operations in EAI projects are the conversions. Applications have different ways of presenting the output and understanding the input. Standardization of message formats and vocabularies is still at a very early stage, which makes the field of conversion and mapping very important and needed in today's EAI projects. First commercial graphical mapper products came into the market in the late nineties by Microsoft BizTalk Server. The development of mapper tools is in the early phase and it is yet to see will it bring the desired cost and time reduction to the EAI projects.

2.5.1. SCHEMA CONVERSION

A schema conversion is the process of changing the structure of a message and remapping the schema so that it is acceptable to the target system. Though it is not difficult, application integration architects need to understand that this process must occur dynamically within the integration solution.

For example, if a message containing accounts receivable information arrives from a DB2 system on mainframe, it may look something like this:

```
- Cust_No              Alphanumeric 10        AB99999999
- Amt_Due              Numeric 10             560.50
- Date_of_Last_Bill    Date                   09/17/02
```

The client/server system created to produce the annual report receives the information and must store it according to the following schema:

```
- Customer_Number    Numeric 20
- Money_Due          Numeric 8
- Last_Billed        Alphanumeric 10
```

Clearly, the schema in the client/server system is different from the schema in the DB2 system. Moving information from the DB2 system (the source system) to the client/server system (the target system) without schema conversion would most likely result in a system error because of the incompatibility of the formats. For the system to communicate

successfully, the information in Cust_No needs to be converted to all numeric information capable holding 20 digits or positions. All data that is not numeric must be translated into numeric data. It can be accomplished either by deleting all characters when translating Cust_No to Customer_Number or by converting characters into numeric representation (A=1, B=1, and so on). This process can be translated dynamically, depending on its content and schema. Moving information from one system to another demands that the schema or format of the message to be altered as the information is transferred from one system to the next.

Although most mapper tools can map any schema to any other schema, it is prudent to try to anticipate extraordinary circumstances. For example, when converting information extracted from an object-oriented database and placing it in a relational database, the mapper tool must convert the object schema into a relational representation before it can convert data within the message. The same holds true when moving information from a relational database to an object-oriented database. Most EAI products break the message moving into their environment into a common format and then translate it into the appropriate message format for the target system [LIT01, p.238-239].

2.5.2. DATA CONVERSION

In the previous transformation example, information in Cust_No (alphanumeric and holding 10 positions) needs to be converted to all numeric with a capability of 20 positions. The alpha component of Cust_No must be dealt with in another matter. It is possible to change the nature of the target application to accept letters, or the alpha component can be either deleted or converted. Deleting characters or converting them into numeric representation is examples of data conversion. The key to successful data conversion is to determine the data formats of the source and target applications, assess the differences between them and adjust to them. For example, which data elements need to be extracted and converted, and where they ultimately need to be placed. Most EAI products are able to understand most message schemas through message identifications. Therefore, They automatically convert data to a workable format. Sometimes, however, it will be necessary to program a rule to address a specific data-type conversion problem. The conversion of numeric information to alphanumeric information, and vice versa, generally requires such a programmed rule.

Although, many formats exist within most application integration problem domains, for example: Alphanumeric, binary integers, floating point values, bit fields, IBM mainframe floating points, etc. In addition to these formats, there are a number of formatting issues to address, including the ability to convert logical operators, like bits, between systems and the ability to handle data types that are not supported in the target system. These issues often require significant customization in order to facilitate successful communication between systems.

In data conversion, values are managed in two ways: carrying over the value from the source to the target system without change, or modifying the data value dynamically. Either an algorithm or a look-up table can be used to modify the data value. One or more of the source application attributes may use an algorithm to change the data or create new. For example, attributes in the source application may represent "Amount sold" and hold the value 8. Another attribute, "Cost of Goods Sold", may contain value 4. However, in the target application, these attributes may have to populate a new attribute, "Gross Margin", which is the amount sold less the cost of the goods sold. In order to make this communication successful, the algorithm "Amount sold minus Cost of Goods Sold" must be applied. When using the look-up table scenario, EAI product may use a currency conversion table to convert dollars to euro, which may be embedded in a database connected to the EAI product. It may also invoke a remote application server function to convert the amount.

Algorithms of this type are nothing more than the type of data conversions that have been done for years when populating data warehouses and data marts. Now, in addition to using these simple algorithms, it is possible to aggregate, combine, and summarize the data in order to meet the specific requirements of the target application.

2.5.3. MAPPER TOOL

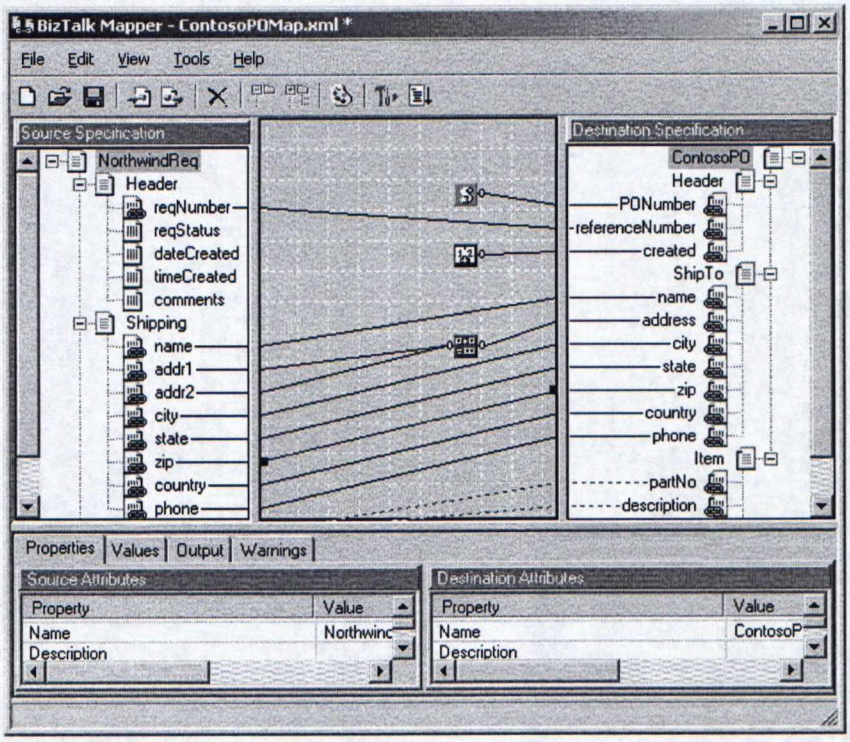


Figure 8. *Mapper Tool by Microsoft BizTalk Server [MIC03].*

Mapper tool is a graphical tool that allows users to map structure and data specifications between different messages. It visualizes the differences between the source and target

applications message presentation. Graphical mapper tools are without an exception based on XML and its transformation and parsing standards.

Mapper is a tool for creating maps, which define the correspondence between records and fields in one specification and the records and fields in another specification. A map contains a XSL style sheet that is used by the EAI product at runtime to perform the transformation described in the map. The tool enables the visual creation of maps by providing drag and drop functionality through which user can drag a connecting line from an element in the source window to the target element in target window [SAM02, p.276].

Behind the mapper tool there are engines that take care of the understanding of different presentations of data and making it possible to visually see the data attributes and the structure of the message of particular application. Conversion engines also take care of the creation of the conversion schema that is graphically implemented with the mapper tool. Mapper tool enables semantic and data conversion. Fields from source message can be combined, values can be fetched from other sources (database, etc.), and complex logical operations can be done, while creating the destination presentation. Data attributes can be converted as well.

2.5.4. TRANSFORMATION TECHNOLOGIES

XML Stylesheet Language Transformation (XSLT) is a standard written by World Wide Web Consortium (W3C). It is one of the preferred mechanisms for transforming content and application semantics as information moves between applications. XSLT is a language designed to transform one XML document into another, changing both its schema and content in the process. XSLT is a kind of text-processing system enabling the programmer to transfer XML documents or generate other standard markup languages such as Hypertext Markup Language, or any text, for that matter.

XML documents are like messages. Since each application has its own unique set of application semantics, documents moving from application to application need to be transformed. Both data structure and content must be semantically correct to load into the target application. XSLT can also perform other types of text-processing and transformation operations, including creating text-based standard data formats such as comma-limited files, Portable Document Format (PDF), or other standard formats that use text.

Before XSLT existed, most XML developers could process incoming XML documents only by creating custom applications that typically invoked one of two APIs: Simple API for SML (SAX) and Document Object Model (DOM). The SAX API is an event-based interface that uses a mechanism through which the parser notifies the application of each piece of information in the document as it is read. In the DOM API, the parser interrogates the document and creates

an object tree structure that represents the structure of the XML document in memory. From that point, a traditional program transforms the tree. The limitation of both approaches is the same. New program is always needed to transform a new XML document.

XSLT provides several advantages over SAX and DOM. XSLT's design is based on the fact that most transformation programs use the same design patterns and, therefore, can be automated using a higher-level, declarative language. The XSLT language is declarative because it describes the transformation behaviour rather than a sequence of instructions necessary to perform the transformation. XSLT describes the transformation, and then leverages the XSL processor to carry out the transformation. When XSLT is used the requirements of transformation can be expressed as a grouping of rules that define what output should be created when a particular pattern is encountered.

2.6. PROCESS MANAGEMENT LAYER

There are several interpretations and definitions for process management. In this context it is related to the integration processes. Term Business Process Management (BPM) and Process Management (PM) in literature is also used in this context.

2.6.1. GENERAL

Process management involves management of several components, including time, cost, quality, scope, human resources, communications, procurement, risk and integration. Process management is designed to streamline the internal and external processes of a company through a combination of several techniques, such as designing workflows, process rules and application components. Process management is a required element to direct data flow among the mass of packaged applications, legacy systems, services and partners. It aims to automate business processes from the beginning to the end, thereby eliminating the need for unnecessary manual intervention and removing all points of friction within the company and with its trading partners. It also enables the formulation of consistent business policies across the organization, which leads to consistent execution of business processes across all applications and channels. Process management allows processes to be executed, monitored, controlled and modified with greater ease than in the past. It also enables companies to track and monitor the actual state of the business process at any time.

Process modelling, EAI and B2Bi go hand-in-hand. These elements or resources included in a workflow definition should be interoperable and provide the workflow engine the capability to invoke, execute and/or launch the services provided by them [SAM02, p.132].

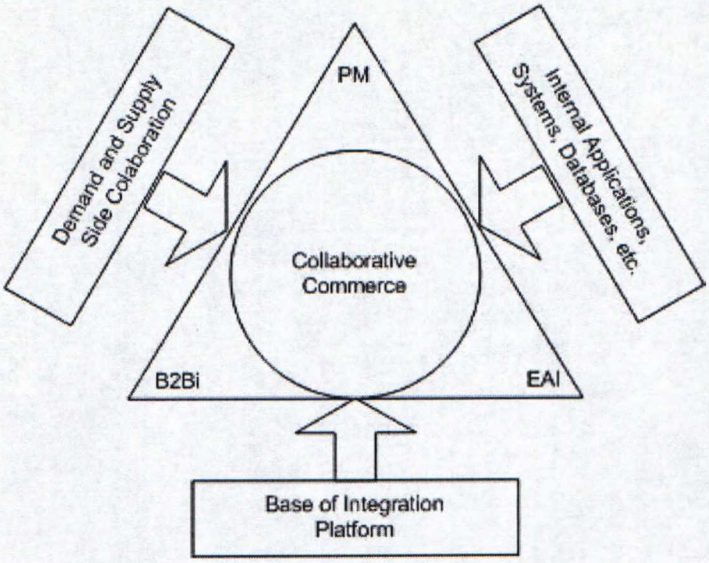


Figure 9. *PM, B2Bi and EAI enable the collaborative commerce [SAM02].*

2.6.2. PROCESS MODELLING TOOL

Process modelling tool is graphical and enables rapid visual design of business processes while hiding the implementation from the users. The tool should provide the functionalities of business rules and process rules that drive the working of a business process. Process modelling involves drawing a workflow diagram that links resources, logic and movement of information between systems. Sub-processes are used when the main process becomes too complex and needs to be broken down to keep it simple. Process modelling tool also provides administration functionalities such as monitoring the execution of the process, state of the process and acting in the events of exception.

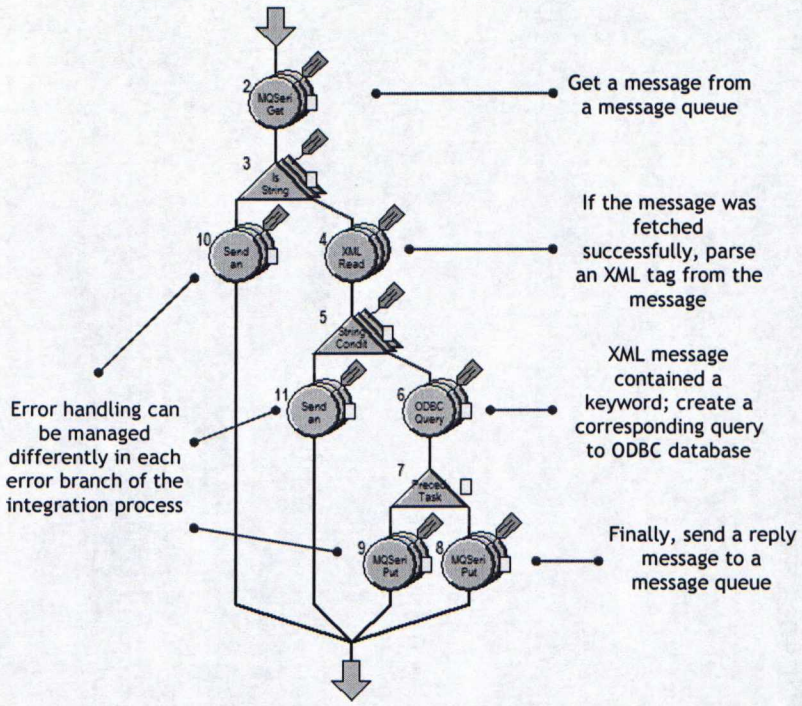


Figure 10. *Process modeling tool by FRENDS EAI Platform 4.1 [FRE03].*

The process-modelling tool provides an environment to bind the components (companies source and target systems) of a business process together to create the final model for the process. Advanced modelling tools also include XML-based rules repository, so that the rules can be shared across multiple applications in a single uniform format.

In process modelling also a process execution engine is required when attaching it to the application integration scene. Process execution engine is responsible for providing an environment for executing the process flow. For example, sequencing and invoking the required element as defined in the process design. The process execution engine manages a set of installed processes and coordinates their execution, thereby integrating applications and operations irrespective of the underlying platform. It interacts with the EAI tool to exchange the information with applications, control execution of individual steps and state transitions for running processes. All actions are persistently logged to the database so that the users can analyze the data through a reporting tool [SAM02, p. 139-141].

2.6.3. STANDARDS FOR BUSINESS PROCESS MODELLING

There are several practical challenges in the real world implementations on process modelling. There are no de facto standards available for process modelling. Multiple groups demand different levels of abstraction in the design and execution of processes. Different process modelling systems communicate by exchanging messages that have no standard formats, making process integration highly challenging. According to Gartner Group, through

2005 no standard will fully eliminate the requirement that enterprises must support multiple, incompatible process management aware applications [SAM02, p.147].

There is a need for universal language, which can be used to design, deploy, maintain and execute processes, such as order management, demand and forecast planning and new product development. This language should represent all the internal and external activities that support any company's business – for instance, communications with all internal and external applications, information dissemination, and transactions. It should be based on open architecture so that the process models among different process modelling systems can be exchanged and should enable the creation of a common business process repository.

Business Process Management Initiative

The business Process Management Initiative (BPMI) is led by several leading companies such as Hewlett-Packard Co., Art Technology Group, Sun Microsystems Inc., Tibco Software Inc., etc. BPMI organization is developing the Business Process Modelling Language (BPML) and Business Process Query Language (BPQL) – open standards. The first draft of BPML was made available to the public on March 8, 2001.

BPML is an XML-based meta-language for the design, definition, deployment and management of business processes that span multiple applications, corporate departments and business partners. BPML is actually an XML schema that provides a standard way to model mission-critical business processes. It aims to allow tighter integration and collaboration in business processes than the current application interface model. BPLM is based on a message-based model where the participants involved in a business process interact through the exchange of messages. The semantic and syntactic structure of messages is defined using BPML XML schema. BPML uses Xpath as the basis for its expression language. Xpath provides a declarative expression language with rich semantics for expressing condition logic, calculations and selecting predicates.

BPML considers e-Business processes as made of a common public interface. This enables the public interface of BPML processes to be described as ebXML business processes or RosettaNet Partner Interface Processes, independently of their private implementations.

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

BPQL intends to provide a standard interface for business process deployment in a process repository and execution in an integration system. The interface for process repository would enable users to manage the deployment of process models contained and managed by the repository. These process models could be exposed by the enterprises as Web services for process registration, advertising and discovery purposes. This language is still under development by BPMI [BPM03].

2.6.4. BUSINESS ACTIVITY MONITORING

Applications integration enables end-to-end integration of companies information systems and applications to reach into another's data. Business Activity Monitoring (BAM) enables real-time insights into what is really happening in complex enterprise processes. BAM offers a centralized sight to survey the steps of business process in the data level.

Integration product gives a good foundation for BAM while it is accessing to companies information systems already and managing the data exchange for the business processes.

Benefits of BAM are:

- Real-time visibility into business processes
- Faster, more effective responsiveness to business issues
- Increased business agility
- Improved business metrics and monitoring
- Enhanced risk management
- Maximized use of IT assets

2.7. TECHNOLOGY STANDARDS AND FRAMEWORKS

The Adapter layer should provide wide range connectivity via its adapters. The following list will introduce some of the standards that are supported by EAI product.

Middleware	IBM MQSeries, Microsoft Messaging, JMS, etc.
Remote execution protocols	Rexec, CORBA, COM/DCOM, Java RMI, SOAP/WSDL, JCA, etc.
Communication Platforms	TCP/IP, UDP/IP, X.25, Modems, SMS, File interfaces, etc.
Communication Protocols	Plain ASCII, Modem protocols, FTP(S),

	HTTP(S), SMTP, POP, IMAP, SNMP, LDAP, SCP, RAS, etc.
Language Adapters	ActiveX/COM, C/C++, Java, CORBA IDL, etc.
Database Adapters	ODBC, JDBC, etc.
Application Adapters	SAP, Siebel, PeopleSoft, Oracle, etc.

Table 1. EAI Product Connectivity

The nature of EAI is to integrate heterogeneous application and system environment that requires tremendous connectivity. At the following subchapters the most known and important standards are clarified.

2.7.1. ODBC

Open DataBase Connectivity (ODBC) is a widely accepted application programming interface (API) for database access. It is based on the Call-Level Interface (CLI) specifications from X/Open and ISO/IEC for database APIs and uses Structured Query Language (SQL) as its database access language. ODBC is designed for interoperability. It enables a single application to access different DataBase Management Systems (DBMSs) with the same source code. Database applications call functions in the ODBC interface, which are implemented in database-specific modules called drivers. The use of drivers isolates applications from database-specific calls in the same way that printer drivers isolate word processing programs from printer-specific commands. Because drivers are loaded at run time, a user only has to add a new driver to access a new DBMS; it is not necessary to recompile or relink the application [MYS03].

2.7.2. JDBC

Java DataBase Connectivity (JDBC) is a standard API for accessing relational databases from a Java program. This interface makes it easy to access a database because it provides an abstract layer that hides the low-level details. It provides interoperability and portability since it allows a single application to access multiple database management systems simultaneously. For example, a single application can query and manipulate a database in MySQL and a database in Oracle. Communication with a DBMS is enabled through method calls. These calls are passed to the driver, which in turn, translates them into DBMS-specific calls. The driver basically acts like a set of library routines. Therefore, to get your program to communicate with a particular DBMS, you need a compatible JDBC driver [MYS03].

2.7.3. CORBA

The Common Object Request Broker Architecture (CORBA) is a distributed object/component technology that is platform and programming language independent and allows remote object creation and method invocation. Object Management Group (OMG) created CORBA. It is a consortium formed in 1989 to define the standards and protocols required for distributed object systems in heterogeneous environments. CORBA was the first successful distributed architecture protocol to be developed [SAM02, p. 235].

CORBA applications are composed of objects, individual units of running software that combine functionality and data. For each object type an interface is designed in OMG Interface Definition Language (IDL). The IDL interface definition is independent of programming language. The interface is the syntax part of the contract that the server object offers to the clients that invoke it. Any client that wants to invoke an operation on the object must use this IDL interface to specify the operation it wants to perform [OMG03].

2.7.4. COM/COM+/DCOM

COM, COM+ and DCOM are the distributed pieces of the Windows Distributed Network Architecture (DNA) platform for building distributed applications in a Microsoft Windows environment [SAM02, p.241].

The Component Object Model (COM) is a component software architecture that allows applications and systems to be built from components supplied by different software vendors. COM is the underlying architecture that forms the foundation for higher-level software services, like those provided by OLE. OLE services span various aspects of component software, including compound documents, custom controls, inter-application scripting, data transfer, and other software interactions [MIC03].

At 1997 Microsoft created COM+ that is an extension to COM. COM+ builds on COM's integrated services and features, making it easier for developers to create and use software components in any language, using any software development tool. COM+ is designed to preserve and extend developers' current investments in COM. Applications created by COM technology will work in the COM+ environment [MIC03].

The Distributed Component Object Model (DCOM) is a protocol that enables software components to communicate directly over a network in a reliable, secure, and efficient manner. DCOM is designed for use across multiple network transports, including Internet protocols such as HTTP. DCOM is based on the Open Software Foundation's DCE-RPC specification and will work with both Java applets and ActiveX® components through its use of the COM.

Most distributed applications are not developed from scratch. Existing hardware infrastructure, existing software, and existing components, as well as existing tools, need to be integrated and leveraged to reduce development and deployment time and cost. DCOM directly and transparently takes advantage of any existing COM components and tools. Developers familiar with COM can easily apply their knowledge to DCOM-based distributed applications. Any component that is developed as part of a distributed application is a candidate for future reuse [MIC03].

2.7.5. .NET

The Microsoft .NET Framework is a platform for building, deploying, and running Web Services and Web applications. It provides a standards-based (HTTP and XML) and multi-language environment for integrating existing and next-generation applications and services.

The .NET Framework consists of three main parts: the common language runtime, a hierarchical set of unified class libraries, and a componentized version of Active Server Pages called ASP.NET. Microsoft's .NET basically extends older Windows Distributed Network Architecture (DNA) by incorporating additional integration functionality and Web services.

2.7.6. J2EE EJB

Java 2 Enterprise Edition (J2EE) is a set of coordinated specifications and practices that enable solutions for developing, deploying and managing multi-tier server-centric applications [SAM02, p.244]. J2EE is an independent framework where the underlying technology is Java.

Enterprise JavaBeans (EJB) is the core of J2EE framework and has become widely adopted server-side component architecture. The EJB specification is an industry initiative led and driven by Sun Microsystems in close cooperation with many vendors from the industry. Sun Microsystems owns the interactive and iterative process of defining, creating and publishing the specification, however, the feedback from the industry and developers are taken into account. It enables developers to write reusable portable server-side business logic for the J2EE platform. The key features of EJB are:

- EJB components are server-side components written entirely in the Java programming language.
- EJB components contain business logic only.
- The EJB server or container automatically manages system-level services, such as transactions, security, threading, persistence, etc.
- EJB components are fully portable across any EJB server and operating system.
- EJB components can interoperate over the network as CORBA objects.

2.7.7. JMS

Java Messaging Service (JMS) is a set of interfaces and associated semantics that define how a JMS client accesses the facilities of an enterprise-messaging product. Enterprise messaging is a part of the middleware layer, and JMS provides a common way for Java programs to create, send, receive, and read an enterprise messaging system's messages.

The JMS API specification defines two different messaging paradigms: point-to-point and publish/subscribe. These two paradigms represent the two leading models of messaging provided by existing messaging products.

JMS Specification defines many interfaces for message services but does not define an implementation. This is deliberately done by the JMS inventors to support the existing message handling software. The strategy is to let programmers develop JMS applications wrapping existing software infrastructures. This way the programmers need not to rewrite a whole application to implement JMS and will not have any dependency to a particular software vendor. Therefore JMS will be vendor neutral and comply with existing products (Sundsted, 1999 March).

2.7.8. Wireless Technologies

According to the Gartner Group, more than 108 million employees worldwide will regularly work outside a traditional office in 2003. It means that companies information systems have to be accessed outside the office. This requirement can be delivered in two ways. Information systems can provide user interfaces accessible wirelessly or the information exchanged between user and system can be delivered via EAI software that provides the wireless communication methods.

From the view point of EAI software this requirement is straightforward. EAI software has already an interface with the back-end information systems and what is to be built is the interface with the users. The interface with the remote users is usually based on protocols over TCP/IP such as HTTP and SOAP. This means that the user has a browser or other Graphical user interface in laptop or other portable device such as Nokia communicator or HP/Compaq iPaq and ability to connect to the data-communication network. Used wireless technologies are Wireless Local Area Network (WLAN), General Packet Radio Network (GPRS), GSM Data, Satellite and Wireless Application Protocol (WAP). Sometimes the exchange of data can be also achieved via Short Message Service (SMS). When communicating with field forces or wanting to send alarms and the amount of needed data to be received or sent is small, SMS is useful. Network operators provide gateways that enable efficient use of SMSs'.

Wireless technologies related to EAI are mostly based on Internet protocols and the most used technologies are WLAN and GPRS. These technologies provide adequate data-transfer rates and support the Internet protocols. In renewed GSM networks and third generation mobile networks the Enhanced Data rates for GSM Evolution (EDGE) is available, which enables high-speed packet data transfer with peak rates of 384 kbps. These enhancements in mobile networks' data-transfer capacity makes in desirable choice in future to access the companies information systems outside office. WAP "was" technology to provide access for mobile devices to access services and applications via bearer networks such as GSM, Code Division Multiple Access (CDMA) and next generation network standards. WAP has been constructed on modified Internet protocols and permits mobile devices on non IP-based networks to connect to the Internet. WAP wasn't commercially or technologically successful.

However, wireless networks and technologies are just components in the standards stack that EAI software must support.

2.7.9. WEB SERVICES

Web Services (WS) is a term that can be understood in many different ways. One meaning for it is the services that are built using XML based standards and services with Simple Object Access Protocol (SOAP) and Web Services Description Language (WSDL). The other "wider" meaning for the term is related to general web based service phenomena. Despite the variety of the explanations behind WS, the idea behind is clear.

Until now, a majority of web-based services have been targeted to users accessing them with web browsers. The next logical step is services provided by applications for other applications. This means that an application component can be a web-based service existing physically on the other side of the world accessed using normal web technology. WS is an architecture that enables the building of loosely coupled distributed systems using technology based on open standards that do not force lock-down to a particular programming language, component model and/or computing platform.

The problem with distributed software applications is the interoperability. Windows based systems usually use DCOM, Java based solutions utilize the Java RMI and Unix based systems might use socket based technology and application specific protocols. The other problem with applying these technologies within and between enterprises is caused by firewalls. Most of the traffic is blocked outside of the enterprise networks. WS is one possible solution to these problems. XML is platform independent and not tied up like binary protocols used with current distribution technologies like DCOM and RMI. XML messages are usually transferred over HTTP, SMTP, or FTP that means they will pass most firewalls without problems. The messages are also human readable and simple so error control is easy. XML solves many

problems that exist with current distribution technologies but it also has some itself. XML validation, data marshalling/unmarshalling and larger data packets to be transferred make XML based communication considerably less efficient than the use of binary protocols.

WS bring many benefits to EAI. It can reduce application development complexity and costs by providing standard external applications and whole enterprise systems. These technologies potentially lead to faster integration of applications and systems because of easier interoperability based on standard XML technologies.

Simple Object Access Protocol

Simple Object Access Protocol (SOAP) is an XML-based communications protocol used by applications to exchange information on the Internet. SOAP is a simple, platform independent and extensible protocol developed as a W3C standard for accessing Web services. It is normally transmitted over HTTP, making it easy to bypass corporate firewalls, but it can also use another transportation protocols if needed.

SOAP has many properties that make it well suited for universal distributed computing. It provides a mechanism for structuring messages into separate clear units. SOAP "fault" provides a mechanism for error handling that is capable of exchanging error-diagnostics information between web service participants. Other main properties of SOAP are extensibility, flexible data type representations, bindings to HTTP and other transport mechanisms and a convention of representing RPC calls with SOAP messages.

A SOAP message is constructed of the following elements:

- Envelope identifies the XML document as a SOAP message.
- Header contains header information.
- Body contains the actual payload information.
- Fault contains information on how to process errors.

There are few rules that SOAP message have to implement to be considered valid. The envelope element is required as it defines an XML document as a SOAP message. The message envelope also needs to use the soap namespace to be considered a valid SOAP envelope. SOAP messages cannot contain DTD references or any XML processing instructions.

Web Services Description Language

Web Services Description Language (WSDL) is a proposal for a W3C standard by Ariba, IBM and Microsoft for describing and locating Web services. WSDL is an XML document describing the location of the service and the operations offered by it. WSDL defines both the

service interface and implementation details of it. WSDL is basically the Interface Definition Language (IDL) of the Web Services world.

The WSDL definition is not a full service description, but it covers the lower level details of the service that is needed to describe the service interface. WSDL is used to describe and answer to the questions what, how and where. It describes what functionality is provided by the service, how the service can be accessed, such as what kind of input is needed from the client, and where the service is located on the web.

The four main elements in WSDL message are types, message, portType and bindings:

- Types element is used for defining data types. XML schema syntax is used in these definitions for platform neutrality.
- The message elements are used to describe the messages used inward and outward communication. These can be compared to parameter definitions in traditional programming languages.
- The portType defines the collection of operations or methods that the service exposes to external users and which messages are to be used in the operation. These can be compared to functions in traditional programming languages.
- A Bindings element is used to define the message format and protocol details for each port.

Universal Description, Discovery and Integration of Web Services

Universal Description, Discovery and Integration of Web Services (UDDI) is a platform-independent framework used for registering Web Services, discovering registered services and integrating business services by using the Internet. The directory stores web service interface description WSDL files and it uses SOAP to communicate with external clients. UDDI is the first Internet standard that enables companies and organizations to programmatically publish information of their products and services using a single, open and secure environment.

Technically UDDI is offering two different kinds of registries: a business registry and a reference type registry. The business registry allows business and organizations to register public information describing themselves and their services that are available for external use. The reference type registry stores information about standards, abstractions and services and lets businesses, standard bodies and industry groups refer them by assigning unique identifiers to them. Each of these registries has a separate root data structure – businessEntity on the business side and tModel with the reference types.

The business entity information, stored as businessEntity, is logically divided into white, yellow and green pages. The white pages contain general contact information to the entity registered such as name, address and contact information. The Yellow pages contain information on the type and location of the services provided by the entity such as a moving service and grocery store. The Green pages contain information on how the service should be invoked. Thai can be an URL or other form of information allowing access to the service.

The reference types registry uses the tModel data structure for storing definitions for service types and various other abstractions. These definitions are called technology models and they can be used freely to specify service type definitions that can later be used by others and combined into new definitions.

3. BUSINESS-TO-BUSINESS INTEGRATION

3.1. INTRODUCTION

B2Bi is about integrating the business processes between companies. The main idea is to model, automate and streamline the business processes. B2Bi is roughly divided into two categories that are public and private processes.

The public process is the part that is defined by the B2Bi standards. It focuses on the business side processes where the dialect, vocabulary and behaviour how to do business between companies is the most challenging issue. B2Bi also defines the technological issues such as communication protocols and security. It gives companies a common standardized way to communicate and do business with each other. It is a standard business interface that makes it easy for any company to connect with it.

The private process is not defined by the B2Bi standards. It is the process that links the public process with the real data from the back-end systems. Companies have different kinds of back-end systems that present the information required in business transactions. A typical business process involved in B2Bi is completed by multiple application systems. The first challenge that companies face in their implementation plan of B2Bi is the fact that the internal applications must be integrated. The part of the private process in B2Bi is the integration of internal applications and providing the required information for the public processes.

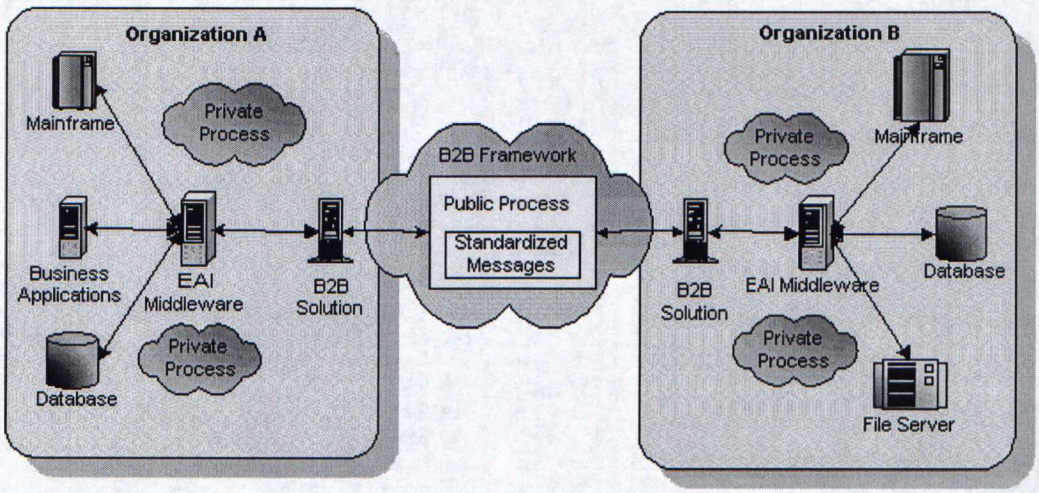


Figure 11. *Private and public processes [HAA03].*

The internal application integration is done with EAI. So the public process in B2Bi is served via EAI, which takes care of the implementation of the private process. EAI integrates the whole heterogeneous internal application infrastructure and provides the required data and information that the public process of B2Bi needs.

In this thesis the application integration within and between companies is divided into two categories that are EAI and B2Bi. From the perspective of B2Bi it means that EAI takes care of the internal application integration as well as the implementation of the B2Bi private process. The role of B2Bi is the to enable the public process and communications to business partners. This can be delivered with different products for EAI and B2Bi that communicate with each other, or with a product that offers functionality for both EAI and B2Bi.

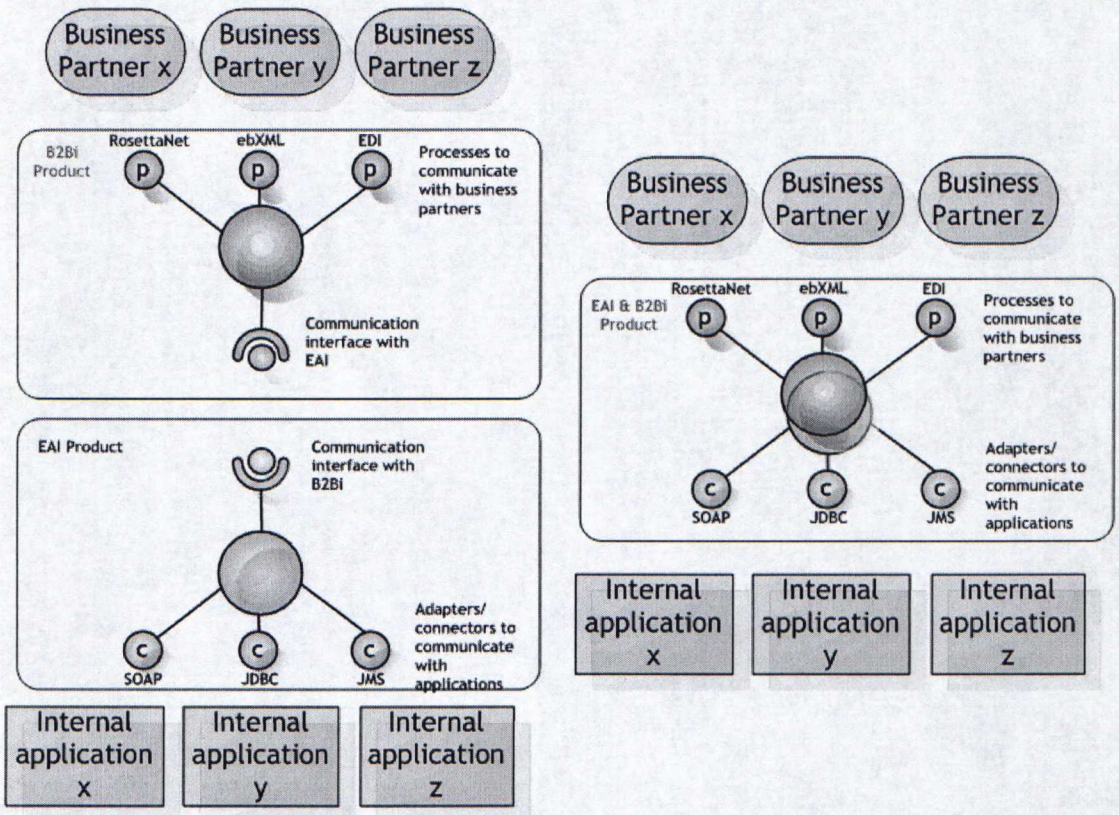


Figure 12. *EAI and B2Bi correlation.*

This chapter focuses on the public processes of B2Bi standards whereas the technology used in B2Bi mainly the same as in EAI. The technology, standards and methodology EAI are discussed in chapter 2 "Enterprise Application Integration".

3.2. Convergence of EAI and B2Bi

The lines between B2Bi and EAI are becoming blurred. Traditional EAI has been characterized by the integration of back-end applications and front-office activities. Many of the traditional EAI services are necessity in many of today's B2Bi models. The entire EAI, B2Bi, and enterprise software marketplace is converging and merging rapidly.

This convergence is most apparent in the latest generation of EAI products where B2Bi and EAI software share many common features, including:

- Data transformation
- Use of application specific adapters
- Intelligent routing
- Workflow and process management

Some of the e-Business software vendors such as IBM and Microsoft are combining EAI and B2Bi services under one product. These vendors are taking the EAI integration process to the next logical step by wrapping and publishing traditional EAI method-level components as Web services via SOAP and UDDI [SAM02, p.121].

The major area of divergence of EAI and B2Bi is security. With EAI solutions companies do not have to worry about encryption, firewall implementations, cross-organization distributed applications, or business-partner management, all of which generate difficult challenges in B2Bi. In addition, B2Bi implementation is more complex from a global perspective, as it has to deal with different international laws regarding the use and export of cryptography technology.

Another area of divergence is in the process orientation. Some of the EAI solutions do not base on process orientation. The exchange of data and information between applications is not bonded tightly with business processes. The idea of B2Bi is to automate the business processes between companies - it is all about process orientation.

As we look ahead, the ultimate application integration solution will be some hybrid of EAI and B2Bi, providing integration within and between enterprises by using similar, compatible infrastructure.

3.3. Business Process Modelling

Business process models define how business processes are described. Business processes represent the "verbs" of electronic business and can be represented using modelling tools. The specification for business process definition enables an enterprise to express its business

processes so that they are understandable by other enterprises. This enables the integration of business processes within an enterprise or between enterprises. Business process models specify business processes that allow business partners to collaborate. While business practices vary from one organization to another, most activities can be decomposed into business processes that are more generic to a specific type of business. This analysis, utilizing business modelling, will identify business processes and business information meta models that can likely be technically modelled and even standardized.

Human-to-human business exchange is successful and efficient because business partners agree upon the process from the most basic level: We produce and hear sounds, use a common alphabet to create words, apply grammatical rules to words to make dialog, use dialog to form business processes, and conduct business through an instrument such as a telephone. Applications exchange information over the Internet in electronic business. HTML/XML function as the alphabet, and electronic commerce applications serve as the instrument by which e-business processes are transmitted. The lack of agreement on the words, grammar and dialog that constitute e-business processes illustrates the need for standards. The figure 13 illustrates how the human-to-human business exchange correlates with B2Bi model as seen by RosettaNet. RosettaNet dictionaries provide the words, the RosettaNet Implementation Framework (RNIF) acts as the grammar and RosettaNet Partner Interface Processes (PIPs) form the dialog.

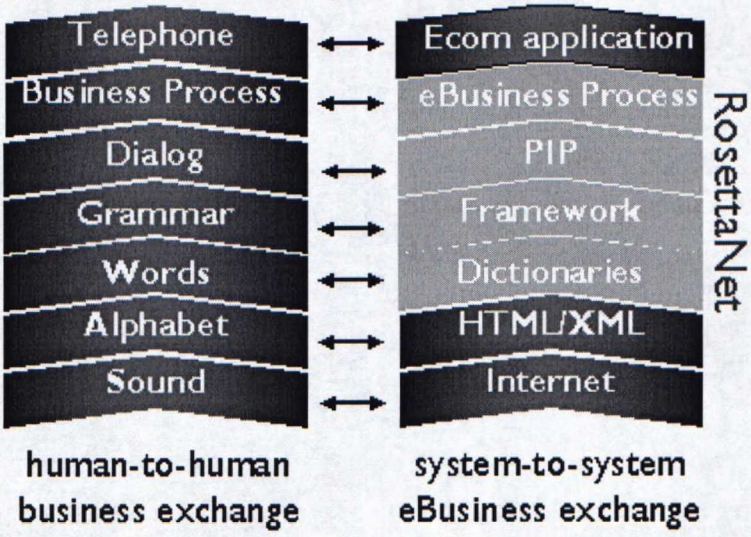


Figure 13. *Electronic business exchange with RosettaNet [ROW03].*

3.4. SECURITY

As mentioned, the major area of importance in B2Bi is security. In B2Bi solutions companies have to worry about encryption, firewall implementations, cross-organization distributed applications, or business-partner management, all of which generate difficult challenges in

implementation. In addition, B2Bi implementation has to deal with different international laws regarding the use and export of cryptography technology.

The main idea in B2Bi security is to assure integrity, confidentiality and availability:

- The assurance that data, programs and other system resources are protected against malicious modification or destruction by unauthorized persons, programs or systems.
- The assurance of confidentiality of "classified" and "secret" data such as salary information, personnel and trading partner contract information, etc.
- The assurance that authorized users has guaranteed access to the service infrastructure. Making sure that for example denial-of-service attack does not disrupt the availability.

Three related technologies make up a powerful toolset to secure a B2Bi solution [EBX01]:

- Encryption,
- Digital signatures and
- Certifications.

The Public Key Infrastructure (PKI) brings these tools together in a comprehensive system of digital certificates and certificate authorities.

3.4.1. ENCRYPTION

Encryption is a science of essential importance in keeping data secret. The basic idea of encryption is that a sender applies an encryption function to the original "plaintext" message, for example containing credit card information, with resulting ciphertext message. This ciphertext message is sent over the Internet to the receiver, which applies decryption function to the ciphertext message and recovers the original plaintext.

The encryption/decryption process generally depends on a key, which is shared between the sender and the receiver, and a known encryption algorithm. There are three basic approaches:

- Secret (symmetric) key encryption,
- Public (asymmetric) key encryption and
- Cryptographic checksums.

In secret key encryption the same secret key is used to encode and decode the message. Symmetric key algorithms are attractive because they require relatively modest computing resources, and when used with keys of sufficient length, they produce virtually uncrackable

ciphertext. The critical issue is how to keep the key secret. In practise secret key encryption is usually combined with public key encryption. Most known symmetric algorithms are the Data Encryption Standard (DES) and RC4, which is used in Secure Socket Layer (SSL).

In public key encryption there are two keys involved. The first key is called the public key, which the receiving party makes known publicly. The second key is called the private key, which is not published, and only the receiving party knows it. The public key is used to encrypt the message, and the private key is used to decrypt it. The sender and receiver only need to agree about the encryption key. There is no need to send the decryption key between two parties. Therefore, there is much less risk of the decryption key falling into the wrong hands. Public key encryption schemes are relatively costly in terms of computing resources because it has to be computationally very difficult to deduce the private key from the public key in order to maintain the security. Because of this cost factor, public key schemes are often combined with secret key scheme. This approach is taken in SSL, the secure email standard S/MIME and Pretty Good Privacy (PGP). Most known public key encryption algorithm is RSA, named by its inventors Rivest, Shamir and Adleman.

Cryptographic checksums provide a method that allows checking the integrity of received data. Basically, an algorithm is used to generate a value called checksum from the data that is wanted to protect. Receiver uses the same algorithm to the data in order to see if the data is tampered with. The algorithms used to generate the checksum are called hashing functions. Well-known hashing standards are Message Digest 5 (MD5) and SHA1 (Secure Hash Algorithm 1).

3.4.2. Digital Signatures

Combinations of the cryptographic techniques described in the previous chapter are applied in the area of digital signatures. A digital signature is basically a block of data attached to a message. The idea is to bind this data to a particular individual or entity. This mechanism can provide message integrity and authentication. It typically involves a signing key, which is private to a sender, and a signature verification key, which is made public. The binding is such that a receiver or an independent third party can verify the signature. A widely used method for computing digital signatures is RSA combined with hashing algorithms such as MD5 and SHA1.

3.4.3. Certification

One important aspect of both encryption and digital signatures is the question of validity. This issue is addressed by Certification Authorities (CAs) such as VeriSign. A certification authority will vouch for the authenticity of a public signature and supply a proof of this through a

public-key certificate. Typically each digital certificate contains a public key, a unique identifier for the subject owning the key and the signature of the CA.

The use of certificates and CAs is becoming the dominant model of assuring trustworthiness. It is relatively known nowadays because of its wide use in web browsers.

3.5. STANDARDS

This chapter introduces the standards used in B2Bi. EDI, RosettaNet and ebXML standards are discussed in more detail while they represent the most used and open standards.

- EDI is the first widely used standard, a foundation, for business document exchange between trading partners.
- RosettaNet represents a vertical e-business standard that has been widely adopted in high-tech industry.
- EbXML is a generic framework that provides a horizontal view to e-business. It has no constrains in vertical businesses.

RosettaNet and ebXML are possible candidates to achieve the "de facto" status in the B2Bi standardization field.

3.5.1. EDI

The first era of e-business was conducted using Electronic Data Interchange (EDI) through Value Added Networks (VANs). EDI involves electronic exchange of routine business transactions. These transactions include documents, such as purchase orders, invoices, inquiries, planning, acknowledgements, pricing, order status, scheduling, test results, shipping and receiving, payments and financial reporting. Through EDI these highly secure documents are exchanged in a compressed, machine-readable form over private Value Added Networks (VANs). EDI trading partners are seldom connected to each other directly. Instead, they use the service of VAN whereby each trading partner connects to the VAN. The service provider of VAN manages the connections to all trading partners [SAM02, p.154].

EDI works by providing a collection of standard message formats and an element dictionary that can be exchanged via any electronic messaging service. X12 EDI is based on standards developed according to the guidelines of the American National Standards Institute (ANSI). The ANSI committee ensures that everyone using a process such as EDI follows the same rules and methods, making the program universally accessible. As a result of the standard, all businesses using EDI share common interchange language, which minimizes the need for change in internal data processing systems. There are also national standardization

organizations that develop and maintain the national message formats for EDI such as Organisaatioiden Välinen Tiedonsiirto (OVT) in Finland.

In order for EDI to work effectively, standards must ensure that the information being transmitted is universally acceptable. The United Nations Electronic Data Interchange for Administration, Commerce and Transport (UN/EDIFACT), more commonly EDIFACT, standards are designed to work across industry, company and national boundaries. Changes and updates to the standards are made by consensus, reflecting the needs of the entire base of EDI users rather than those of a single organization, business sector or nation. The concept is to create a single international EDI standard [UNE03].

EDI has been used in companies for the past 30 years. It has helped the simplifying and speeding up customers, suppliers and other partners transactions. Today's B2Bi standards are based on XML, which has lead to the discovery of XML/EDI. XML and EDI will coexist for a long time. Their interoperability is one of the key success factors for large, medium and small sized companies doing business on the Internet. EDI/XML translation software enables the conversion of any X12 or EDIFACT EDI documents into the XML based dialects, standards such as BizTalk, cXML, ebXML and RosettaNet PIPs.

XML/EDI is a fusion of many concepts:

- Uses the XML and XSL for data interchange and presentation.
- Provides a standard for formatting documents.
- Can be integrated with traditional EDI.
- Can be used with all standard Internet transport mechanisms such as HTTP, FTP and SMTP.
- Uses modern programming tools such as Java and ActiveX to allow data to be shared between programs.
- Uses agent technologies for data manipulation, parsing, mapping and searching.

Limitations of EDI

The setup procedure for traditional EDI is expensive, complex and time consuming. The inherent nature of EDI communication requires the trading partners to synchronize their internal systems and business processes with those of their partners. The different applications used by the trading partners use different schemas and data exchange protocols. The format and data content of the files generated by these systems also vary widely. More than 300 documents have been defined as standard X12 EDI transactions, but only few of these are applicable to different industry verticals.

Key limitations are:

- EDI is primarily a one-to-one technology, while a typical B2Bi over the Internet requires many-to-many connectivity.
- Adding trading partners under traditional EDI requires customized mapping of each new partner's document formats.
- As usage of VANs is expensive, EDI messages are compressed using algorithms. This makes the message only machine readable and extremely difficult to debug. Just a small mistake in message formation can lead to hours of problem detection.
- EDI requires installation and maintenance of dedicated servers that are very expensive. Also the operational costs are high. This is one reason why small and medium sized companies do not adopt EDI.

Advantages of EDI

More than 300.000 companies worldwide have adopted EDI [SAM02, p.156]. In Finland 10 % of companies employing more than 5 people use EDI. For large companies the penetration is more than 40 %. Retail and industry are the most common users of EDI [TIL03].

Key advantages are:

- Long history as a standard. More than 30 years of simplifying and speeding up transactions on the private and public sectors.
- It has been well adopted by large corporations all over the world.
- XML/EDI is more dynamic, less costly and simpler than traditional EDI. It reduces the entry costs for small and medium sized companies.
- XML standards significantly reduce the number of trading partner specific maps required.

3.5.2. ROSETTANET

General

The name RosettaNet has two meanings: it denotes both a B2B standard and the organization developing the standard. The organization is a self-funded, not-for-profit consortium of major solution providers and information technology, electronics, and semiconductor manufacturers. Over all there are more than 400 participants. The consortium was formed in June 1998 to define B2Bi data and process standards. The pilot implementations of RosettaNet were a year after the founding, in June 1999, and the first production system implementations were made in October 2000.The consortium is constantly extending the standard from the basic B2B interactions, such as purchase order (PO) management and

shipment notices, to encompass invoicing, collaborative forecasting, and payments. In August 2002 RosettaNet merged with the Uniform Code Council (UCC) and become a subsidiary of it. The UCC is also a not-for-profit standards organization, founded in 1970. Its history is in the grocery and retail business but nowadays it has support across a variety of industries. Many international research reports predict that RosettaNet to be the probable de facto electronic business standard in the area of process integration.

The RosettaNet standard is divided into four groups: Dictionaries, RosettaNet Implementation Framework (RNIF), Partner Interface Processes (PIP), and product and partner codes. The standard itself combines these four modules to a framework that should be open and interoperable. Breaking the standard into modules has the advantage that when new and innovative solutions arise, a module can be replaced with a better solution easier than changing the whole standard.

- ***Dictionaries.*** These are two RosettaNet dictionaries functioning as the words and vocabulary of communication by providing common properties for B2Bi processes. The technical directory provides a common language for defining products and services. The business dictionary designates the properties used in business activities, such as business data entities and properties.
- ***RosettaNet Implementation Framework (RNIF).*** The RNIF is the communication protocol for RosettaNet. It assumes the use of XML and the Internet and specifies the transport, routing and packaging, and security.
- ***Partner Interface Process (PIP).*** PIPs are the core of the RosettaNet standard. They define business processes between companies: each supported B2Bi process is described as a sequence of steps required to complete the process. PIPs are the public part of a B2Bi process. The PIPs are categorized in eight clusters, such as order management and product information, according to their type.
- ***Product and partner codes.*** These codes are not actually defined by RosettaNet but are commonly used business and product identifiers that are utilized to expedite the adoption of RosettaNet. The codes like Data Universal Numbering System (D-U-N-D) are used as identifiers in the PIPs.

From technological standpoint RosettaNet builds up on existing solutions. RosettaNet uses HTTP as the basic communication protocol, Secure Socket Layer (SSL) for communication security, and Public Key Cryptography Standard number 7 (PKCS#7) for digital signatures. Experiences from EDI are utilized in creating the RosettaNet processes and messages.

RNIF v. 2.0

The RosettaNet business documents are exchanged in packets, which contain headers, content, attachments, and digital signatures. The parts of a packet are XML documents,

except if attachments are used. The attachments may be whatever data format. The parts are packaged together using MIME or, if the packet is digitally signed, S/MIME. The packet is known as a "RosettaNet Business Message". The XML documents are validated against appropriate Document Type Definition (DTD).

- Preamble Header is used to identify the version of the standard that this message complies with. The values of the preamble are set by the conversation initiator and must not be changed in the following, related messages.
- Delivery Header specifies the routing and message instance information. The routing is based on the message sender, receiver, and a unique instance identifier.
- Service Header provides the process context for a message with properties like the PIP code, initiating partner, message sender's role, the number of attachments, etc.
- Payload part contains the service content and zero or more optional attachments. Unlike the headers whose format is defined in the RNIF specification, the service content changes according to the particular PIP in use. The service content is either an action or signal message. An action message is a message of a business nature such as a purchase order or order acknowledgement. The action message guidelines and DTDs are specified as a part of the corresponding PIP.

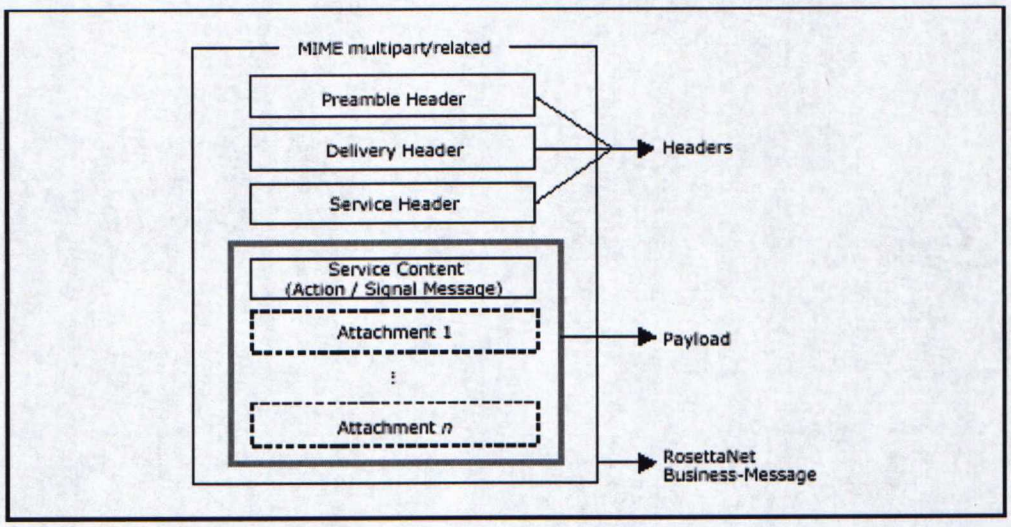


Figure 14. *Parts of a RosettaNet Business Message [ROS03, p.10].*

A signal message is a response to an action. The signals are positive and negative acknowledgements and only actions are acknowledged. The signal messages are specified in the RNIF.

PIP Model

Since RosettaNet is about processes rather than data, the most important aspect of RosettaNet is the development of common Partner Interface Processes (PIPs) [LIT01].

The real value of PIPs is the ability to allow manufacturers to seamlessly add new products to their partners' catalogues. For example adding a new part number using a common format and information interchange standard.

A PIP specification comprises three views to the business process it describes [ROS03, p.6]:

- The Business Operational View (BOW) provides the semantics of the business data entities and their exchange flow between roles during normal operations. The information for a BOV is derived from the business analysis during the development of a PIP. For example, in purchase order (PIP 3A4) the BOV states that there are two participants, a buyer and a seller.
- The Functional Service View (FSV) defines the network component services, agents, and functions required to execute PIPs. These include all transaction dialogs in a PIP protocol. The FSVs are semantically derived from the BOV and include two major components, which are the network component design and the network component interactions.
- The Implementation Framework View (IFW) defines the network protocol message formats and communications requirements between protocols supported by network components. The specifications are based on the requirements in the BOV, FSV, and the format of the service content. These messages are exchanged when software programs execute a PIP.

The PIPs can be divided into four categories according to their process model and message choreography that are one-action and two-action asynchronous, and one-action and two-action synchronous [ROS03].

Synchronous messages are possible only through HTTP connections as the synchronous reply is sent back on the same HTTP connection. If the message was received through another protocol, it must be treated in any case as an asynchronous message. An asynchronous reply is sent separately, from the transport protocol point of view, from the request. A request sender can ask for synchronous response by setting an extension header in the HTTP POST message.

A one-action PIP comprises only one sent action message while in a two-action PIP the action messages are sent between participants. In a one-action PIP the initiating trading partner sends a request and the recipient responses with an appropriate signal message, usually a receipt acknowledgement. An example of such a PIP is 3B2 (Notify of Advance Shipment).

This is used to communicate shipment contents and expected arrival date among other shipment information. A two-action PIP is basically two one-action processes executed in series. An example of a two-action PIP is 3A4 (Request Purchase Order). It is used to issue and confirm purchase orders.

3.5.3. ebXML

Electronic Business XML (ebXML) is a framework created by the United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) and the Organization for the advancement of Structured Information Standards (OASIS). They established the ebXML Working Group to develop a technical framework that enables XML to be utilized in the exchange of all electronic business data and electronic business interoperability.

The ebXML framework, an 18-month project, was delivered in May 2001. It is a specific project that has been completed and there is no ebXML organization, not even a virtual organization. ebXML represents a collaborative effort of two organizations, their members and supporters [EBX01]:

- UN/CEFACT is a global organization responsible for worldwide policy and technical development in the area of trade facilitation and electronic business for trade facilitation. It is well known for delivering the UN/EDIFACT framework for EDI.
- OASIS is a not-for-profit, member-based consortium that identifies, builds and maintains industry-standard specifications for interoperability. It has a strong background in providing a forum for developers and vendors to identify and resolve interoperability issues regarding XML software products. OASIS also maintains the XML.org portal, which is a registry for XML schemas and XML news source aimed toward industry.

The idea of ebXML is to provide a modular suite of specifications that enables enterprises of any size to conduct business over the Internet. It offers companies a standard method to exchange [EBX03]:

- Business messages,
- Conduct trading relationships,
- Communicate data in common terms and
- Define and register business processes.

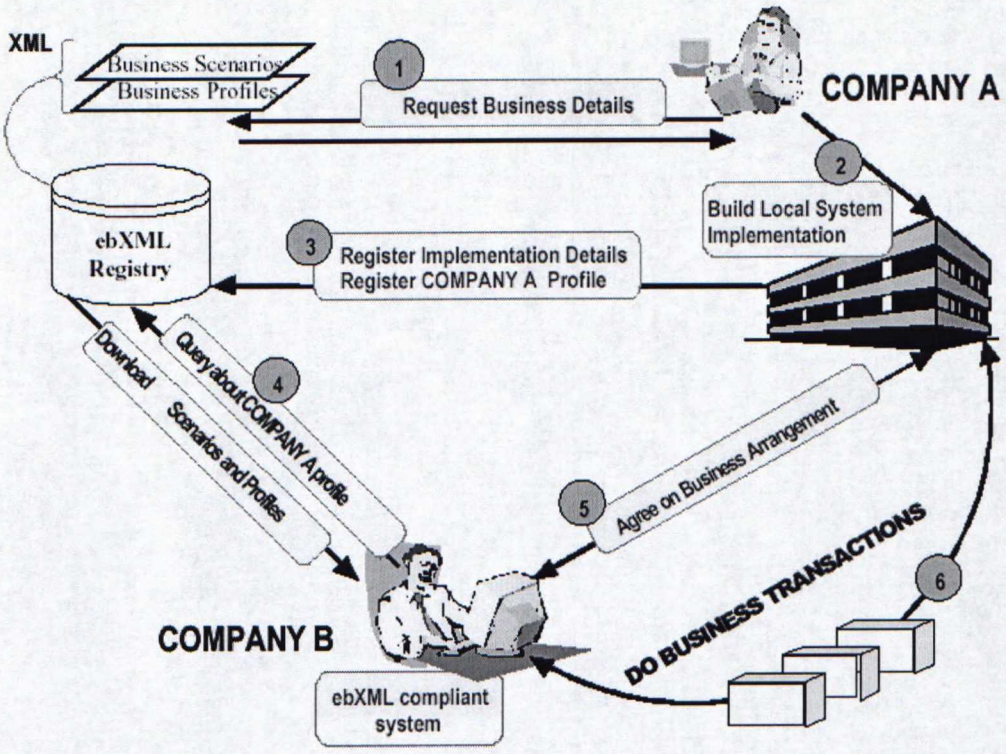


Figure 15. Scenario of the interaction of two companies conducting eBusiness using ebXML [EBX03].

There are several scenarios how companies could automate and streamline their business with trading partners and other constrain-groups. Figure 15 shows step by step how an e-Business scenario is delivered by means of ebXML [EBX01]:

1. Company A has become aware of an ebXML Registry that is accessible on the Internet.
2. Company A, after reviewing the contents of the ebXML Registry, decides to build and deploy its own ebXML compliant application. Custom software development is not a necessary prerequisite for ebXML participation. ebXML compliant applications and components may also be commercially available.
3. Company A then submits its own Business Profile information (including implementation details and reference links) to the ebXML Registry. The business profile submitted to the ebXML Registry describes the company's ebXML capabilities and constraints, as well as its supported business scenarios. These business scenarios are XML versions of the Business Processes and associated information bundles (e.g. a sales tax calculation) in which the company is able to engage. After receiving verification that the format and usage of a business scenario is correct, an acknowledgment is sent to Company A.
4. Company B discovers the business scenarios supported by Company A in the ebXML Registry.

5. Company B sends a request to a Company A, stating that they would like to engage in a business scenario using ebXML. Company B acquires an ebXML compliant application. Before engaging in the scenario Company B submits a proposed business arrangement directly to Company A's ebXML compliant software Interface. The proposed business arrangement outlines the mutually agreed upon business scenarios and specific agreements. The business arrangement also contains information pertaining to the messaging requirements for transactions to take place, contingency plans, and security- related requirements.
6. Company A then accepts the business agreement. Company A and B are now ready to engage in e-Business using ebXML.

EbXML Components

EbXML framework builds up from components, which together form a foundation for enabling e-Business between companies. The components of ebXML framework are [EBX01]:

- ***Business Process Specification Schema (BPSS)*** is an XML-based specification language that formally defines the public business processes. It focuses on the collaboration of trading partners, the binary collaborations that these trading partners are engaged in bilaterally and the business transaction activities they perform in the context of those collaborations.
- ***Core Components (CCs)*** provide the business information that is encoded in business documents that are exchanged between business partners. These should be able to be re-used and assembled from public or private registries. CCs are tagged with universal identifiers and they facilitate multilingual environments. The work on CCs was not completed during the 18-months project and it is currently being worked on as part of follow-on activities.
- ***Registry and Repository*** is a platform independent open storage for housing the description and facilitating the exchange of business artifacts, and discovering businesses via collaboration profiles.
- ***Trading Partner information: Collaboration Protocol Profile (CPP)*** and ***Collaboration Protocol Agreement (CPA)*** are XML documents that encode a party's e-business capabilities or two parties' e-business agreements. These are closely related to BPSS. With the messaging service, CPPs and CPAs provide configuration information to generic, high-level ebXML compliant B2Bi products. With the registry these support business discovery and the process of setting up new e-business relations.
- ***Messaging Services*** provide a general purpose messaging mechanisms such as transport, routing and packaging. This is quite mature specification that is required

by many other components. The ebXML messaging service is layered over SOAP with attachments, and can transport arbitrary types of business content.

- ***Security*** is a topic that is important to all components and is critical for a production of e-business system.

Business Process Specification Schema

The ebXML Business Process Specification Schema (BPSS) provides a standard framework for business process specification. As such, it works with the ebXML Collaboration Protocol Profile (CPP) and Collaboration Protocol Agreement (CPA) specifications to bridge the gap between Business Process Modeling and the configuration of ebXML compliant e-commerce software, for example an ebXML Business Service Interface, as shown in Figure 16. This is an example how the public process (BPSS, CPP and CPA) is mapped to the private process Business Service Interface (BSI).

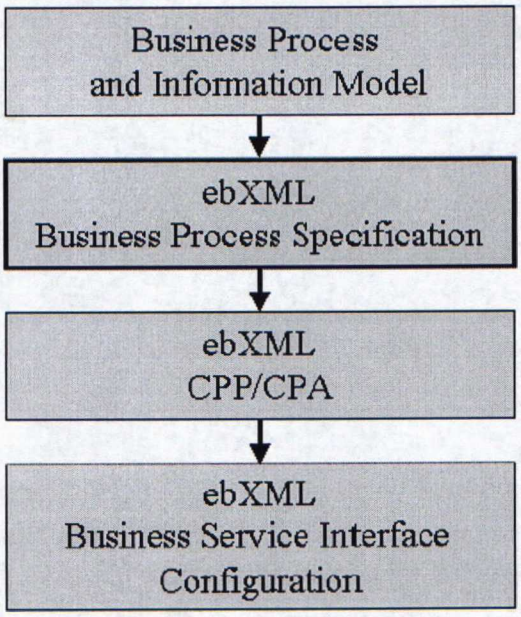


Figure 16. Relation of Business Process Specification and BSI Configuration [EBP01].

Using Business Process Modeling, a user may create a complete Business Process and Information Model. Based on it and using the ebXML BPSS the user will then extract and format the nominal set of elements necessary to configure an ebXML runtime system in order to execute a set of ebXML business transactions. The result is an ebXML Business Process Specification. Alternatively it may be created directly, without prior business process modeling.

An ebXML Business Process Specification contains the specification of business transactions and the choreography of business transactions into business collaborations. This ebXML Business Process Specification is then the input to the formation of ebXML trading partner CPPs and CPAs. In turn these CPPs and CPAs serve as configuration files for ebXML Business Service Interface software.

The architecture of the ebXML BPSS consists of the following functional components:

- UML version of the BPSS
- XML version of the BPSS
- Production Rules defining the mapping from the UML version of the BPSS to the XML version
- Business Signal Definitions

Together these components allow specifying all the run time aspects of a business process model [EBP01].

Core Components

Business transaction between trading partners are expressed as an exchange of electronic business documents. The idea of Core Components is to offer re-usable components to business documents. A Core Component captures information about real world business concepts and the relationships between those concepts. It forms business information objects and a contextual description that describes how a core entity may be used in a particular ebXML e-business scenario.

A Core Component can be either an individual piece of business information, or a natural family of business information objects that may be used as a part of business document. The users of ebXML may adopt and/or extend components from the ebXML Core library.

Registry and Repository

An ebXML Registry provides a set of services that enable the storing of business artifacts, sharing of information between trading partners and discovering businesses via collaboration profiles. A Registry is a component that maintains an interface to metadata for a registered item. Access to an ebXML Registry is provided through interfaces exposed by Registry services. Repository is the storage for the actual content.

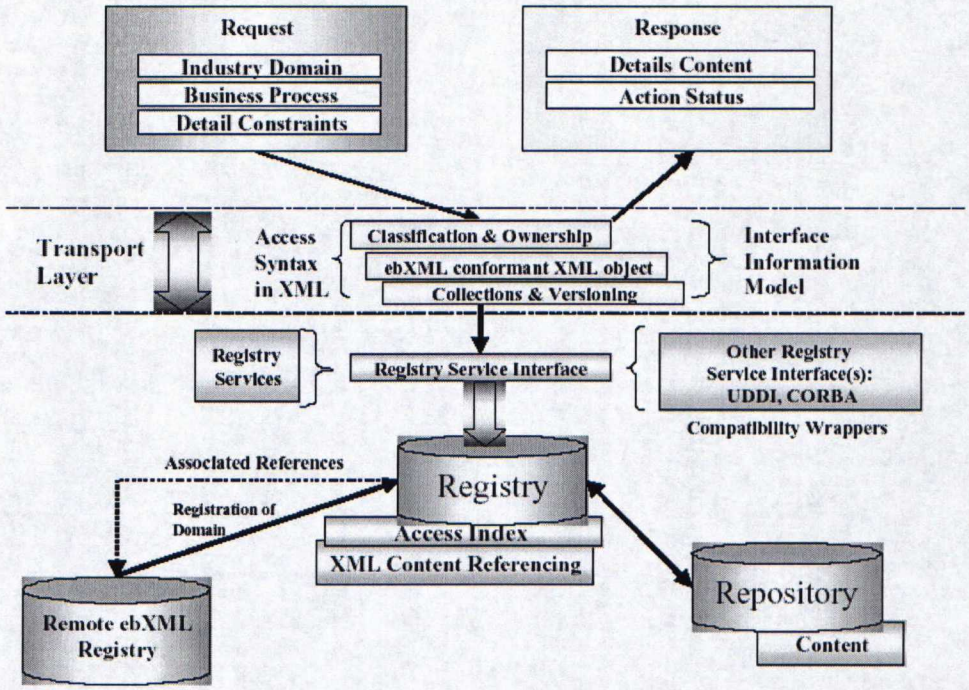


Figure 17. *Overall Registry Architecture [EBA01].*

Trading Partner Information

The Collaboration Protocol Profile (CPP) describes the specific capabilities that a trading partner supports as well as the service interface requirements that need to be met in order to exchange business documents with that trading partner. The CPP contains essential information about the trading partner such as contact information, industry classification, supported Business Processes, Interface requirements and Messaging Service requirements. CPP's may also contain security and other implementation specific details. Each ebXML compliant trading partner should register the CPP(s) in an ebXML compliant registry service, thus providing a discovery mechanism that allows trading partners to find one another and discover the business process that other trading partners support.

A Collaboration Protocol Agreement (CPA) is a document that represents the intersection of two CPP's and is mutually agreed upon by both trading partners who wish to conduct e-Business using ebXML.

A CPA describes the Messaging Service and the Business Process requirements that are agreed upon by two or more trading partners. Conceptually ebXML supports a three level view of narrowing subsets to arrive at CPA's for transacting e-Business. The outer- most scope relates to all of the capabilities that a trading partner can support, with a subset of what a trading partner "will" actually support.

A CPA contains the Messaging Service Interface requirements as well as the implementation details pertaining to the mutually agreed upon Business Processes that both trading partners

agree to use to conduct e-Business. Trading partners may decide to register their CPA's in an ebXML compliant Registry Service, but this is not a mandatory part of the CPA creation process.

Messaging Services

The ebXML Message Service mechanism provides a standard way to exchange business messages among ebXML trading partners. The ebXML Messaging Service provides a reliable channel to exchange business messages without relying on proprietary technologies and solutions. An ebXML message contains structures for a Message Header, which is necessary for routing and delivery, and a Payload section.

The ebXML Messaging Service is conceptually broken down into three parts:

- An abstract Service Interface
- Functions provided by the Messaging Service Layer
- The mapping to underlying transport service(s)

The relation of the abstract Interface, Messaging Service Layer, and transport service(s) are shown in Figure 18.

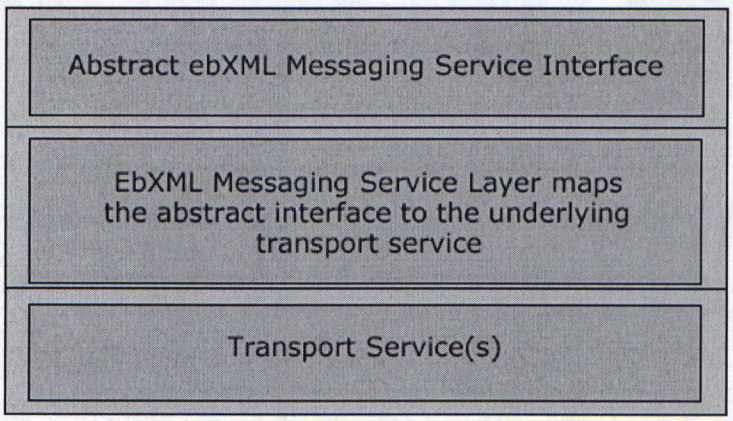


Figure 18. *ebXML Messaging Service [EBX03].*

The ebXML Messaging Service provides a secure, consistent and reliable mechanism to exchange ebXML messages between users of the ebXML infrastructure over various transport Protocols such as SMTP, HTTP/S, FTP etc. It prescribes formats for all messages between distributed ebXML Components including Registry mechanisms and compliant user applications. The ebXML Messaging Service does not place any restrictions on the content of the payload.

The ebXML Messaging Service supports one-way and request/response (either synchronous or asynchronous) message exchanges. It also supports sequencing of payloads in instances where multiple payloads or multiple messages are exchanged between trading partners.

The ebXML Messaging Service Layer enforces the "rules of engagement" as defined by two trading partners in a CPA. The definition of these ground rules can be formed in many ways, including formal CPA(s), interactive agreements established at the time a business transaction occurs (e.g. buying a book online) or other forms of agreement. The Messaging Service Layer functions take care that these rules are executed. Any violation of the ground rules result in an error condition, which is reported using the appropriate means.

Security

The main technical challenge of secure technology within an open framework like ebXML has two sides. Such framework has to demonstrate sufficient flexibility to evolve with new demands placed upon it and it must also meet the requirements of interoperability. EbXML is based on XML technology that is also the fundamental component of all security-related initiatives within ebXML. Since security initiatives and security standards in XML are not as far advanced as some of the other specifications and standards, many of the security initiatives around ebXML are still very much under development.

The important security initiatives of ebXML are [EBX01]:

- ***Security Assertion Markup Language (SAML)*** is an initiative to define an XML-based framework, which facilitates exchange of authentication and authorization information. SAML is currently in its early stages but it is a very promising security initiative that groups and controls various other initiatives in order to come up with an uniform security architecture.
- ***XML Access Control Markup Language (XACML)*** is a language for defining security measures based on access limitation via an access control list. It is the part that takes of the policy decision point within the SAML security environment.
- ***XML Digital Signature*** is an initiative of the W3C and IETF. XML signatures provide integrity, message authentication and/or signer authentication services both XML and non-XML data.
- ***XML Key Management Specification (XKMS)*** is an XML-based solution for the management of public keys. VeriSign together with Microsoft and WebMethods submitted the XKMS proposal to the W3C in March 2001.

3.5.4. OTHERS

Web Services

Web services are a collection of standards that define web-enabled application interfaces. The standards are introduced in chapter 2.4.4. "Web Services". It means that application can publish an identity for itself, describe the services it offers, and provide an interface for other applications. Therefore it is often discussed alongside EAI and B2Bi. Web services are seen as an easy-to-use standard way for creating necessary interfaces for various enterprise applications. The idea is good and would benefit many companies. However, Web services technologies are still young and there are many open issues.

In B2Bi Web services lack the required data syntax and semantics and business process definitions. It seems that Web services are too simple and low-level solution for B2Bi. Still the simplicity, ease-of-use and tested standards behind it are the strengths of Web services. When Web services come more popular and enterprise applications implement Web services interfaces, it may be used as a data transfer level in integration solutions within and between companies.

cXML

Commerce XML (cXML) was created in a unique collaboration between buyers, suppliers, and Internet technology companies. More than 40 organizations were involved in the process including leading e-commerce companies such as Ariba, Microsoft and WebMethods. cXML is a standard for business-to-business electronic commerce. cXML defines a request/response process for the exchange of transaction information. These business processes include purchase orders, change orders, acknowledgments, status updates, ship notifications and payment transactions. The cXML specification was made publicly available in March 1999. At the end of 2003 the version of cXML is 1.2 [CXM03].

NACS

The National Association of Convenience Stores (NACS) is an U.S. based industry trade association representing over 2,000 convenience retailers operating over 100,000 stores worldwide since 1961. The NACS Technology Standards Project has started in 1995. It focuses on four key areas: electronic data interchange, common data communications between the back office and the point-of-sale terminal, electronic payment systems, and device interfaces with the point-of-sale terminal [NAC03].

Open Buying on the Internet (OBI)

The Open Buying on the Internet (OBI) Consortium is a non-profit organization dedicated to developing open standards for business-to-business Internet commerce. The OBI Consortium is an independent collaborative managed by CommerceNet. Membership in the consortium is open to buying and selling organizations, technology providers, financial institutions, and other interested parties on an annual fee basis. The OBI standard is an open, flexible framework for business-to-business Internet commerce solutions. The initial focus of OBI is on automating the high-volume, low-dollar transactions between trading partners that account for 80% of most organizations' purchasing activities. The OBI standard provides access to easy-to-use, open, standards-based Internet purchasing solutions for the procurement of goods and services.

IXRetail

International XML Retail Cooperative (IXRetail) was founded in 1999 by the National Retail Federation of U.S. It builds on the Association for Retail Technology Standards (ARTS) Data Model to develop XML schemas and message sets to ease application to application integration within a retail enterprise. IXRetail include XML schemas for digital receipt, transaction log, stored value, inventory, remote equipment monitoring, and payment in retail industry. Most of the worldwide known retailers and software producers for retail industry are members of ARTS such as IBM, Microsoft, SAP and Shell [NFR03].

BizTalk

Microsoft's BizTalk initiative has three fundamental components: BizTalk Framework, BizTalk Server Suite and BizTalk Schema Library. BizTalk Framework is a platform-neutral e-commerce framework that is based on XML schemas and industry standards. BizTalk server is a flagship product of the Microsoft .NET enterprise server family. BizTalk library is a repository of published schemas submitted by participating companies that is maintained on the http://www.biztalk.org/ site. Through this site, the schema for any BizTalk message is universally accessible.

BizTalk is Microsoft's initiative to build a set of tools and standards for B2Bi [MIC03].

OASIS UBL TC (Universal Business Language Technical Committee)

Organization for the Advancement of Structured Information Standards (OASIS) is a not-for-profit, global consortium that drives the development, convergence and adoption of e-

business standards. It was founded in 1993 and has more than 600 corporate and individual members in 100 countries around the world.

The purpose of the OASIS Universal Business Language Technical Committee (UBL TC) is to develop a standard library of XML business documents, such as purchase orders, invoices, etc., by modifying an already existing library of XML schemas to incorporate the best features of other existing XML business libraries. The TC will then design a mechanism for the generation of context-specific business schemas through the application of transformation rules to the common UBL source library. UBL is intended to become an international standard for electronic commerce freely available to everyone without licensing or other fees.

Historically, UBL was developed in order to provide the document schemas for ebXML, but ebXML and UBL are not formally connected. UBL can be (and was designed to be) a business vocabulary for ebXML, but ebXML can also be used with other business vocabularies as well. Companies like SUN Microsystems and SAP are members of OASIS UBL TC.

Release of UBL 1.0 Beta is scheduled for mid-November 2003, beginning a three-month public implementation phase before finalizing the specification for submission to OASIS in February 2004 [OAS03].

papiNet

Paper Commerce on the Internet (papiNet) is an international initiative with implementations in 4 continents and over 80 companies. It was started in 1999 and it focuses on the paper and forest products industries. papiNet is the global initiative to develop, maintain and promote the implementation of standard electronic transaction standards to facilitate the flow of information amongst the parties engaged in the buying, selling, and distribution of forest, paper and wood products. The standard includes common terminology and standard business documents that support the entire supply chain. The papiNet standards are open and freely available [PAP03].

Adeona

Adeona is a not-for-profit consortium that drives the development, adoption and maintains the registry services of e-business protocol for printing media industry. Adeona was founded in Finland, 2003. Current members of Adeona are Xerox, Canter and FRENDS Technology.

Adeona is based on ebXML. It is an e-business standard that describes the data communication, data transfer and technical integration between publishing and printing systems. The idea of this e-business standard is to automate the process of generating a print from a published material. This process includes the automatic bidding of different printing vendors that are members of the Adeona registry, and it enables automatic delivering of the

required material with proper publishing parameters to the printing vendor to complete the task.

Adeona is in piloting phase at the moment. More information is available in www.adeona.org .

4. INTEGRATION MARKET AND PRODUCTS

4.1. MARKET

Application integration is a vast business. It's the largest segment in the middleware market, which also includes integration brokers, application servers, portal products, messaging middleware and transaction processing monitors. In 2002, product license revenue for integration brokers was more than 1.37 billion USD, representing a 23 percent share of the middleware segment of the market [Gartner Dataquest, June 2003].

The integration market has experienced tougher trading conditions in the years 2000-2003 and has begun to consolidate after a period of dramatic growth in the nineties. Gartner group expects market consolidation to continue well into 2005. Consolidation among integration vendors is part of a much broader trend in the software industry. In the years 2000-2003, approximately 25% of the leading software companies were involved in a merger, acquisition, or divestiture. In the years 2004-2007, the pace is only expected to accelerate, reaching as high as 50 percent.

Here are some reasons for the consolidation in the integration market:

- IT spending has slowed dramatically.
- Vendor selection decisions are clearly shifting toward lower-risk choices of suppliers, benefiting the titans such as IBM, with their broad product portfolios and more stable financial situations.
- New competition has emerged in the integration segment from Web services and application server vendors such as Microsoft, BEA, SAP, and Oracle. This has forced existing integration broker suite vendors to review and modify their current go-to-market strategies. Application server vendors will be continuing to add more integration products into their product portfolios and moving into application platform suites.
- The emerging category of application platform suites is the result of the convergence of application servers, portal products, and integration brokers. The convergence is expected to continue through 2004, which means even more market segment and vendor consolidation in the near future.

As buyer behaviour continues to stay cautious and more focused on the foreseeable future, larger companies in the integration software market, such as IBM, BEA, SAP, Microsoft, and Oracle, are going to make progress at the expense of smaller vendors.

Position	Vendor	Market Share %
1.	IBM	19.1
2.	TIBCO	9.3
3.	WebMethods	8.9
4.	SeeBeyond	4.9
5.	Sun	4.5

Table 2. Top-five Integration Broker Vendors by Market Share in 2002 [COR03].

In 2002 IBM won the new-license revenue market share battle in the integration broker suite market. The result of the IT spending slowdown for most integration software vendors was that IBM and BEA gained new-license revenue market share at the expense of TIBCO, SeeBeyond, Mercator, and Vitria. WebMethods maintained its market position. Other large vendors, such as BEA and Microsoft, gained market share and broke into the top 10. While these vendors caused only little market-share disruption in 2001, Microsoft started to eat away at the value proposition of high-end integration broker systems with "good-enough" technology and aggressive pricing for low-end integration broker technology. BEA took advantage of its installed base of application servers and expanded its integration product portfolio, gaining momentum in the integration market [COR03].

The market shares for different integration broker vendors in Finland are quite difficult to clarify. Official rankings or market shares are not available. Application server vendors such as BEA have advantage of its installed base also in Finland. IBM has several users of their MQ Series queue system but the amount of IBM's business integration suite is not as wide. FRENDS Technology's FRENDS EAI Platform has about 50 customers and Microsoft's BizTalk approximately 30 in Finland. I believe Finland's market will follow the same trend as Gartner stated. BEA and Microsoft will gain market share as well as the local player FRENDS Technology.

4.2. Cost of Integration

The cost of application integration can be divided roughly in three components:

- Architecture
- Integration and
- Operation.

Architecture costs are capitalizable costs related to the initial deployment such as integration development, execution and operations environments. They include the license cost that is negotiated with the software vendor, the cost of new hardware required to develop, run and

monitor integrations and the cost to implement architectural software and hardware. Accenture's Christy Bass states that roughly 80 percent of architecture costs are incurred within six months of implementation while additional expenses may be incurred for hardware or licenses as usage spreads [BAS03]. Architectural costs are driven by the complexity of the integration software and the number of discrete business entities to which it is deployed. While comparing to custom integration, the architectural investment is much higher for an integration software solution than custom integration. This is mainly because of the license costs.

Integration development costs are separate from the architectural costs. Integration costs are often capitalized and relate to the development of interfaces and collaborations between systems. Integration costs are variable and are driven by the number of interfaces that are developed. Integration costs with integration software are generally between 25 and 40 percent lower than with custom integration [BAS03]. Development is less expensive because adapters come pre-built with the integration software architecture and the architecture provides a graphical interface in which to perform mapping as well as many pre-built functions. Examples include message transport, guaranteed delivery and process control including the ability to re-use process steps.

Another important point is that because all applications communicate with a common middleware, far fewer interfaces need to be developed. It's also important to note that the integration software contains functionality that makes developing, running and monitoring the integration system much easier. This includes business process management facilities, reusable transformation and formatting components, auditing, logging, and debugging functions; pre-built adapters with upgrades for future releases of software, message receipt acknowledgement and much more.

Operating costs are expensed and include on-going operations and maintenance of the integration system for architecture and integrations. The number of interfaces that need to be maintained generally drives operating costs. Integration software generally provides a 50 to 80 percent reduction in application maintenance cost by reducing the number of interfaces that need to be maintained and offloading much of the costs of interface maintenance onto the integration software solution provider [BAS03].

4.3. PRODUCTS

This chapter presents three products designed for application integration: IBM WebSphere Business Integration Solution, Microsoft BizTalk Server, and FRENDS EAI Platform. The reasons for choosing these products are following:

- IBM is the market share leader in the integration market segment and it is a lower-risk choice for suppliers because of its stable financial situation and historical background.
- FRENDS EAI Platform is a local player in Finnish market with own product designed in Finland. It is an interesting and substantial challenger in Finnish integration market.
- Microsoft BizTalk Server is an interesting product. It is expanding also to smaller markets because of the aggressive prising and the worldwide distribution of Microsoft products.

4.3.1. IBM WebSphere Business Integration Solution

General

IBM is the market leader in integration broker market with its Message Queue Series (MQSeries) product family [COR03]. IBM is also one of the biggest, oldest and most known information technology products and services providing corporation with stable financial background. These issues make IBM very desirable and low risk choice for companies to choose MQSeries product family to its integration solution.

Product components

The MQSeries product family builds up from four main products, which are WebSphere MQSeries, WebSphere MQ Integrator Broker, WebSphere MQ WorkFlow and WebSphere InterChange Server. This is also called the "Business Integration" –solution or WebSphere Business Integration Server. It can operate in multiple different platforms, such as Microsoft Windows, Linux and different UNIX and mainframe platforms. The strategy of IBM is also to use open standards. In application integration domain IBM uses the J2EE platform as integration platform and an open-source project called ECLIPSE is aiming to deliver a development-tool-framework for WebSphere solutions development.

The functionality of WebSphere Business Integration Server components is following:

- WebSphere MQ provides the infrastructure for message based data transfer, programming interfaces for applications, and a wide range of application adapters.
- WebSphere MQ Integrator Broker enables the message transformations, intelligent routing and the controlling of workflows.
- WebSphere MQ WorkFlow is a graphical tool for describing and executing graphically designed workflows.
- WebSphere InterChange Server is for process automation that manages multiple discrete business applications as one. It provides multi-threaded and concurrent business logic execution, assures data integrity, contains a high-availability

configuration option and provides a web browser-based system and configuration management.

The idea of the IBM's Business Integration solution is to serve the whole application integration domain by providing the following functionalities via modular architecture:

- Business process modelling and simulation (WebSphere MQ WorkFlow)
- Business process management and monitoring (WebSphere InterChange Server)
- Message Broker functionality (WebSphere MQ Integrator Broker)
- Reliable transport layer (WebSphere MQ)
- Connectivity to applications via adapters
- Interfaces to partners via B2B gateway

IBM also provides a lighter version of the Business Integration solution that is called WebSphere MQ Express. It is a new product that is aimed to middle-sized companies via its "lighter" prise and some restrictions in the functionality and the scale of use. It enables the use of IBM's integration products in a smaller environment and with smaller needs.

Pricing and Licensing

The prices of IBM's products are always based on independent customer agreement that is called Passport Advantage Agreement (PA – agreement). The level of PA-agreement is based on the customer's IBM product purchasing volume. The prises are always tailored for each customer separately depending on the PA – agreement level.

The principles of licensing are following:

- Server Processor License Unit must be purchased for every processor that exists in the development configuration/environment of IBM's integration product. There is an exception with WebSphere Business Integration Server. The Integrator Broker, WorkFlow and InterChange Servers can be installed to three different servers with the prise of one Server Processor License Units.
- WebSphere Business Integration Server Adapters are purchased for each application separately and it has no prising relation to the amount of processors used. If one adapter, for example SAP R/3 adapter, is used to connect several "different" SAPs inside the same installation, unlimited use of adapter can be purchased by price of two adapters.
- All solutions can be clustered to achieve high usability. Clustering does not require any additional licenses while it is used only for fail-over as a back-up server. The back-up server cannot be used for load balancing.

Component	Prise
WebSphere Business Integration Server (including MQ WorkFlow, InterChange Server and MQ Integrator Broker)	100.000 EUR
WebSphere MQ / message queue	5.000 EUR
WebSphere Business Integration Adapters / application	40.000 – 100.000 EUR
WebSphere Business Integration Monitor	60.000 EUR
WebSphere MQ Express	From 4.000 EUR

Table 3. *IBM's Business Integration solution Pricing in Finland [IBM03].*

4.3.2. FRENDS EAI PLATFORM

General

FRENDS Technology is one of the leading application integration software providers in Finland with its FRENDS EAI Platform. FRENDS Technology is Finnish company and it has been focusing on application integration since early 1990's. FRENDS Technology has subsidiary in U.S. and more than 200 installations in Finland, Germany and U.S. Most important customers of FRENDS Technology are Metso, Neste, Stockmann and Tradeka.

Product components

FRENDS EAI Platform is a packaged software product designed for application integration. It operates on Windows operating system and it is highly scaleable. It consists of three functional layers that are connector, data transformation and integration process layer [FRE03]:

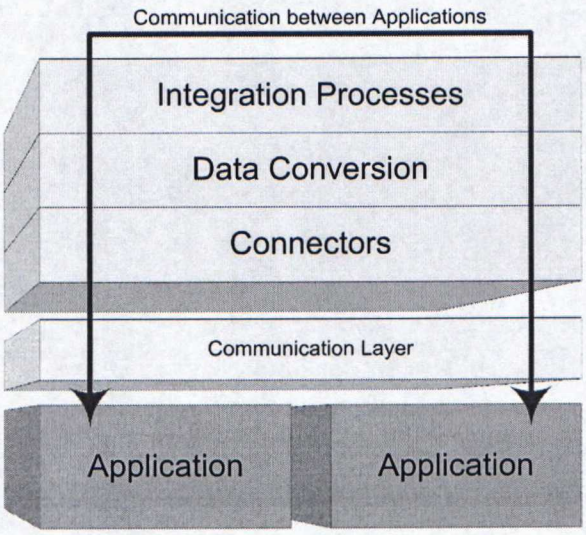


Figure 19. *FRENDS EAI Platform layers [FRE03].*

- ***Connector layer*** provides connectors that either connect directly to the applications or communicate with applications via a certain communication protocol. The connectors of FRENDS EAI Platform support e.g. modem-based, X.25 or TCP/IP based communication protocols. There are also a number of middleware and database interfaces available. Web Service protocols are also fully supported.
- ***Data conversion layer*** provides data conversion utilities such as support for different kinds of text-based conversions, XML parsing (SAX, DOM) and XML conversions (XSLT). Third-party conversion software and mapper can be attached as a part of the integration solutions based on FRENDS.
- ***Integration Process layer*** contains the integration application logic: rules with instructions for converting and routing data, scheduling and triggering different integration transactions, as well as the monitoring and reporting functionality required in a well-defined integration solution. Integration Process layer offers intuitive graphical tools for generating these rules – no traditional programming languages are needed.

Pricing and Licensing

The prising of FRENDS EAI Platform is straightforward compared to IBM and Microsoft. It is based on three pricing components:

- Amount of CPUs running FRENDS EAI Platform. CPUs are counted from the installed FRENDS server(s) that form the integration solution.
- Amount of integrated applications connected to the integration solution. Integrated applications and/or databases are for example SAP R/3, Oracle database, i2 supply chain management software, etc.
- Amount of connection points connected to the integration solution. Connection points are considered when the task is to process same functionality with several external systems, for example updating product information to several convenience stores.

Pricing table shows the approximated prices for different components in Finland.

Component	Price
FRENDS EAI Platform	12.500 – 6.250 EUR / CPU
Integrated Application	7.500 – 4.500 EUR / application
Connection Point	750 – 500 EUR / connection point

Table 4. *FRENDS EAI Platform Pricing*

4.3.3. MICROSOFT BIZTALK SERVER

General

Microsoft is the largest software company in the world, which makes it a very interesting vendor in application integration market.

Microsoft's BizTalk initiative has three fundamental components: BizTalk Framework, BizTalk Server Suite and BizTalk Schema Library. BizTalk Framework is a platform-neutral e-commerce framework that is based on XML schemas and industry standards. BizTalk server is a flagship product of the Microsoft .NET enterprise server family [MIC03]. BizTalk server is the product that enables the EAI and B2Bi functionality.

Product components

BizTalk server is a flagship product of the Microsoft .NET enterprise server family. It enables the EAI and B2Bi integration of enterprises. It includes a suite of tools and services for visually designing, building and maintaining processes and securely integrating applications, independent of their operating system, programming model, or programming language.

The BizTalk Server Suite toolset includes four modules: BizTalk Messaging Services, BizTalk Orchestration Services, BizTalk Mapper and BizTalk Editor:

- ***BizTalk Messaging Services*** include sending, receiving, parsing and tracking documents. It also contains receipt generation and correlation among data mapping, integrity and security.
- ***BizTalk Orchestration Services*** enable the creation and orchestration of business processes. The services include the integration of long-running processes with the applications that run those business processes.
- ***BizTalk Mapper*** is a tool for creating maps, which define the correspondence between records and fields in one specification and the records and fields in another specification. The tool enables the visual creation of maps by providing drag and drop functionality through which user can drag a connecting line from an element in the source window to the target element in target window.
- ***BizTalk Editor*** is a tool for creating, editing and managing document specifications, which are based on industry standards such as EDIFACT and XML or on plain flat files. Document specifications define a way to translate between the document's original data format and the server's internal XML format. It is also used for directly uploading BizTalk Framework compliant XML-schemas to BizTalk library through Web Distributed Authoring and Versioning. The trading partners can then download and access this schema.

Pricing and Licensing

BizTalk Server is available in four editions: Enterprise Edition, Standard Edition, Partner Edition and Development Edition.

- BizTalk Server 2002 Enterprise Edition is targeted at large organizations, trading hubs, and digital marketplaces. This edition has support for integrating unlimited internal applications with unlimited trading partners over the Internet with multiple processors.
- BizTalk Server 2002 Standard Edition is designed for small and medium-sized organizations. This edition has support for integrating up to five internal applications with up to 10 external trading partners, such as exchanges or digital marketplaces. It does not support multiprocessor or clustered deployments.
- BizTalk Server 2002 Partner Edition is designed for trading partners who need to quickly connect their top customer or electronic marketplace. This edition has support for integrating up to two internal applications with up to two external trading partners. It does not support multiprocessor or clustered deployments.
- BizTalk Server 2002 Developer Edition provides developers with all the tools they need to rapidly integrate applications, trading partners, and to orchestrate business processes.

BizTalk Server requires always Microsoft SQL Server that must be purchased separately. BizTalk Server accelerators are designed to accelerate the development of data transaction solutions that rely on BizTalk Server.

Pricing table shows the prices for different editions. Each component is presented by prise per processor.

Component	Enterprise	Standard	Partner	Developer
BizTalk Server	24.999 USD	6.999 USD	999 USD	499 USD (per developer), free
BizTalk Accelerator for Financial Services	19.999 USD	4.999 USD	-	-
Accelerator for HIPAA 2.0	19.999 USD	4.999 USD	-	-
Accelerator for RosettaNet	19.999 USD	4.999 USD	-	-
Accelerator for Suppliers	4.999 USD	4.999 USD	-	-
Adapter for MQSeries	14.999 USD	14.999 USD	14.999 USD	-
Adapter for SAP	14.999 USD	14.999 USD	14.999 USD	-
Adapter for Web Services	Free	Free	Free	-

Table 5. *Microsoft BizTalk Server Pricing*

It is important to take both the BizTalk Server accelerator license price and the BizTalk Server license price into consideration, because they must be licensed on a one-to-one basis. For every processor of BizTalk Server in the configuration, you must license one processor of a BizTalk Server accelerator.

4.3.4. PRODUCT COMPARISON

The following table shows the key differences and similarities between these three products: IBM Business Integration Suite, FRENDS EAI Platform and Microsoft BizTalk. Each product gets ranking for different factors. The product that best fulfils the factor will be ranked number one (1.). The comparison factors, presented in the table, are clarified in more detail in the next chapter "Integration Opportunities and Challenges".

Factor	**IBM**	**FRENDS**	**Microsoft**
Process Orientation	3.	**1.**	2.
Error Handling and Monitoring	2.	**1.**	2.
Performance and Scalability	**1.**	2.	2.
License prise	3.	**1.**	2.
Vendor Liability	**1.**	3.	2.
R&D	2.	3.	**1.**
Visibility	2.	3.	**1.**

Table 6. *Product comparison table.*

Process orientation is extremely important in integration. The integration level should support the business processes. The integration with FRENDS EAI Platform is based on graphical tools. With these tools all integration events are built in work flow manner as shown in figure 10. This tool also enables real-time monitoring of the execution of these integration processes. It also enables error handling with rapid application development tools by adding error handling and retries to the error branches of the process. This is the reason why FRENDS is ranked number one in these categories. In performance and scalability IBM is naturally ranked number one because of its knowledge and experience among large multinational companies. Also its stable financial situation and historical background raises IBM number one in vendor liability. The license prising comparison in previous chapters showed that FRENDS is the number one. Microsoft is the largest software company in the world. It has largest resources for marketing, distribution and R&D. This means a potential market share growth for their rather new integration product BizTalk.

5. INTEGRATION OPPORTUNITIES AND CHALLENGES

This chapter concludes the issues that generate the opportunities and challenges in application integration. My purpose is to deliver these issues from two perspectives: technology and business. These conclusions can be used as a list by the decision-makers from both, the side of technology and business, for the purpose for evaluating the relevance of application integration for their respective companies.

There are certain issues in the application integration domain that can be understood as both challenges and opportunities. This is why these issues are delivered separately in sub-chapters. Every sub-chapter analyses whether the issue is a challenge or an opportunity. Main focus is on the challenges and opportunities presented by EAI and B2Bi products.

According to the EAI survey by Forrester Research, the following factors, in order of their importance, make integration challenging [SAM02, p.37]:

- Internal personnel issues.
- The diversity of the applications in the legacy systems that must be linked.
- Standards and industry issues.
- A bewildering array of application integration options provided by different vendors to choose from.
- Data format problems.

5.1. TECHNOLOGY

In this chapter the challenges and opportunities in application integration are analysed in more detail from the point of view of technology.

As Forrester Research's EAI survey revealed, the technological issues are not the biggest challenges in application integration. However, technology is not irrelevant in application integration.

Standardization forms the biggest challenge in application integration. There are so many standards for application integration that there are no standards. An integration solution should be able to communicate with all applications, and therefore it should support "all" standards. This makes the development and maintenance of an integration product extremely difficult and most of all expensive for EAI and B2Bi product vendors. However, the situation

with the standardization is the same for everyone. One could argue that it is also a factor that generates business for application integration products because interoperability is otherwise extremely difficult.

Despite the standardization several opportunities are provided by application integration. Such opportunities are the reusability of components, smart error handling, efficient change management, ability to monitor business activities, performance and scalability.

Technological issues are challenging from the product point of view. Companies have different needs and technical environments that drive the decision-making. IBM is the oldest player in the field of application integration and it has extremely reliable and good quality products. IBM offers the widest package of tools to all EAI and B2Bi layers mentioned earlier in the study. However, IBM's offering builds up of multiple different software components that make the controlling of the whole integration solution a bit challenging. FRENDS Technology offers full package to EAI and B2Bi needs, except it does not have an own message queuing system such as MQ Series or MS Message Queue. FRENDS EAI Platform has a wide variety of supported interfaces and transformation tools including excellent process modelling tool as a foundation for the integration implementation. FRENDS EAI Platform has been one single package for more than 10 years. Though, FRENDS EAI Platform has not got so many adapters, one could argue that a whole package is more than the sum of its parts. Microsoft BizTalk is clearly a challenger with its "good enough" technology. Nevertheless, the R&D resources that Microsoft has can change this situation rapidly.

5.1.1. STANDARDIZATION

Computers on the Internet can communicate at a low level due to standards such as TCP/IP. The HyperText Transfer Protocol (HTTP) enables text and graphics to be displayed on virtually any PC, and e-mail can be exchanged between different systems if they support Simple Mail Transfer Protocol (SMTP). But beyond the network, browser and e-mail standards, there is only little agreement on protocols to let applications share information without human intervention.

There are so many standards for application integration that there are no standards. There are different approaches even inside a corporation when it comes to standardization. At the corporate level standards are chosen in the form of competitive differentiation. At the department level it is often considered that the respective department is different and special by its operations and application needs, and this drives the standard decisions. At the individual designer, architect, or programmer level the standard decisions are made on the basis: "I know best".

A few years ago XML and Java technologies such as Java to Platform, Enterprise Edition (J2EE) were considered revolutionary standards. It was presumed that these standards would solve all the problems in application integration. However, XML is just a language to express data in a structural and easy-to-read manner. XML makes it easier to manipulate and convert data formats compared to flat files but it doesn't help when there are hundreds of different e-business standards based on XML. XML is a technology standard not an e-business standard. J2EE technology provides a component-based approach to the development, assembly and deployment of enterprise applications and integration. The J2EE platform provides a distributed application model, an ability to reuse components, a unified security model, platform independency and flexible transaction control – pretty much everything needed in application integration. However, J2EE does not function properly on different platforms as claimed. Different application server vendors, such as BEA and IBM, aim to differentiate from each other by developing their own proprietary services to be used in J2EE applications. Consequently, these solutions are not portable between different platforms. Furthermore, J2EE designed applications lack performance because of the "additional" layer, the Java virtual machine.

At the moment standardization is more of a challenge. It is not very helpful that in application integration different application and enterprise systems use different standards. Information conversion, special adapters and connectors are needed to enable communication between source and target applications. Future standards such as Web services, RosettaNet and ebXML are possible candidates for simplifying the application integration challenges. It remains yet to see what shall happen as standards come and go.

5.1.2. REUSABILITY

Adapters, connectors and modules provide connectivity to a specific application and/or database, which enables the reusability in EAI. B2Bi standards create common business process models and documents to exchange information and do business between trading partners. This enables the reusability in B2Bi.

As mentioned earlier standardization is more of a challenge in application integration. It means that integration products must offer a large-scale support for different connectivity and information exchange standards. It is what application integration products really have implemented as a part of their offering. These products are built from components that enable different kinds of services in application integration, such as adapters to packaged applications, connectivity to common interfaces and libraries of business documents to enable reusability.

When individual coders connect applications with tailored scripts, reusability cannot be achieved. Reusability is a real opportunity in application integration, when EAI and/or B2Bi products are used.

5.1.3. Point-to-Point vs. Many-To-Many Model

Point-to-Point and Many-to-Many models were discussed in chapter two. When there are several systems to be interconnected in the company, the number of interfaces becomes crucial. In the point-to-point model the amount of interfaces grows exponentially. It creates a number of interfaces between applications. This kind of environment is expensive and difficult to maintain, develop and change. While having 10 different applications, the amount of interfaces can raise up to 45.

Many-to-many model minimises the amount of connections needed between different applications. It reminds the architecture of a telephone network where subscribers' terminals are like applications and the switchboard is like a many-to-many model integration product. It means that the amount of connections equals the amount of applications.

Even small and medium sized companies have several applications to support their business operations today. Also the need to network with partners and suppliers further increases the need for the exchange of information. Many-to-many model is better when companies have more than a few different application and/or trading partners to connect with.

5.1.4. Change Management

Several waves of technology over the past 30 years have each introduced new, largely incompatible techniques and protocols without totally replacing the previous wave of legacy systems. We are now confronting a heterogeneous environment of mainframe and client/server systems, procedural and object-oriented code. The next wave of service-based component frameworks, such as Web services, is just emerging. One could say that it will add another set of incompatible applications.

Integration products lay in the middle of heterogeneous system environment and it has to adapt to the changes and re-configurations of systems. While the integration product creates the logic of integration processes and the interfaces to different systems, changes can be easily re-parameterized to the integration product. Just create a new interface to the new system and adopt it as part of the other existing systems. Usually integration products have automatic documentation mechanisms for the integration processes, whereas documentation and information sharing are always challenges in the change management.

When integration is carried out separately in each system (point-to-point), changes require re-parameterizing in each system separately. This requires more work, more knowledge of all

the systems, more people involved that understand the different systems, and eventually it takes more time. This means more costs and downtime for the systems.

Change management is an opportunity that integration products take into account.

5.1.5. Error Handling

The integration product is connected to companies' business-critical systems. There is no doubt that errors occur either in different applications, communication methods and/or in the integration solution. Therefore error handling must be easily defined for each purpose.

Integration solution is in the center of everything. It has mechanisms to control applications, handle retries, do alternative processing and/or alert external systems and users via different media. It also gives the user a process oriented point of view to error handling because all this functionality can be attached to the created integration process. Integration products themselves are highly fault-tolerant because of their nature being in middle of everything and controlling of business-critical applications. Error handling can be controlled from one point and errors can be easily pointed out and delivered to the maintenance, to the help-desk and to the network management software. These are the main reasons why integration solution can be in a key role in error handling also. When error handling is taken care of independently in each application, maintenance is extremely difficult and expensive. It doesn't support a process point of view because business processes span over different applications, partners, network management systems service providers', etc.

Error handling is an opportunity provided by the integration product.

5.1.6. Monitoring

Integration solution transfers data between business-critical systems. This data and the integration processes that describe how the data is transferred and handled can be interesting for different groups. Integration solutions can access this data and many integration products provide Business Activity Monitoring (BAM) solutions that enable the real-time surveillance of these processes. Interesting monitoring data can be exported to other systems such as databases that can be used by customized web-tools or directly to users. BAM solution offers centralized sight to the processes that goes through several different applications. This is valuable by means of technology as well as business vice. It can reveal new useful business data as well as be an initiative to streamline business processes.

Monitoring provides opportunities to streamline business operations and it offers a real business solution "on top" of the integration product.

5.1.7. QUICK-FIX SOLUTIONS

Business and competitive pressures have driven many organizations to implement quick-fix solutions to meet aggressive time-to-market deadline. Once the quick-fix integration is implemented, no one seems to have the time or money to go back and do it the proper way. This leads to the building of several non-reusable interfaces, one at a time, over many years, between for example just two legacy systems. This scenario is very common both in the private and public sector.

Quick-fix solutions represent a challenge in the whole area of information technology. In application integration this is extremely common.

5.1.8. PERFORMANCE AND SCALABILITY

Integration solution usually builds up from several applications where the information exchange is high by means of volume. As stated earlier the integration solution is changing constantly while the requirements with existing applications change and new applications join the integration solution. These are the main reasons why performance and scalability are important issues in the integration domain. For example, performing online transactions, server clustering and balancing the load to different servers are challenges while creating integration solution.

Performance and scalability are issues of great importance in application integration. EAI and B2Bi products deliver these functionalities automatically, whereas, tailored integration solutions have to implement these functionalities separately, which generates costs and decreases reliability.

5.2. BUSINESS

This chapter analyses more detailed the challenges and opportunities in application integration from the perspective of business.

The biggest challenges in application integration are not technical issues. In integration project the idea is to connect several information systems. This means that several departments with different people are involved with the project. Everyone has their own opinion about their operations and systems. This leads to extremely difficult task to deliver cost effective integration solution that satisfies all needs. Among internal personnel issues and organizational issues, the amount of different products and methods in application integration field represent major challenges.

Nevertheless, application integration delivers several opportunities that enhance business operations. Automation of internal and external information exchange processes leads to cost

savings and it will streamline the business processes as well. This gives more time to concentrate in business issues rather than using time to manual IT work. Application integration leads also to more dynamic business relations. This is an opportunity for companies to bring into their new associates and automate cross-enterprise business processes with them - faster, easier and safer.

In the end, all decision makers are interested in what will be the Return Of Investment (ROI) and Total Cost of Ownership (TCO) of their investments. The calculation of ROI and TCO is extremely difficult for application integration because it affects to the whole IT infrastructure. The primary EAI and B2Bi advantage come in the area of interface development and the maintenance of them. These issues provide the efficient ROI and TCO that can be calculated case by case.

From the product standpoint the best TCO will be delivered with FRENDS EAI Platform. It delivers the cheapest license costs, fastest deployment and implementation. This is due to graphical rapid application development tools and efficient change management. IBM's business integration suite has the most expensive license costs. The deployment and implementation is also time consuming because it needs lots of hand-coding and different component installations. Stiff architecture leads also to problematic change management. Microsoft BizTalk server has the same difficulties that IBM's business integration suite but in smaller scale.

5.2.1. INTERNAL PERSONNEL AND ORGANIZATIONAL ISSUES

As stated in the Forrester Research's EAI survey, internal personnel issues are most challenging in integration projects. This is quite obvious. Integration is involved with almost all applications within company, with different departments, with trading partners and their systems and the personnel working close to these business systems. The following reasons issue the challenges:

- Several people are involved and should share their information about the interfaces, business processes, etc. Between all this they should co-operate with each other.
- A common practise is to organize business units around vertical products or markets. Each unit has a great deal of autonomy and is motivated to optimize operations around their particular segment with little regard to horizontal consistency. Different departments have their own proprietary solutions for their "special" applications that are always the "best" solutions. This also applies to the public sector. Different government agencies have highly customized, unique application solutions with few information exchange standards.
- Communication with trading partners and choosing the B2Bi standard for doing business.

- The resistance of change.

While several people are involved and the cooperation is difficult, it means that lots of money and resources are wasted, project will last long and the result is more like compromise than the most optimal integration solution.

Internal personnel issues are most challenging in integration projects.

5.2.2. AUTOMATION

Automation is a big source of operational efficiency and profitability. When a process is automated the cost usually diminishes. An automated process requires less human resources, produces constant quality, involves fewer errors and is predictable. The benefits of automation are obvious starting from the manufacturing process all the way to the whole value chain. Application integration is one of the key elements of automating the whole value chain. It enables the communication between various systems and partners, which is a foundation to the automation of business processes. For example, it enables the automation from supply chain and inventory management to bidding competition.

Automation is an opportunity provided by EAI and B2Bi products.

5.2.3. PRODUCT VARIETY

There are lots of different integration products, both for EAI and for B2Bi, in the market. As mentioned earlier in the introduction chapter, both of these terms are rather new in the area of application integration. This is one of the main reasons why there is such variety. The following reasons issue the challenges of choosing integration product:

- It is not easy to know what kind of services and functionalities the product has when it is stated to be EAI or B2Bi product, platform, server etc.
- All products offer large-scale support for "all known standards in the world" but are usually highly proprietary themselves. The implementation of integration solution can be done with proprietary tools and need extra training. Reuse of these solutions with other products is difficult or impossible. Replacement of integration product is expensive.
- Information system providers, especially ERP providers, offer integration products combined to their products. It makes the lock-in to customer stronger with these information system providers while providing also an integration product. This battles against the foundation of centralized integration, where companies can choose the best suitable applications for their needs, knowing that it can be interconnected to other systems via EAI or B2Bi product.
- License prises and the prises of implementation vary a lot.

At the moment the product variety is more of a challenge but there is also opportunities rising. Some of the products relay on open standards in implementation and the solutions can be reused with other products. Competition drives openness and accelerates the consolidation in the integration market that has already begun.

5.2.4. Mergers and Acquisitions

Mergers and acquisitions, the integration of companies, generate naturally application integration needs while all parties have their own systems for their business needs. It means that companies integrate their existing systems to work together and/or consolidate some key systems like ERP and CRM systems. Companies have their own visions, architectures and roadmaps how they develop the existing application infrastructure and how the integration is done. Even if an enterprise can define and enforce consistent application integration architecture, the first merger or acquisition that comes along can ruin the whole architecture. This occurs especially when the companies equal in size. It is difficult to choose, which architecture, possible integration products and business applications are chosen from each company.

However, mergers and acquisitions are always challenges in application integration but good architecture and efficient EAI and B2Bi products can remarkably help the inevitable integration project.

5.2.5. Return Of Investment and Total Cost of Ownership

Application integration projects within and between companies are typically big projects that require new software, dedicated hardware and special skill in planning and implementation. The projects are usually expensive and last for a long time. These projects are also very risky.

The Return Of Investment (ROI) helps to assess the feasibility of a project. It is based on a simple calculation in which the benefits are divided by the costs [EVA2002]:

$$ROI = \frac{S_1 + S_2 + S_3}{C} \quad (2)$$

Where

S_1 = IT cost savings from the integrated system having no overlapping functionalities.

S_2 = Cost savings from the improved process management.

S_3 = Additional revenues gained because of the integration.

C = Cost of the integration project.

However, ROI is extremely difficult to calculate because the value generated is not directly visible on the bottom line. One can only speculate what would happen if such an investment was made and, on the other hand, what would have happened without the investment. On the bottom line the operating systems and applications are still the same with or without an integration project.

The Total Cost of Ownership (TCO) means all the expenses that integration product and constantly ongoing project generate. As mentioned in the case ROI, it is extremely difficult to calculate exact figures. Here are some reasons why EAI and B2Bi solutions deliver better TCO than tailored integration solutions or cases without any integration solution:

- Savings in the amount of interfaces.
- Integration product has efficient tools that enable rapid integration and interface generation via its adapters and connectors.
- It generates a centralized architecture that enables companies to choose the best suitable application components for their operations, knowing that it can be interconnected to the other systems via EAI or B2Bi product.
- It can expand the use old applications that still serve the purpose that these were purchased for but may not have support for some new services such as Web services and other connectivity. EAI and B2Bi products can deliver these.
- Automation and reduction of human mistakes in business processes.

Integration investments share the similarity to companies' normal business investments. For example, company x can manage business without investing to an upgrade of a production line but if the investment is well planned and executed it can be an advantage for the company. This is quite often the issue in application integration. Companies have connected their applications and some of their partners' applications by using various different methods, such as scripts, point-to-point model, applications' own tools, etc. They have environments that have high maintenance costs, are highly unreliable and difficult to control.

The primary EAI and B2Bi advantage come in the area of interface development and the maintenance of them. These issues provide the efficient ROI and TCO that can be calculated case by case. The bottom line is that EAI and B2Bi are strategic investments that can reduce costs, not only over the long run, but also often on the very first project.

One of the most important opportunities of application integration projects is the delivered ROI and TCO. Most of all successful EAI and/or B2Bi project deliver remarkable cost savings among consistent integration architecture with efficient maintenance.

5.2.6. DYNAMIC BUSINESS RELATIONS

Application integration enables the dynamic business relationships and processes that respond flexibly and quickly to new business models and changing customer demands. It makes it faster, easier and safer for companies to bring into their new associates and automate cross-enterprise business processes with them. Only a tightly integrated and highly automated organization can have this level of flexibility and adaptability to newer partners and business initiatives [SAM02].

5.3. FUTURE

Companies are streamlining and automating their business processes by applying more information technology. It is happening already and it will continue in future. Competition drives to the increasing need of rapid information sharing within and between companies. The increased number of information systems applied within companies and the increased amount of information exchanged leads to more obvious need of the "man in middle". Otherwise the information sharing and management would create even bigger problem for companies than it does today.

Who will deliver this integration module in future? I believe that there will be two paths how the integration will be handled. Strong competition will continue in the "specialized" integration software market and the winning products will increase their market share. The most important information system vendors, such as SAP, represent the other path. This means that the information systems that lay in the middle, such as ERPs and CRM systems, will have integration capabilities. The whole EAI and B2Bi is created and managed via these systems that represent the most important business information systems. These signs already exist. Information system vendors co-operate with application integration vendors and offer EAI and B2Bi product integrated with their own information system. SAP has gone even further. They had a WebMethods EAI and B2Bi product integrated with SAP for couple of years. In 2003 SAP presented their own integration product.

The technological development in application integration will focus more to the process orientation and orchestration. It will climb closer to support the business processes. Process orientation comes even more relevant in EAI while the new B2Bi standards are already highly process oriented. It seems that information systems are becoming more open by providing several interfaces to communicate with them. Also the information systems are publishing easy to access services that enable more business vice communication between systems. At the moment Web Services standard represent this idea. In future these services will be even cleverer, and repositories and metadata presentations will make the integration closer to plug and play.

According to Yefim Natis, vice president and research director at Gartner: "Over time, lack of Service-Oriented Architecture (SOA) will become a competitive disadvantage for most enterprises. Mainstream enterprises should invest today in understanding SOA and building SOA design and development skills." Gartner predicts that by 2006, more than 60 percent of enterprises will consider SOA a guiding principle in designing their new mission-critical business applications and business processes.

The holy grail of integration technologies has been the ability to truly achieve an environment where each enterprise resource is exposed as a service that is accessible by any other service - a true SOA. The main impediment has been the pressure on customers to utilize a proprietary approach from Web services, development tools and integration technology vendors. By being forced to focus only on J2EE, .NET or some other isolated platform, rather than addressing the real issue of interoperability and vendor neutrality, customers have been unable to bridge all the architectures and technologies across their enterprise.

6. CONCLUSION

This study introduces the integration of applications within and between companies, EAI and B2Bi, and analyses the opportunities and challenges delivered by it.

Application integration has existed since the first applications. There is no doubt that companies are streamlining and automating their business processes by applying more information technology. It is happening already and it will continue in future. The increased number of information systems applied within companies and the increased amount of information exchanged leads to more obvious need of the "man in middle".

In EAI the obvious movement is towards process-oriented integration. The process integration layer will play major role in future. This kind of approach will decrease the gap between business processes and integration processes. Business processes can be easily and visually mapped to the whole information system architecture via integration processes. One of the main drivers that has lead EAI towards process orientation is B2Bi. B2Bi is all about communications between business partners in a highly process oriented way.

As stated in chapter 3.2 EAI and B2Bi are converging. Integration of companies' internal systems has existed much longer than integration between partners. This is one reason why it seems more likely that EAI market and vendors are invading the B2Bi sector to provide the "complete" integration package for companies. Another reason for this is the fact that it is extremely difficult to integrate processes and applications with trading partners if companies' internal integration, processes and applications, are not taken care of. This convergence issue was one reason why I wanted to include both integration aspects, EAI and B2Bi, into this study.

Main points of this study were the issues that represent the opportunities and challenges in application integration. This study and my own experience of four years in Finnish integration market clearly shows that EAI and B2Bi offer more opportunities than challenges.

EAI and B2Bi concepts enable the control of heterogeneous and all the time expanding IT infrastructure, the automation of business processes, networking with trading partners, etc. Application integration also clarifies the roles in IT infrastructure. Applications just provide interfaces for communication, and the information what they provide and need is delivered and controlled via integration system. There is analogy between application integration and telecommunications. The exchange takes care of the linking of subscribers with each other in telecommunications network. Subscriber just has one connection to the exchange that takes care of the linking, additional services, etc. However, the biggest difference in this comparison

is the standardization. Standardization is the main challenge in EAI and B2Bi. It is still to see if standardization succeeds in application integration and what happens to standards such as ebXML, RosettaNet and Web Services. Other challenge that is extremely difficult to remove is the internal personnel issues. Application integration is involved throughout the organization and partners, which brings a huge amount of parties involved with the issue.

The amount of information systems, information itself and communications between business partners is evidently increasing. Companies want to automate their business processes and enable the exchange on information between applications, personnel and partners. Application integration will play very important role in information technology in near future.

7. REFERENCES

[BAS03] Bass, Christy and Lee, Michael "Building a business case for EAI", www.accenture.com. (Christy Bass is an Accenture partner leading the organization's global EAI team. Michael Lee is an associate partner for Accenture's global EAI practice.)

[BPM03] Business Process Management Initiative's website, http://www.bpmi.org/bpql.esp, Accessed September 2003.

[COR03] Correia, Joanne "Change for business Integration", Business Integration Journal, September 2003. (Joanne Correia is a vice president at Gartner Dataquest, software team, focusing on the application integration, middleware, and portal software markets.)

[CXM03] Commerce XML website, http://www.cxml.org/ , accessed October 2003.

[EBA01] "ebXML Technical Architecture Specification, version 1.0.4", www.ebxml.org/specs/ebXML_TA_v1.0.4.pdf , 2001

[EBP01] "ebXML Business Process Specification Schema, Version 1.01", www.ebxml.org/specs/ebBPSS.pdf , 2001

[EBX01] Chappell, David A.; Chopra, Vivek; Dubray, Jean-Jacques; van der Eijk, Pim; Evans, Colleen; Harvey, Betty; McGrath, Tim; Nickull, Duane; Noordzij, Marcel; Peat, Bruce; Vegt, Jan "ebXML Foundations", Wrox Press, 2001.

[EBX03] electronic business XML website, www.ebxml.org , accessed August 2003.

[EVA02] Evans, N. D. "Business Innovation and Distributive Technology: Harnessing the Power of Breakthrough Technology for Competitive Advantage", Management Strategy, Financial Times Prentice Hall, www.ft-ph.com, 2002.

[FRE03] "FRENDS EAI Platform White Paper", www.frends.com , 2003.

[HAA03] Haapaniemi, Ari. "EAI and ebXML-based approach to Business-to-Business integration", Master's Thesis - Helsinki University of Technology, 2003.

[IBM03] Phone interview of Jari Päivinen, Sales Specialist IBM Software Group, Business Integration, 15.10.2003.

[LIN01] Linthicum, David S. "B2B Application Integration, e-Business-Enabled Your Enterprise", Addison-Wesley Information Technology Series, 2001.

[MIC03] Microsoft Corporation's website, www.microsoft.com , accessed September 2003.

[NAC03] "NACS Convenience Store Industry Technology Standards Project Overview", http://www.nacsonline.com/NACS/Resource/TechStandards/techstandards_overview.htm, accessed November 2003.

[NFR03] The National Retail Federation, http://www.nrf-arts.org/, accessed November 2003.

[OAS03] Organization for the Advancement of Structured Information Standards (OASIS), http://www.oasis-open.org/, accessed October 2003.

[MYS03] MySQL website, http://www.mysql.com, accessed August 2003.

[OMG03] Object Management Group – CORBA, www.omg.org, accessed September 2003.

[PAP03] Paper Commerce on the Internet (papiNet), http://www.papinet.org/, accessed November 2003.

[ROS03] "RosettaNet Implementation Framework: Core Specification, Version: V02.00.01", www.rosettanet.org/rnif, accessed October 2003.

[ROW03] RosettaNet Organization's website, www.rosettanet.org, accessed November 2003.

[SAI01] "Selecting an EAI solution – an introduction", Saint Consulting, 2001.

[SAM02] Samtani, Gunjan "B2B Integration: A Practical Guide To Collaborative E-Commerce", Imperial College Press, 2002.

[SHA01] Shapiro, J.F. "Modelling the Supply Chain", Duxbury, 2001.

[TIL03] Tilastokeskus, statistics of EDI users, http://www.stat.fi/tk/yr/tietoyhteiskunta/edi.html, accessed November 2003.

[UNE03] UN/EDIFACT website, http://www.unedifact.com/edifact/edif_m1.htm, accessed November 2003.

LIST OF TABLES AND FIGURES
