Structured communication for dynamic business

an architecture for flexible B2B communication

Fred van Blommestein

Published by: University of Groningen
Groningen, The Netherlands

Printed by: Ipskamp Drukkers B.V.
Enschede, The Netherlands

ISBN: 978-90-367-6396-7 (printed version)
978-90-367-6395-0 (electronic version)

RIJKSUNIVERSITEIT GRONINGEN

Structured communication for dynamic business

an architecture for flexible B2B communication

Proefschrift

ter verkrijging van het doctoraat in de
Economie en Bedrijfskunde
aan de Rijksuniversiteit Groningen
op gezag van de
Rector Magnificus, dr. E. Sterken,
in het openbaar te verdedigen op
donderdag 26 september 2013
om 16:15 uur

door

Frédéric Bernard Eugène van Blommestein

geboren op 28 december 1950
te 's-Gravenhage

Promotor:	Prof. dr. ir. J.C. Wortmann
Copromotor:	Dr. H. Balsters
Beoordelingscommissie:	Ao.univ.prof. Mag.rer.soc.oec. Dr.rer.soc.oec. C. Huemer Prof. ir. A.J.M. Beulens Prof. dr. G.B. Huitema

SAMENVATTING

Bedrijven en andere organisaties gebruiken computerapplicaties voor veel interne bedrijfsprocessen. Zodra echter de processen de grenzen van de organisatie overschrijden, wordt de bijbehorende informatie handmatig verwerkt. Alleen waar langdurige samenwerkingsverbanden bestaan met goed gedefinieerde en gestandaardiseerde processen, worden computersystemen van verschillende organisaties met elkaar verbonden. Het implementeren van zo'n verbinding kost veel geld, inspanning en tijd.

De meeste samenwerkingsverbanden tussen organisaties (koper-verkoper, klant-dienstverlener, toezichthouder-bedrijf) zijn veel te kortdurend en veranderlijk om de inspanning voor het koppelen van de systemen op applicatieniveau lonend te maken. In dit proefschrift wordt een architectuur gepresenteerd die het mogelijk maakt om applicatiesystemen aan elkaar te koppelen zonder die inspanning.

Applicatiesystemen die gekoppeld worden moeten in staat zijn om te onderhandelen over de uit te wisselen informatie. Ze moeten kunnen aangeven welke informatie nodig is en welke informatie ze beschikbaar hebben en wensen te delen. Welke informatie moet worden uitgewisseld is afhankelijk van de betreffende activiteit in het interorganisatorische bedrijfsproces. De choreografie van dat proces is daarom onderdeel van de onderhandeling.

Informatie is subjectief. Eigenlijk worden tussen organisaties geen feiten uitgewisseld, maar meningen. Waarnemingen kunnen verschillen. Informatie wordt daarom weergegeven in de vorm van 'speech acts'.

De architectuur die hier wordt voorgesteld kan direct worden geïmplementeerd in bedrijfsinformatiesystemen, zoals ERP systemen. Hij kan ook worden geïmplementeerd in 'middleware' die vervolgens gekoppeld wordt aan bestaande bedrijfssystemen. De architectuur kan ook worden aangeboden als service van een externe partij. Bestaande systemen die (nog) niet in staat zijn om alle aspecten van de architectuur te ondersteunen kunnen worden gecomplementeerd met een web-formulieroplossing. Zo'n oplossing wordt in dit proefschrift eveneens beschreven.

Dit proefschrift biedt een fundament voor gestructureerde communicatie tussen applicatiesystemen van organisaties die dynamische business relaties onderhouden.

SUMMARY

Enterprises and other organizations use computers for many of their business processes. Yet, whenever those business processes cross the organization's boundary, manual processing of information takes over. Only in cases where long lasting alliances between organizations deploy well defined, standard business processes, computer application systems are interconnected. Establishing such interconnection is costly and time consuming.

Most alliances between organizations (seller-buyer, client-service provider, regulator-civilian) are too incidental and too volatile to afford computer interconnection at the application level. In this thesis an architecture is presented for business application systems to interconnect on the fly.

In order to interconnect, application systems must be able to negotiate information requirements and information availability, under the condition of information sharing policies. Information requirements are dependent on specific steps in the business process, so business process flows should be negotiable as well.

Among trading partners information is subjective. Observations may deviate and opinions may be different. Business partners do not exchange facts, but opinions. Therefore the aspect of speech acts is introduced into the business communication and into the negotiation of the information flow.

The architecture that is defined in this thesis may be natively implemented in business information systems (such as Enterprise Resource Planning systems), in middleware, or it may be offered as a service to interconnect existing systems. Legacy systems that are not capable to establish external connections may be complemented with a web form based solution. Such a solution is described in this thesis as well.

This thesis proposes a fundament for organizations to use structured communication between their application systems in order to support dynamic business relationships.

TABLE OF CONTENTS

PART I INTRODUCTION AND ANALYSIS

PART II DESIGN

PART I INTRODUCTION AND ANALYSIS

Introduction

Supply chains span multiple enterprises and organizations. Tasks within organizations are increasingly being outsourced. Most business processes cross organizational boundaries. Yet, IT support for inter organizational processes is almost non-existent. Whenever information leaves an organization, it is 'degraded' by including it onto an unstructured document, an e-mail or in a telephone call. Interconnection between applications on the level of structured information has only been established in contexts where the organizations have long during relationships. This phenomenon will be analysed in chapter 1. The main cause is the rigidness of B2B systems, mainly at semantic levels, as opposed to the heterogeneity and dynamics of business.

The challenge this thesis addresses is to find mechanisms to allow businesses to interconnect their IT systems as business is developing, e.g. right after a business contract has been concluded, or even during contract negotiation. As will be described in chapter 2, currently the establishing of an EDI connection or other structured data exchange between computer systems needs high investments and time consuming involvement of IT personnel. Methods and standards that have been developed by scholars and industry organizations, and the advancement of information technology have not made the implementation of structured data communication as quick and easy as business requires.

The problem of computerized business communication may be approached from the bottom up (starting with technical interconnection, via session management and messaging to business process management) or vice versa, starting with business process management. In this thesis we have chosen the latter approach. Technology and application system configuration are rapidly changing, but the way business is conducted is basically stable for centuries. Therefore requirements posed by business processes on supporting IT systems can best be derived from the basic characteristics of business co-operation. The requirements for a mechanism to instantly establish connections between business information systems are derived in chapter 3.

In part II of this thesis such a mechanism is designed.

In chapter 4 a reference model is defined, where the concepts and architectural elements of an inter-organizational business communication system (in the sequel named as B2B or business to business system) fit in. Keeping the top-down approach, the starting point is the way human business people communicate, negotiate and co-ordinate their activities. They use interconnected computer systems as their communication channel. As the main concept in the reference model, a knowledge base is introduced that is shared by the business partners and that is populated in the course of a business relationship. Physically, the information in the knowledge base is stored in the respective business information systems of the business people. The information is kept in sync between the systems by means of data communication.

The knowledge base is filled with *utterances* of the business partners. An utterance is defined as the atomic part of a business document, message or information exchange, in which a property is assigned to a business object (a product, transaction, payment, etc.) or to a class of business objects. The knowledge base contains information on the actual business objects, but also on their structure or meta-information: the types or classes of business objects and their relationships. Both can be manipulated by the business partners by means of utterances. We even show in chapter 5 that utterances that propose a structural change (changing information on classes, the meta level) may be shaped the same way as utterances with operational information (changing information on instances).

In chapter 5 a simple but effective rule is introduced: each utterance must be based on a previously exchanged utterance. The rule allows the business partners (or their systems on their behalf) to build up their business case in the knowledge base, both on instance and on meta- (or type-)level. Utterances are attributed to allow for cardinalities, intentions and mechanisms to control future population of the knowledge base (and so refining the business relationship).

Seven types of utterances are defined: definitions, expansions, restrictions, instantiations, observations, perceptions and states. Some utterance types expand or alter the knowledge base structure, adding or altering information on classes, while other add information on instances. Information on instances must be based on information on the class of the instances that must have been exchanged previously. Information on a class must be based on information on a superclass.

The structure of the knowledge base and the attribution of utterances are represented in a table in chapter 6. The table structure, which is just an illustration of how the knowledge base may be represented, allows to illustrate the various attributes and to show a number of examples. At the end of chapter 6 a more formal meta-model of the knowledge base is given.

In chapter 7 we show that the knowledge base structure can also contain information defining the business process. By defining preconditions to future utterances, business partners may precisely agree on the sequence of business activities.

The knowledge base is to be implemented in local information systems, therefore in chapter 8 the meta-model and the mechanism is mapped to various modelling languages. The mechanism to populate the knowledge base needs to build upon a set of basic processes and ontologies. These are outlined in the subsequent chapter 9. In chapter 10 it is reasoned how an enterprise may specify e-business strategies departing from its business strategy. Such e-business strategy can be formalised in e-business profiles that lead to e-business agreements with business partners. Chapter 11 elaborates on the implementation of the proposed infrastructure and on the interfacing with business information systems.

Part III of the thesis illustrates the architecture that is proposed by showing and discussing in chapter 12 a number of e-business models and implementations in various business applications. In chapter 13 it is shown how it can be implemented in a browser-form based environment. This implementation may be used by enterprises that do not have an advanced business information system, or do not wish (now) to adapt their system. Proliferation of such open, web based environment lowers the threshold to use open, standardized electronic business enormously and removes the dead lock situation where most organizations wait for each other to create critical mass. Chapter 14 concludes the thesis with a discussion and with recommendations to standards organizations, technology providers, implementers and researchers.

1 Problem statement: B2B Systems

Summary

In this chapter, an assessment is made of the adoption of EDI and other B2B communication in relation to IT adoption by enterprises and organizations in general. It is concluded that adoption of B2B communication lags far behind IT adoption in other areas. Therefore, a huge productivity improvement potential lies ahead.

Unless mentioned otherwise, in this thesis the terms EDI and B2B refer to connections between computer systems of different organizations, at the level of structured data. EDI (Electronic Data Interchange) refers to the use of standardized messages using a standard syntax such as UN/EDIFACT [1]. B2B refers to the more general case where data from a sending application is directly read by a receiving application, i.e. not rekeyed, not printed-and-scanned. Data may however be reformatted, recoded, (manually) filtered or complemented. Data exchanged may also be (temporarily) stored and be distributed.

Conditions of B2B implementations between organisations appear to be different from IT implementation within organizations. Implementation of IT applications within organisations can be planned, decided and controlled. A hierarchical decision making structure exists. Though IT and Information Management have certain peculiar aspects, controlling the application of Information Technology is not fundamentally different from management of other technologies.

The implementation of communication systems between applications of different organizations however cannot hierarchically be planned, but must be negotiated. Organizations are autonomous. They have their own interests and policies. This context hinders B2B adoption. In a communication link, the IT implementations of both partners need to be aligned. Moreover, the technology is complex and rapidly developing. Many organizations have hundreds or even thousands of business partners. In a dynamic market the portfolio of business partners is changing all the time. Business processes, which determine IT configuration, innovate constantly.

The present B2B paradigm, which assumes that B2B communication is based on standards that have been agreed upon in international standardisation committees and/or is manually reconfigured bilaterally, is not capable to bring B2B adoption beyond a marginal market niche. An architecture is missing for middleware that automatically (re)configures connections with (new) trading partners.

1.1 Information Technology adoption

It is virtually inconceivable to us now, but twenty years ago most administrative activities in the business world and in government mainly involved rewriting or -typing information on forms. Data was copied from an order form onto a sales order, onto to a production order, a warehouse pick slip, a consignment note, an invoice, etc. Order lists

and production reports were updated by hand. This applied to all businesses, from large to small, from specialists to confectioners. Large companies had many employees who were dedicated to routine administration. For civil servants, it was even the most important part of their tasks.

These days, in most large companies and organizations data is only entered once in a computer system, which then triggers the work processes, produces the outgoing forms and can usually supply up-to-date management reports at the 'press of a button'. In smaller companies, too, the use of computer systems – mostly in the form of servers or PC's – has increased enormously.

In 2007 in the Netherlands all enterprises used an Internet connection, while in 1995 less than 10% was connected to the Internet. 87% had a broadband connection in 2007. 83% had their own website. 72% of the companies with 10 employees or more deployed an order processing system. 60% had linked their order processing system to a system for invoicing and financial administration. 24% even had integrated order processing and financial administration in Enterprise Resource Planning (ERP) software. [2]. Europe wide (in 2006) more than 70% of companies with 10 employees or more deployed accounting software and more than 16% used ERP [3].

The large-scale use of computer systems is the result of:

- The desire to streamline and accelerate activities.
- The opportunities offered by technological developments.

These developments have led to computer systems becoming ever smaller, more powerful and cheaper. Particularly the latter has led to the use of the computer also being accepted within smaller companies.

The automation of (repetitive) administrative activities has led to substantial cost decreases and productivity increases. Not only is less work involved with the administration, but fewer mistakes are made and the process is faster, much faster. A smoothly running administration also offers major advantages from the logistic perspective. Stocks can be reduced and the customer service improved.

A clear correlation between productivity and IT spending is however hard to find. Brynjolfsson has formulated this problem as the "productivity paradox", after the remark of Robert Solow: "You can see the computer age everywhere but in the productivity statistics" [4]. This paradox is due to the way productivity is measured in statistics. Productivity is the quotient of output and cost. Output of administrative processes, measured in dollars does not increase on the long term due to market competition. Productivity, measured in real output volume, has increased spectacularly. Ref

IT support of business processes did not only decrease cost of administration or increase labour productivity. IT considerably increased the speed and the quality of administrative services. Producing quarterly financial accounting figures of multinationals is nowadays a matter of days. Automated stock accounting and inventory control allow retail and other trading companies to hold an average inventory of days instead of weeks or months.

When ordering books or clothes via the Internet, consumers today expect overnight delivery.

Summarizing, in 20 years ICT has become an essential factor in the execution of business processes within most companies and governmental organizations. Without computers (and even without Internet) the economy would come to a halt.

1.2 Adoption of B2B Systems

What 20 years ago applied to the administration within companies, still applies to interaction between companies: there are considerable amounts of (repetitive) administrative activities, which are performed manually at both sides of a commercial relation. It therefore seems reasonable to expect that the use of computer systems in inter-organizational processes will also lead to, at least, the same improvements as within companies. The computer systems of companies must then be interconnected. And nowadays this is quite possible, with the global and simple means of communication known as the internet.

However, despite the advantages recognized or acknowledged by many parties, the expectation of substantial improvement in inter-organizational exchange has not been met until now, even though data exchange between organizations has been automated for years in some sectors. In these sectors, Electronic Data Interchange (EDI) is deployed, a technology that was developed before the internet emerged.

Statistics show that in 2009 in the Netherlands no more than 22% of the companies (with 10 employees or more) had connected their order processing systems to the systems of one or more customers or suppliers [5]. On a European scale in 2006, 12% of the companies in 10 selected industry sectors had interconnected their IT systems with suppliers, see figure 1.1 [3]. Surprisingly, these figures are stable; the population of companies that interconnect their IT systems seems not to increase. In 2002 in the Netherlands 18% of the companies had interconnected systems [6].

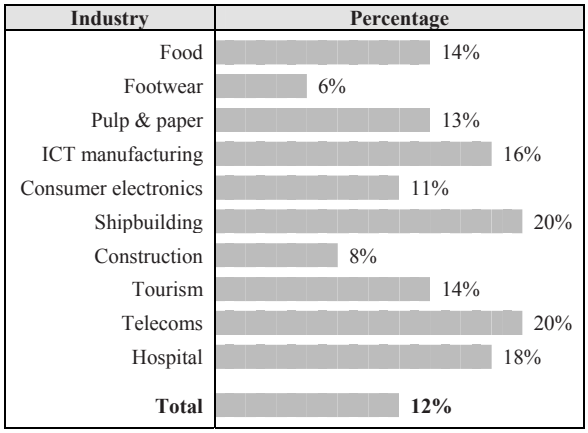


Figure 1.1 Companies whose ICT system is linked with those of suppliers [3]

In a survey made for the Dutch ministry of finance it was investigated with how many trading partners electronic invoices are being exchanged by Dutch companies that send or receive such invoices [7]. The outcome was that the average company using EDI at all for invoicing exchanged those electronic invoices with only 4 partners. The average number of trading partners of a middle-sized company may be several hundreds. A study conducted for the European Commission revealed that more than 50% of the firms that purchase online said that these purchases account for less than 5% of their total procurement. Only 13% of firms purchased online for more than 25% of their total procurement [8]. Note that these percentages are calculated over the volume in Euros, not in transactions.

The percentage of companies that linked their ICT system to the system of their customers or suppliers is therefore misleading. These companies linked their systems only to systems of a small group of selected customers or suppliers. With the majority of customers and suppliers, even among the 13% EDI adopters, communication is conducted in the traditional way by paper document or telephone.

Emmelhainz [9] assessed the situation in 1992 as follows: “EDI has been set to 'take off' for a number of years. Following its introduction, EDI was anticipated to become the norm in business first by 1985, then by 1987, then by 1990, then by 1992, and now by 1995 and beyond”. Today, 2012, it can be concluded that inter-organizational system interconnections are still exceptions rather than the rule.

1.3 Profitability of B2B Systems

With most internal business processes within organizations being supported by information technology, most manual administrative work is now dedicated to the external communication of organizations. A study in the Netherlands [10] revealed that in industry 13% of the total workforce is dedicated to administration and invoicing of sales and purchases. In wholesale trade 18% of the workforce is handling purchase and sales related documents. Interconnecting computer systems may save up to 90% of the manual document handling. The ECP.NL study [10] revealed that if automatic links between order processing systems would be normal rather than exceptional, in the Netherlands a saving potential of 7% GDP exists.

Garicano and Kaplan [11] classify transaction costs as coordination and motivation costs. They argue that B2B e-commerce affects both types of costs. Coordination costs are related to the determination of prices and to bring potential buyers and suppliers together to conduct a transaction. B2B e-commerce improves the efficiency of these business processes. Coordination costs are also reduced by reducing search costs in finding suppliers and by providing better information on the availability, characteristics and prices of products [12].

Motivation costs are related to the costs of informational incompleteness and imperfect commitment. B2B e-commerce reduces informational incompleteness costs through the standardization of product information. E-commerce contributes to reducing costs of imperfect commitments by standardizing processes and allowing for electronic tracing of transactions and products. According to Goldman Sachs analysts ([13], cited in [12]), in the United States the percentage total operational cost saving that may result from migrating from traditional procurement systems to B2B e-commerce is up to 39%. These cost savings are the result of the combined effect of reductions in transaction costs and greater competition among suppliers.

Industry	Cost savings
Aerospace machinery	11%
Chemicals	10%
Coal	2%
Communications/bandwidth	5% - 15%
Computing	11% - 20%
Electronic components	29% - 39%
Food ingredients	3% - 5%
Forest products	15% - 25%
Freight transport	15% - 20%
Healthcare	5%
Life sciences	12% - 19%
Machinery (metals)	22%
Media & advertising	10% - 15%
Maintenance repair and operating supplies	10%
Oil & gas	5% - 15%
Paper	6%
Steel	17%

Figure 1.2 Estimated B2B cost savings per industry [12]

The (macro-economic) saving potential of ubiquitous B2B interconnection of automated information system therefore is high. The enabling technology (automated order processing systems, Internet) exists. However, there seem to be obstacles in reaching a breakthrough in adoption of B2B computer interconnections.

1.4 EDI and B2B implementation practice

Before assessing the reason of non-adoption of B2B system interconnection, the state of the art of establishing such interconnection must be sketched. An organization, intending to interconnect its system with the system of a trading partner must manage the interconnection establishment as a project [14]. Standard order processing software usually only has basic tools to enable interconnection, and needs extensive scripting, parameter setting and sometimes programming before the connection is operational. In many cases additional tools and middleware need to be acquired and installed. Middleware is roughly defined here as the IT infrastructure and the software tools that allow business applications and business people to interconnect. Middleware includes the network and network management tools, interface mapping software, store and forward facilities, protocol and syntax converters and workflow supporting software. Part of the middleware is used to support B2B connections. It should support business people in establishing new connections with business partners and change existing ones.

One would suppose that with proper middleware, it should not be needed to involve IT specialists when connecting to a new trading partner. Negotiating between organizations about communication settings (incl. information structures and process flows) should be performed by business people or behind the screen by the middleware on their behalf.

Present B2B systems however have a different paradigm. Each connection to be established is seen as a separate IT development and implementation project.

Apart from the technical infrastructure, detailed mappings must be made between the transactions and data fields of the application and the messages and data-elements that are sent and received. Such mapping at the minimum needs studying and interpreting the meanings and formats of the messages, and often needs negotiation with the trading partner. Note that often the infrastructure needs to be implemented once, but the mappings must be made for each trading partner separately.

Many efforts to standardize electronic documents have resulted in useful specifications. These specifications cover the entire stack of functions needed for business communication: from bit oriented concrete syntaxes (ISO 646, ASN.1) via languages to structure the information present in documents (XML, UN/EDIFACT) to high level modelling methodologies for B2B relationships (UMM, ISO 14662) and libraries of standard data types, elements business information entities and messages (UN/CEFACT). These initiatives are elaborated on later in this thesis. Yet, the set of standards seems not to be complete or adoption of these standards seems to fail, given that each interconnection needs to be engineered individually.

At message definition level the standards leave too many options open. If some local community or business partnership decides to use the UN/CEFACT libraries, the probability that they will select a subset that is interoperable with another community is close to zero. There is much information defined in the library, which is not needed in a specific trade relation. Messages need to be customized for almost each trade relation. As an illustration: the UN/EDIFACT Purchase Order standard counts 1200 data elements [15]. Only 25 are used in practice, but in each environment 25 different ones. The question, which elements are to be used, must be negotiated in a bilateral manual process. Subsequently, interfaces must be developed for each customized message to the business information systems of the business partners. The threshold to initiate or change an EDI connection is therefore high.

Ten EDI projects were investigated in a study in the Dutch transportation industry in 1992 [16] (one industry sector in a limited geographical area) by means of interviews and study of the project documentation. Each of the projects appeared to have defined different processes, different document or message structures, different information definitions and different code sets. In 1992 the international EDI standardization of transport messages was well advanced. However, the interpretation of those standards differed between projects, in a way that inhibited interoperability among the projects.

Ten years later, in a study for the Dutch ministry of Transportation, interviews were held with EDI managers and EDI ‘opinion leaders’ (consultants and executives of sector organizations) about EDI proliferation in the transport industry [17]. The interviews revealed that by then still each EDI connection in the transport sector needed to be engineered and negotiated in a project, which cost between € 20 000 and € 50 000 for

each partner. A survey in the Dutch Transport sector showed that in 2002 only 4% of the transport orders were received by means of EDI messages [18].

Numerous studies have been conducted to investigate adoption factors of EDI, especially by small and medium sized enterprises, see [19] for an overview. Many of those studies identify factors that stimulate or inhibit adoption. Among the factors investigated are size, IT readiness, sector, market power and trading partner requirements. Only a few studies identify (lack of) standardization as an issue. Most studies (and statistics) measure the fact whether or not a company uses EDI at all, and do not assess with how many partners or for how many processes the company was interconnected.

All studies take the existing practice or paradigm, as sketched above, as a given. Individual EDI connections are to be developed as a project. So an EDI connection is an investment that is only profitable if the transaction volume (with the specific trading partner) is large enough and if the relation with that trading partner is lasting long enough. A lock-in into the relationship and consequently the loss of market power is inevitable. Moreover, an EDI system requires specific technical expertise, it needs a different workflow and organization to be in place and it may impose security risks.

Not all markets and sectors have a situation that is similar to the transportation industry as sketched above. In some (sub) sectors EDI standards are well defined and well maintained. An example is the Dutch food retail sector [20]. As all messages to be exchanged in that sector are well defined (including the semantics of codes), connecting a new trading partner is really a push of a button, if an EDI infrastructure is in place. The hind sight is that the process flow is fixed: no innovation is possible. The sector is also a relatively fixed community. To extend the community to other markets (say: fashion or toys) or to other processes (e.g. marketing) is much more difficult.

The current practice assumes that message and data definitions are in place when initiating a B2B connection. Most EDI standards are message oriented. Only message structures and (ambiguous) semantic definitions of elements have been standardized. Process choreographies (message sequencing) and additional constraints to the information to be exchanged must bilaterally be agreed. However, in many industries even standard message definitions do not exist yet for many processes.

Steel [21] identified 8 problem areas for the slow penetration of B2B system integration (summarized):

1. It takes far too long to navigate the standardization process.
2. There are multiple standards organizations involved.
3. When standards are updated, users stay with the version they are using, multiplying the number of 'standards' in use.
4. There should be a rule base to guide the use of conditional fields. This is not done, so each industry has a working committee to produce Implementation Guidelines which 'interprets' in its own way how to implement the 'standard'.
5. The result is a myriad of implementations of versions of EDI and XML messages.

6. The concept of standard EDI segment contents means that by the time the actual interchange is implemented, there are only one or two data elements actually used in each segment (or, for XML, in each complex type).
7. With the advent of the concept of public data bases, the whole UN/EDIFACT and X12 concept falls down completely, because it is oriented towards peer to peer messaging.
8. The current standardization is very cumbersome when applied in an open EDI environment where prior arrangements between trading partners do not occur.

EDI standards and systems have long been a very specific niche in the IT world. Most main stream suppliers of software products and services left the market of UN/EDIFACT translators and EDI Value Added Networks to a small group of specialized companies. With the advent of XML this is changing rapidly. XML is a language to structure the data contained in messages that are exchanged between applications, just like UN/EDIFACT. But while UN/EDIFACT was designed with a single application in mind (B2B communication) XML is multipurpose. XML moreover has a formal definition language for message structures (XML Schema) and integrates neatly with the other Internet standards. XML is supported by all main stream providers of tools, applications and services.

XML and other Internet standards may contribute to solve the infrastructure problem (an EDI infrastructure becomes easily available), but the semantic negotiation problem remains. Instead that the result of semantic standards and negotiations are documented in textual documents, they are documented in XML Schema. Like UN/EDIFACT, also XML does not offer a mechanism to automatically negotiate on semantics.

So although technology has advanced, Steels' assessment is still largely valid. Semantic standardization is still being performed the same way for XML messages as it was for EDI messages. Even worse: while most sectorial and regional EDI organizations at least conformed to the UN/EDIFACT or X12 libraries (the problem was interpretation rather than local development), many XML communities developed their semantic structures from scratch.

Many relationships between organizations are not stable enough to justify an EDI or XML B2B connection, neither in the business community nor in government, given the effort needed. Companies are constantly looking for new markets and more suitable suppliers and are permanently innovating their logistics and work processes. Government procedures also change regularly under the influence of new legislation and the pursuit of more efficient operational management. Neither EDI, nor XML can keep pace with these changes because the technology is too rigid. Dynamics and flexibility are too expensive for the business when use is made of EDI with the current approach.

In short, the main flaws of EDI standards are the lack of specificity and detail of the standards (too much local customization is needed) and the absence of a rapid (formal) mechanism to manage configuration and change. A third flaw that will surface when electronic relationships get more mature is the lack of process specification. All these

issues are addresses in subsequent chapters of this thesis. They are inspected more closely and assessed against research findings. An attempt is made to resolve some of them, at least at conceptual level.

Lehmann [22] sketches an infrastructure with which businesses may use their computers to bilaterally negotiate the way their EDI connection is to work. He presents an example dialog between two computer systems that results in the exchange of operational EDI messages. The computer systems, nor their users, have dealt with each other before. Lehmann's example is very illustrative and is therefore repeated here (figure 1.3).

A. Identify Parties
Bran: I am **BransonExplosives**. My **DUNS** number is **1234567.**
My encrypted signature key is **AAAAAAA**.
I respond to your **REQUEST FOR PROPOSALS #244** for **100,000 VOTARY CANDLES** and **10,000 CANDLE-HOLDERS**.
Chap: I am **USMC-Chaplaincorps-Procurement**.
My encrypted signature key is **BBBBBBBB**.
I accept that you are **BransonExplosives**.

B. Prior Accord
Chap: Have we dealt before?
Bran: Not directly. I sold training warheads to your parent system **USMC-LOGISTIC-BARSTOW** on **9/9/1996.**
Chap: **USMC-LOGISTIC-BARSTOW** just confirmed that to me, I accept it.
Bran: Three ontological protocols were agreed upon: generic **MONEY**, **TIME** and a customized **SAFETY** agreement.
Chap: I don't have **SAFETY**; I inherit **TIME** and **MONEY** from **USMC-LOGISTIC-BARSTOW**, which have not changed.
Bran: **TIME** and **MONEY** have not changed for me either. Let's agree to use our earlier **TIME** and **MONEY** ontologies for our transactions.
Chap: Agreed.

C. Common Grounding
Chap: I have access to the **CCAT** core ontologies:
SPACE, PART-WHOLE, ABSTRACT-ALGEBRA, EVENT-OBJECT-PROCESS, CAUSALITY, SITUATIONS, REPRESENTATION, MEASUREMENT-UNITS and **DEEP-CASE.**
I have **CCAT** non-core ontologies **GENERAL THESAURI, DIGITAL SYSTEMS, INFORMATION SYSTEMS, GEOMETRY, MATERIALS, HUMAN-ACTIVITY, QUASIRATIONAL-AGENT, ENTERPRISE MODELS, TRADE ACTIVITIES,** and **ADDRESSES.**
I have **CYC** ontologies **TYPICALAMERICAN, COMMERCESTUFF** and **GOVERNMENTWORK.**
For English words I have **ROGET-TAGGED.** I have ...
Bran: I too have access to those **CCAT** ontologies except for **GEOMETRY, MATERIALS.** Of the **CYC** ontologies, I have only **COMMERCESTUFF.**

D. Term Definitions

Chap: My special **PAYMENT-TERMS** for procurement are **NEXT-QUARTER.**

Bran: I have only **EDIFACT PAYMENT-TERMS** as listed in Element 4279; there is no EDIFACT data code there called **"NEXT-QUARTER".**

Chap: I will define it for you in terms of our shared **TIME** and **MEASUREMENT-UNITS** ontologies. See the formal ontological definition of **QUARTER (EDIFACT** Data Element Value **2151:3M**). Any **YEAR** has **4 NONOVERLAPPING OFFICIAL TIME-PERIODS** of **3 MONTHS** each, called **QUARTERS,** consisting of the **JANUARY** to **MARCH** period, the **APRIL** to **JUNE** period,...
If an **INVOICE** is **RECEIVED** by us on a **DATE,** one **MONTH** is **ADDED** to that **DATE**; the resulting **DATE** occurs **WITHIN** a **QUARTER** and we **PAY** the **INVOICE** in **US-MONEY** by **MAILED CHECK** to the **SELLER** on the **LAST DAY** of the **QUARTER NEXT AFTER** that **QUARTER.**

Bran: Understood and Agreed.

[It's also conceivable that the product itself could be mutually defined in addition to the usual EDI terms:]

Bran: Your Request for Proposals requires **100,000 "VOTARY CANDLES"** and **10,000 "CANDLE HOLDERS"**; I can supply **100,000 "ROMAN CANDLES"** and **10,000 "CANDLE HOLDERS"**. What exactly is a **"VOTARY CANDLE"**?

Chap: A **"VOTARY CANDLE"** is a **CYLINDRICAL OBJECT** with a **"WICK"** which is to be **LIGHTED** and **BURNED.** Its **PURPOSE** is **BURNING** from one **END** to the other, thereby **RADIATING LIGHT** to be seen by **PERSONS.**

Bran: My **"ROMAN CANDLE"** is a **CYLINDRICAL OBJECT** with a **"FUSE"** which is to be **LIGHTED** and **BURNED.** Its **PURPOSE** is **BURNING** from one **END** to the other, thereby **RADIATING LIGHT** to be **SEEN** by **PERSONS.** Does my **"FUSE"** mean your **"WICK"**?

Chap: A **"WICK"** is a **PIECE** of **STRING** which is **LIGHTED** and **BURNED** at one **END** so as to **LAST** a **TIME-PERIOD.**

Bran: So is a **"FUSE".** My **"ROMAN CANDLES"** may comply with your **REQUEST FOR PROPOSALS**. What is **"VOTARY"?**

Chap: **"VOTARY"** means something which is **BROUGHT** to an **ALTAR** by a **PERSON** for a religious **PURPOSE.** A typical **VOTARY CANDLE** is made of **BEESWAX.** and it **BURNS QUIETLY** for **12 HOURS** to **36 HOURS.**

Bran: My **ROMAN CANDLES** could be brought by a **PERSON** to an **ALTAR.** A typical **ROMAN CANDLE** is made of **GUNPOWDER** and it **BURNS LOUDLY** in from **0.25** of a **SECOND** to **3 MINUTES.** My **"CANDLE HOLDERS"** are **METAL** and fit within your specified **PDES/STEP SHAPE** and **MATERIALS** definition for **"CANDLE HOLDER".**

E. Assess Mappings

Bran: I understand **PAYMENT-TERM: NEXT-QUARTER** since our definitions are now logically equivalent. This is a perfect mapping. My **CANDLE-HOLDERS** fully comply with your **PDES/STEP** specification.

Chap: Yes, agreed.

Bran: OK. Our strict definitions of **"CANDLES"** are logically inequivalent but not inconsistent. The concepts could overlap.

Chap: I require more than possible overlap for this **REQUEST FOR PROPOSALS.** I require an **"EGG/YOLK"** mapping reliability level 13" or better for the **VOTARY-CANDLES** concept. *[in EGG/YOLK reliability theory for data mapping (see [23]), level 13 requires at least an overlap between the typical instances of both concepts]*
Your typical candle **BURNS LOUDLY** in from **0.25** of a **SECOND** to **10 MINUTES;** my typical candle **BURNS QUIETLY** for **12 HOURS** to **36 HOURS.** The intersection of these typical classes of candles is empty, so the reliability of the class-mapping is less than EGG/YOLK level 13. Apparently I must reject it.

F. Reconcile Differences
Chap: My requirement for agreement on **"CANDLES"** precludes my accepting that your **ROMAN-CANDLES** are **VOTARY-CANDLES** because **EGG/YOLK** reliability level 13 is not achieved.
Bran: Will you accept the risk that my **"CANDLES"** are incompatible with your **"CANDLES"** if I offer them at a deep discount?
Chap: No. I will not accept that **ROMAN-CANDLES** means **VOTARY-CANDLES** at any price.

G. Agree on Transactions
Chap: I require from you a **PROPOSAL** for **10,000 CANDLE-HOLDERS** only (no **CANDLES);** if it is satisfactory then I will send you a binding **EDIFACT**-style **"ORDERS"** purchase order; you will confirm with **EDIFACT** form **"ORDRSP".** Then you will ship me the **CANDLE-HOLDERS** in boxes bar-coded as **SHIPMENTS** with a **MANIFEST** message. Then you will send **EDIFACT "ADVANCE-SHIPPING-NOTICE"** and **"INVOICE"** to me for Payment. This will be done for each box of 100 **CANDLE-HOLDERS.** Payment terms will be **NEXT-QUARTER** as we agreed**.**
Bran: Yes, but I want t o ship in lots of 1000 instead of lots of 100.
Chap: Agreed.

H. Agree on Data Required
Chap: Does your proposed **INVOICE** contain the **PARTYs,** their **ADDRESSes,** the **INVOICE-DATE,** some **REPRESENTATION** of the **PRODUCT,** a **SHIPPER** and a **PRICE** in **US DOLLARS?**
Bran: All but the **SHIPPER.**
Chap: Add the **SHIPPER** to your invoice form and I will accept it.
Bran: Agreed. **SHIPPER** is defined in the **TRADE-ACTIVITIES** ontology and in my local database meta-data; I know my **SHIPPERS.**
Which data do you normally include in your **EDIFACT "ORDERS"** purchase order form?
Chap: All the **EDIFACT "ORDERS"** fields with non-empty values.
Bran: All I need is **0030** (date-time segment), **0120** (party segment), **960-STEP** (item description segment, but in **STEP** terms)**, 980** (quantity segment), and **1150** (price segment).
Chap: OK I'll skip all the rest **.**

I. Agree on Formats
Chap: I can send my purchase orders in the usual **EDIFACT** format.
Bran: Don't bother. Just use a flat file with tagged fields: XXX, YYY, and ZZZ, comma-delimited and in any order.
As a military agency you use a 24 hour clock. Convert it to 12 hour for these transactions, 4 numeric characters followed by 1 alpha character: **HHMM{A/P}.**
Chap: Agreed.

J. Agree on Channels
Chap: I use the **XYZ** VAN service, or encrypted **MIME** email.
Bran: Use encrypted **MIME** email.

K. Agree on Liability
Chap: We will be bound by **CYBER-UCC-500 SCHEDULE 6** for government buyers. You will be bound to perform thereunder, and in addition to indemnify us against any and all damage claims by third parties.
Bran: No. We will not indemnify you for "any and all damage claims" by third parties; we will only be liable for performance, breach, and actual damages due to negligence in manufacturing, as is already provided by **CYBER-UCC-500 SCHEDULE 6.**
Chap:*[Checking perhaps with a person or an expert system]* Agreed.
Bran: We have agreed on everything necessary for our negotiated series of binding transactions. Let us commence. My **PROPOSAL** will follow.

Figure 1.3 Example dialog between two systems

In this thesis we take this example as a challenge and attempt to describe how an infrastructure is to support such dialog.

1.5 Conclusion

The cause of the low penetration of B2B communication that has generally been accepted is the high set up and maintenance cost of a B2B connection. For each new trading partner to be connected, and sometimes even for each process step or message type, on each layer of the communication stack software must be developed, configured and/or installed. In most cases this is done after detailed negotiations with the trading partner. Each change has to go through the same laborious process.

B2B communication can only be as wide spread as internal IT support if a company needs to prepare its application interfaces and middleware only once and if the middleware is self-configuring itself to trading partners to be connected. Supposing that (legacy) applications stay rigid for still some time and that a typical application landscape consists of a number of applications (possibly based on various technologies), then the functionality to flexibilize B2B communication can only be performed by middleware.

The socio-economical problem that is addressed in this thesis is therefore:

How can enterprises and other organizations interconnect their business information systems on both semantic and technical level, as flexible and as dynamic as they do business, without the need for manual configuration and programming for each individual connection.

The main instrumental problem that is the cause of the socio-economic problems stated is therefore that:

An architecture is missing for middleware that supports dynamic business to establish B2B connections ("on-the-fly").

After such architecture has been designed, the components of a system that complies with the architecture should be specified.

The reference case that should be kept in mind while reading this thesis is the case of two organizations in different parts of the world, operating in different industries (say, a construction company in the Netherlands and a fasteners manufacturer in China), that have not done business before but wish to do business. Those companies should be able to interconnect their (ERP and order processing) computer systems when they negotiate commercial terms and the interconnection should be in place when the goods flow starts.

References

[1] ISO Internal Organization for Standardization: Information technology: Electronic data interchange for administration, commerce and transport (UN/EDIFACT) -- Application level syntax rules ISO 9735:2002

[2] Centraal Bureau voor de Statistiek, Voorburg, the Netherlands – De Digitale Economie 2008 (in Dutch)

[3] European Commision: eBusiness W@tch 2006

[4] Brynjolfsson: The Productivity Paradox of Information Technology: Review and Assessment, Communications of the ACM, December, 1993;

[5] Centraal Bureau voor de Statistiek, Voorburg, the Netherlands - De Digitale Economie 2011 (in Dutch)

[6] Centraal Bureau voor de Statistiek, Voorburg, the Netherlands - De Digitale Economie 2003 (in Dutch)

[7] EIM: Frequenties Factureren – Onderzoek voor het Ministerie van Financiën (in Dutch), 2002

[8] European Commision: e-business w@tch 2008

[9] Emmelhainz, M.A., EDI. A Total Management Guide, Second ed. Van Nostrand Reinhold, New York. 1993

[10] ECP.NL, Leidschendam, the Netherlands: Interoperabiliteit in Nederland (in Dutch), 2007

[11] Garicano, Kaplan: The Effects of Business-to-Business E-Commerce on Transaction Costs, Journal of Industrial Economics, Blackwell Publishing, vol. 49(4), 2001

[12] UNCTAD: E-Commerce and development report 2001

[13] Goldman Sachs Investment Research, 1999

[14] Leyland: Electronic Data Interchange, a management view, Prentice Hall, New York, 1993

[15] United Nations Centre for Trade Facilitation and Electronic Business: Trade Data Interchange Directory, 2009

[16] Berenschot: EDI Berichtenstandaards in de Nederlandse transportsector (in Dutch), Ediforum 1992

[17] Berenschot: ebXML in het goederenvervoer (in Dutch), Ministerie van Verkeer en Waterstaat, 2003

[18] Heliview: Automatiseringsgraad in het Nederlandse wegvervoer (in Dutch), TLN, 2002

[19] Arendsen: Geen bericht, goed bericht (in Dutch), PhD thesis, Amsterdam University Press, 2008

[20] http://www.gs1.com/ecom/eancom, consulted 2010-04-04

[21] Steel: Another Approach to Standardising EDI, EM - Electronic Markets No. 12, September 1994

[22] Lehmann: Machine-Negotiated, Ontology-Based EDI. In: Proceedings of CIKM-94 Workshop on Electronic Commerce, Springer, 1995

[23] Lehmann: The egg/yolk reliability hierarchy. In: Proceedings of CIKM-94 Workshop on Electronic Commerce, Springer, 1995

2 State of the art in B2B theories and standards

Summary

The various aspects of B2B communication can be positioned in a layered model, in which the more technical aspects are at the bottom and the more managerial and semantical aspects at the top. Each of the layers have been modelled in various theories and have been standardized in a multitude of standards. In fact in too many standards. The number of options at the lower layers are still limited. One can design handshake mechanisms that allows middleware to automatically negotiate what options to use in some bilateral communication link. Such handshake mechanisms have been proposed in literature and have even been standardized for limited scopes.

For the top layers, that cover semantics and process flow, the degrees of freedom are almost infinite and no mechanism exists to automatically reconcile different requirements and views of business partners. The paradigm for establishing B2B connections at these levels is to define standards for business sectors. However, in many cases the standards need to be adapted during implementation, which makes the implementation costly.

The survey of the state of the art underpins the need for a mechanism for computer systems to expose their technical, syntactical and semantic capabilities to each other and "shake hands" on a communication protocol that complies with those capabilities.

2.1 Introduction

Within Information Technology a number of theories, standards and products have been developed to solve the B2B interconnection problem. These initiatives have led to established practices. Some theories and standards however were not adopted by industry, or were only developed recently. In this chapter relevant initiatives are inspected. We focus on scientific and standardization initiatives that are directly targeted towards B2B systems.

In sections 2.2 through 2.8 relevant theories and standards are assessed against the current implementation practice and against the problem that is described in the previous chapter. In section 2.9 some conclusions are drawn. The main conclusion is that no theory (and certainly no implementation) exists yet that supports the automatic establishment of a B2B connection. Especially at the semantic matchmaking level B2B connections need to be designed and tailored. When trade relationships are stable and business processes stay the same over a longer period of time, businesses may use the standard design for the industry they participate in. But for many businesses the latter option is unacceptable. These businesses need to keep the opportunity to innovate, in products, processes and partners.

In the sequel of this thesis a method is designed to allow B2B connections to be established in an automatic fashion, especially at the semantic level. The design makes use of results in scientific disciplines that did not have a specific focus on B2B communication, such as ontology engineering and conversation analysis.

Most theories and standards regard the B2B interconnection problem as a technical interfacing problem. B2B interoperability from this perspective is studied with the layers of abstraction in a communication stack in mind: from technical interoperability via semantic interoperability to organizational interoperability and legal context. This communication stack is pictured in figure 2.1 (from [1]).

It is shown in section 2.7 that B2B communication can also be described as a (psychological or economical) human conversation phenomenon. Then methods become available to manipulate semantics and process choreography. Semantics and choreography can be formalised. Formal manipulation may be automated. This is elaborated upon in later chapters of this thesis.

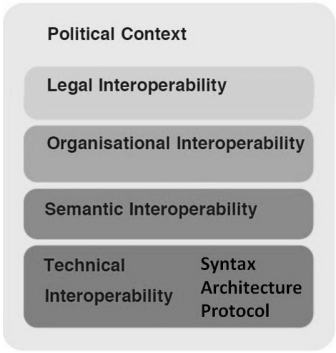


Figure 2.1 B2B Interoperability stack (adapted from [1])

2.2 Technical interoperability: Protocol

An inter-organizational business system can be designed top-down, starting with enterprise strategy, through business models and client relations onto operational and data processing capabilities. One can however also start with the computer infrastructure that makes use of network facilities to interconnect business applications. When an infrastructure is in place, it can then be used to (pragmatically or systematically) exchange information to support business processes. With some luck, both exercises meet in the middle.

This section focuses on the technical interface between computer systems of different organizations. Technological development with regard to the two lower layers in figure 2.1 is surveyed and assessed against the problem as stated in chapter 1. Conclusion is that although standardization of Internet communication protocols has converged considerably compared to the pre-internet era, too many technical options exist. The

number of those options cannot be justified by the number of business requirements. What is also missing is a handshake protocol, for machines to agree on a configuration.

The technological environment is an important context of the B2B connections as defined at higher stack layers. The higher layers cannot be described, understood and assessed without that context. The lower layer flaws (too many options and the absence of a handshaking mechanism) that hinder the establishment of an open B2B environment are however not further addressed or resolved in the sequel of this thesis. This thesis focuses on the higher layers, especially the semantic and process flow layers.

The IT industry has struggled with two phenomena that have hindered the bottom-up design of application interconnection. The first one was that on each layer of the communication stack there always have been multiple competing technologies. The second one was that technology advanced so rapid, that many standards were outdated before they could be implemented. With the explosion of Internet adoption fortunately now some convergence can be spotted.

An infrastructure to support communication between computers can be described as a stack of layered services. Each service in a layer makes use of the services in lower layers and is called by services at higher layers. The advantage of layering is that services may be upgraded or replaced without affecting services on higher or lower layers, if the interfaces between the layers are not changed or if these interfaces change in an upward compatible way. Such modularity of communication services not only let lower services fulfil a multitude of requirements on higher levels but also enables dynamic upgrading of the infrastructure as technology advances.

ISO has published in 1984 the Open Systems Interconnection reference model as ISO/IEC7498 [2]. See figure 2.2. In the reference model communication links between applications are divided over seven layers. Although most subsequent ISO and ITU standards to fill the layers have now been overtaken by IETF and IEEE standards that are used for Internet connections (and though those standards do not always follow the OSI layering to the letter), the OSI reference model still proves to be very useful to separate the functions within a connection.

OSI Model			
	Data unit	**Layer**	**Function**
Host layers	Data	7. Application	Network process to application
		6. Presentation	Data representation and encryption
		5. Session	Interhost communication
	Segment/Datagram	4. Transport	End-to-end connections and reliability
Media layers	Packet	3. Network	Path determination and logical addressing
	Frame	2. Data Link	Physical addressing (MAC & LLC)
	Bit	1. Physical	Media, signal and binary transmission

Figure 2.2 Open Systems Interconnection reference model

With Internet connectivity in place, the lower 4 OSI layers now are a commodity for almost all businesses. At the top of this lower part of the stack, protocols like HTTP, FTP and SMTP are situated. For electronic business one of these protocols needs to be chosen and within the protocol still many options exist that mainly have to do with security aspects and quality-of-service. All lower layers may stay invisible, not only for the business user but also for his information and communication manager.

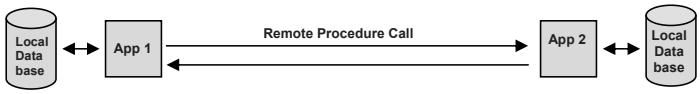
Unfortunately the upper three OSI layers have got very crowded. Numerous products and standards have been developed to support application interconnectivity. Applications may exchange data synchronously or asynchronously, secure or insecure (with security to be implemented at a layer of choice), reliable or not reliable, binary or textual (where sometimes a text file is represented in binary format by the application, re-represented in ASCII by the middleware and encoded in some binary format again by the lower layers). It is no wonder that this complexity has prevented a breakthrough of electronic business, and then we did not even touch the thorny issue of semantic alignment.

The key to avoid the necessity to agree bilaterally on the technical protocol to use when establishing a B2B connection is to standardize technical profiles based on real business requirements, preferably at ISO or W3C level, and to define an automatic mechanism by which two systems may agree which options are selected. In section 2.4 the middleware architecture is sketched that may support such handshaking.

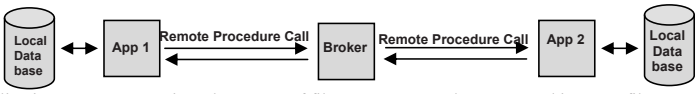
2.3 Technical interoperability: architecture

In this section the technical architecture of B2B systems and infrastructures is inspected. The main purpose of this section is to define the technological context of the higher layers of B2B communication. There exist multiple technical mechanisms to share information among businesses. Some examples are here illustrated by means of simple diagrams: squares represent applications, computer programs or processes, cans represent data stores or data bases.

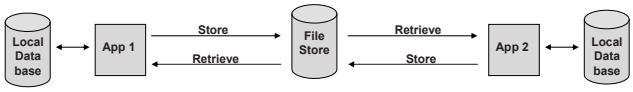
1. Computer programs, running under responsibility of different organizations, may communicate with each other by means of Remote Procedure Calls:



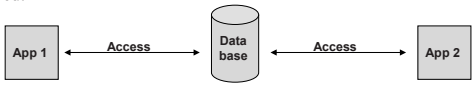
2. Like 1. but they may use some intermediate process or broker to avoid too tight dependencies:



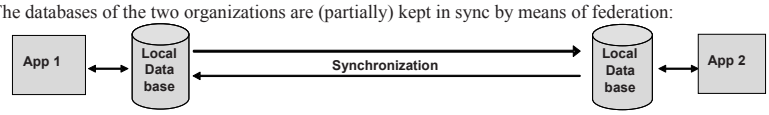
3. Applications may communicate by means of files or messages, that are stored in some file system, reachable by both:



4. Both organizations may have access to some central application of database, in which the common information is stored:



5. The databases of the two organizations are (partially) kept in sync by means of federation:



6. Applications in both organizations may have access to each other's database:

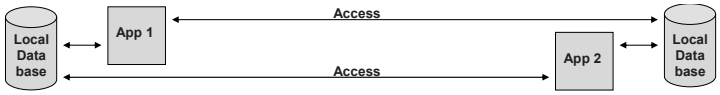


Figure 2.3 Interoperability architectures

The communication mechanisms can basically be grouped in one of two categories: Peer-to-peer or service oriented, as illustrated in figure 2.4. In a peer-to-peer system both partners listen (regularly or permanently) to the network until the other partner has sent or posted some message. The message contains information that alters the partners' relation in some way. In a service oriented system one of the partners listens and the other may request or send information.

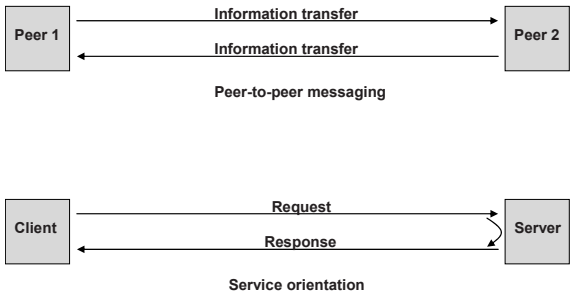


Figure 2.4 Peer-to-peer versus Service orientation

Peer-peer information transfer may be implemented by means of a service oriented infrastructure. If both partners have implemented a service to receive messages, that service may be called by the other partner. If only one partner has implemented a service, that service may be called by the other partner to leave a message or to pick up a message.

All mechanisms require that applications and/or database systems have a common way to exchange data, using the protocols mentioned in section 2.2. In most cases, the protocols that may be used are specific for an architecture.

Although this thesis is not concentrating on these technical protocols and mechanisms, it is important to realize that protocols and mechanisms on higher (e.g. semantic) levels should interwork seamlessly with these lower level protocols. Moreover, possible semantic handshaking protocols need to use these lower level protocols as well in order to be operational. Identification of the most frequently used standards and products on the lower level is therefore needed.

Bussler [3] has designed a high level reference architecture for e-business systems. He uses concepts like ‘business events’ and ‘ports’. His ‘business event’ maps more or less on the UMM BTV concept of ‘Transaction’ (see section 2.7). Bussler's approach should be characterised as bottom-up.

Bussler makes a distinction between design time and run time, and between the Public process (that is shared with the trading partner), the Private process (over several applications) and the Application process of a specific application.

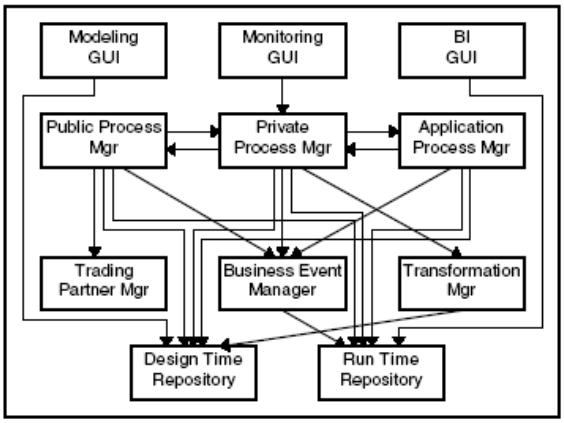


Figure 2.5 e-business system architecture

Bussler lists the following architectural components (figure 2.5):

- **Business event manager**: manages the instances of business events and their states
- **Private process manager**: executes private processes (e.g. workflows)
- **Public process manager**: executes public processes and the related business event instances, it has also to deal with security, reliability and transport
- **Application process manager**: controls the application processes and the related business event instances
- **Trading partner manager**: stores information about trading partners, their unique identifications, protocols as well as security keys
- **Transformation manager**: transforms business events (messages) into the various representations (UN/EDIFACT, XML, etc.)
- **Design time repository**: stores all metadata and master data that are required to execute B2B integration
- **Run time repository**. Stores run time data.

A similar configuration has been developed in the OpenXchange project [4].

From the perspective of an application, B2B communication means that the application must be prepared to receive information from trading partners (under certain conditions) and that it must send information under other conditions. In 1987 Klaver and Verberne [5] designed and prototyped a system that extracts data from a relational database whenever the data is in some predefined state. That data is then transferred to a trading partner who includes it in his database. That system illustrates the functions of a business communication system. In such a system one needs to define under which triggering conditions which data should be extracted and transferred. At the other end the pre- and post-conditions should be specified in order to check the validity of the data received (see figure 2.6).

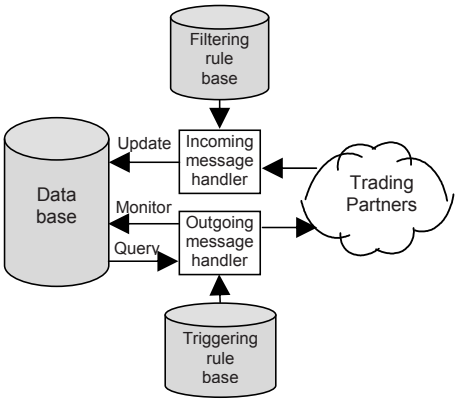


Figure 2.6 Application as reactive system

Security is an important issue in B2B communication. Security includes integrity, confidentiality and non-repudiation. Basically most security features can be provided by a combination of [6]:

- a public key infrastructure (with trusted certificates)
- encryption of messages
- electronic signatures
- protocols with acknowledgements

In addition the right Quality of Service should be installed to ensure reliability.

The problem with the security options is that there are so many to choose from. Most of the mechanisms mentioned above can be installed at multiple layers in the communication stack. It would be overdone to install them on each level, so a choice must be made, for each of the features. That complicates the matchmaking between systems.

We may conclude that for each business situation a suitable technical architecture with supporting technical protocols is available. The problem is that the number of technical options exceeds the set of business requirements in terms of responsiveness, reliability, and security. Yet, it should be possible to implement a handshaking protocol that business systems may use to agree on both an architecture and a communication protocol, especially if industry standards limit the choices somewhat. Technical handshaking is outside the scope of this thesis. However, as it is an important part of an open B2B system, it is briefly addressed in the next section.

2.4 Handshaking

The Bussler architecture was further elaborated in the EU project OpenXchange [4], with the aim to enable support for handshake protocols, also at higher stack levels. Based on

the OpenXchange findings, semantic and process level handshake protocols will be designed in subsequent chapters of this thesis.

The OpenXchange architecture (figure 2.7) includes design time, profiling time and agreement time along with transaction time processes.

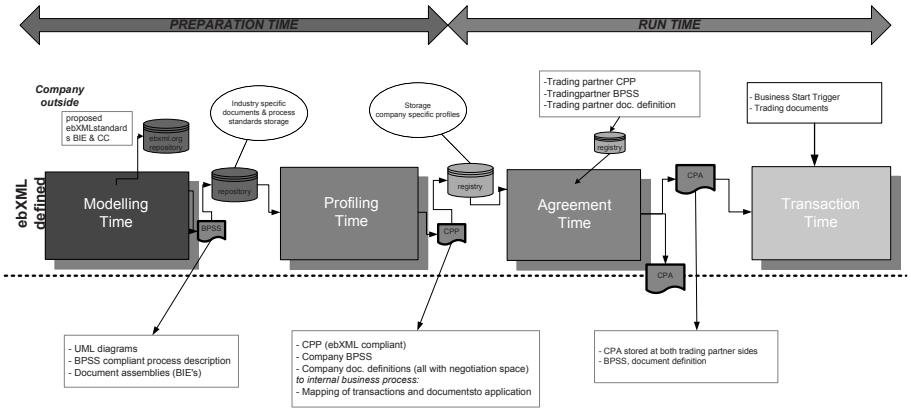


Figure 2.7 OpenXchange e-business phases [4]

At modelling time, sets of models and model modules are created and published. Models include process specifications and definitions of the information to be exchanged during process steps. Models are usually created by industry sector consortia and standardisation committees. The models (and model parts) they create are not rigid and carved in stone, but allow many degrees of freedom.

At profile time, individual companies create a profile of or subset from the standardized processes. The individual profiles are based on the company policy and its (technical and semantic) capabilities. Profiles are stored in a machine readable (XML-)format.

At agreement time, the company (automatically) negotiates with some trading partner how the profiles of the two companies can be combined and matched. As the profiles were based on the company's capabilities, the resulting Agreement can be directly implemented and is supported by both local systems. The Agreement is also stored in a machine readable format that can be used by the middleware of the companies to control the processes and information exchanges.

At runtime, the middleware takes care of the transfer of information between the information send/received to/from the trading partner and the local applications.

Essential parts of the OpenXchange architecture are the machine readable profiles and agreements. The structures of these artefacts have been defined in the ebXML set of standards as Collaboration Partner Profile (CPP) and Collaboration Partner Agreement (CPA) [7]. The ebXML set also contains a Registry Specification [8]. In an ebXML

Registry the standardized processes and information for an industry sector can be stored and published. Using this architecture a company could implement one or more mechanisms and automatically negotiate with its peers which one to use for a specific process.

The OpenXchange architecture was designed to introduce the dynamics and flexibility in B2B communication, as meant in chapter 1. During the OpenXchange project however, no technology was developed to support the architecture. In subsequent chapters of this thesis the OpenXchange principles for matchmaking are used to design a handshaking mechanism on semantic level.

A profile matching standard for B2B communication, spanning all standardizes protocols that are used for B2B communication and all features that fulfil user requirements is still a white spot on the standards map. Development of such profiling standard is beyond the scope of this thesis. This thesis concentrates on the semantic levels of communication. The technical handshaking should complement a semantic level handshaking protocol.

2.5 Technical interoperability: syntax

Traditionally, EDI standardization focused on the syntax of the information representation, not on the technical protocols and weakly on the semantics. A syntax is a mechanism to represent structures of elementary pieces of data or data elements in a stream of characters. The characters on their turn may be encoded in a binary format. The resulting bits are transmitted using the lower level protocols as described in section 2.2.

EDI syntaxes were standardized to enable computer systems to parse and interpret the bit stream, and pass the data elements on to an application. The semantics (layer 4 in figure 2.1) were supposed to be described in a human readable rather than in a computer readable way, therefore standardizing the way semantics were to be defined was not seen as important.

In the 1980s, when larger enterprises started to use computers for order processing and logistic control, in a number of industries the requirement came up to interconnect computers to exchange orders, invoices and consignment notes. In the USA a standardization committee (X12) was accredited by ANSI to develop and maintain both a mechanism to structure data in messages or “transaction sets” that could be exchanged between business applications and a library with standardized transaction sets. The mechanism and the library [9] have since been implemented in thousands of company interfaces that are still operational today.

In the UK, mainly in the retail sector, a slightly different standard for Electronic Data Interchange (or EDI, as the new technology was named) had been developed, named GTDI (Generic Trade Data Interchange). The American and European standards were combined by a joint US/European EDI working group (JEDI) in one EDI standard, UN/EDIFACT, that was adopted by ISO in 1988 [10]. At the same time a library was set

up with standardized messages, segments, elements and code lists. Later, the JEDI committee was reorganized to become a true standardization body: UN/CEFACT.

UN/EDIFACT messages are defined on a specific syntax, defined in ISO 9735: the UN/EDIFACT application level syntax rules.

The syntax is not very precise. It was developed out of practice and was not designed for mathematical beauty. If messages are not designed carefully they may become ambiguous, even if they conform to the syntax rules. Therefore the syntax rules were complemented by UN/CEFACT (the maintenance organization of UN/EDIFACT) with Message Design Guidelines [11] to avoid such ambiguity.

A syntax that is unambiguous is ASN.1 or ISO 8824-1 [12]. ASN.1 is mainly used in the telecom industry for the specification of (low level) network protocols. The encoding rules produce files that can be parsed and interpreted in a very efficient way.

Another approach to define electronic (business) documents is to start from paper or human readable document structures. Initiatives that attempt to standardize business documents include SGML, ODA/ODIF, Tex and PDF. Electronic documents are structures that can be represented in binary streams that may represent the document content, document representation and sometimes document navigation. The term "Document" stands here for bundles of information that are (also) human readable.

SGML [13], registered as ISO 8879 in 1986, is a generic standard for the structuring of documents. The original aim of SGML was to add mark-up annotations to the document content in order to type-set it. SGML in fact is a meta-language that describes how document features may be tagged and structured. Specific document type definitions need to be laid down in separate specifications.

XML, the eXtensible Markup Language is based on SGML, just like HTML, the language with which websites are developed. In fact well-formed HTML, named XHTML, is an XML instantiation (see figure 2.8). In contrast to HTML, which only defines *how* information is displayed on a screen, in XML the structure and the meaning (the '*what*') of this information may be defined as well. This makes XML a candidate for replacement of existing EDI syntaxes such as UN/EDIFACT. Unlike UN/EDIFACT XML is supported by major vendors of business administration systems and middleware. UN/EDIFACT functionality is only offered by niche players.

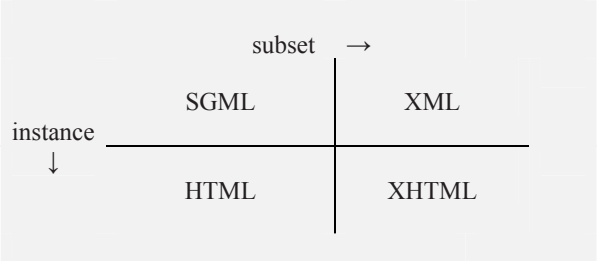


Figure 2.8 Mark-up languages

The structure of an XML document type (and the meaning of the tags) can be defined in a computer readable document schema. The structure of a schema has been standardized in the W3C XML Schema specification [14].

The XML Schema language is a very rich and flexible language. The same XML structure may be described in XML Schema in various ways [15]. Different B2B communities have selected different schema mechanisms. In some cases the selection is motivated by sector specific requirements. Examples are business reporting (where hierarchical data structures need to be represented) and statistics (where data sets have multiple dimensions). In other cases the selection was arbitrary.

For B2B messaging, UN/CEFACT has developed a specification for definition of message types in XML Schema: the UN/CEFACT Naming and Design Rules (NDR [16]). The NDR defines an XML Schema style, lists the XML Schema features to be used, defines conventions on naming and namespaces and specifies what annotations to include in the schemas. Annotations are in fact extensions to XML Schema.

Presently, most B2B communities converge to using XML as the syntax for B2B communication. XML, as a syntax, may be regarded as the syntactical vehicle for future B2B developments. In this thesis, in which we develop methods to align on semantics, we rely on XML to be used as the machine readable syntax for both meta-data and instance data.

2.6 Semantics

The definition of B2B semantics is in many industries synonym to the definition of messages. Messages are enveloped packages of information. In many cases an electronic message is equivalent to a (paper) business document. Usually each message is given a name, its function is defined and its structure, which consists of groups of data elements, some mandatory (they must be present in each instance), some optional (they may be present in an instance) and some conditional (they must be present under certain conditions). Groups and elements may repeat. Data element groups may have some hierarchy (groups inside groups).

Most B2B and EDI messages have been defined within an industry sector in some geographical region, e.g. the automotive industry in the USA or the Fast Moving Consumer Goods sector in Europe. Internationally and cross-industry message development is coordinated by UN/CEFACT.

The paradigm of Electronic Data Interchange is that electronic (instead of paper) documents are “printed” into the syntax (e.g., UN/EDIFACT) format by the sending application and “read” by the receiving one, that immediately includes the information into its database. Message formats are standardized, so application interfaces can (and must) be written upfront.

This paradigm assumes that messages can be standardized, that the number of different message structures stays small and that scenarios stay simple. In practice however these assumptions proved not to be true. The number of UN/CEFACT maintained message types is now well over 200. In practice, though, that number is much higher, as each sector based initiative and even each implementation interprets the standard messages in a different way. These interpretations sometimes are documented as “Message Implementation Guidelines”, sometimes as Interchange Agreements and sometimes they have not at all been documented but were directly programmed into the application interfaces.

Various industry sectors have defined their own libraries of messages and elements. The Health Care sector uses a library called HL7 (nowadays based on a data model), the European gas distribution industry Edig@s, etc. In some cases these industry specific libraries are based on the UN/CEFACT libraries, but in many cases they were developed and are being maintained independently.

Steel [17] initiated the Business Semantic Repository (BSR). In the BSR, the semantics of terms and data elements from various libraries can be documented. The BSR could then be used to make mappings to and translations of those data libraries. The work on the BSR has resulted in UDEF [18], a coding system of objects and object types, which is heavily supported by NATO and the US Department of Defense, but not by other industries.

Numerous industry and professional sectors have published specific XML schema, in which semantics are implicitly or explicitly defined [19]. Presently more than a thousand of these XML dialects exist. They actually result in 'EDI messages' in a new XML-look, including the lack of dynamics and flexibility.

Messages are however not being exchanged in isolation. The exchange is part of a business process in which usually multiple messages are being exchanged, though they do not all need to be in electronic format. The information definition and structure of the messages that are exchanged within the same business process should be consistent. Such consistency may be ensured by deriving all those messages from the same data model. When negotiating semantics of messages therefore one should first negotiate the

semantics and structure of the common data model. Therefore it is of interest how data models are designed and represented in present B2B communities.

Different industry sectors have used different data modelling languages. UN/CEFACT, as the co-ordinating international body, has chosen a UML profile to represent its data models. The UN/CEFACT Core Component Technical Specification (CCTS) [20] offers a way to store the data portion of UML Class diagrams in a registry and to represent parts of a UML data model in electronic (e.g. XML) messages.

2.7 Process flow

B2B information exchange is part of an inter-organizational business process. Both an ISO working group and a group within UN/CEFACT have therefore chosen the process flow as the focal point. In their view message scenarios and information content should be developed from a process perspective.

OASIS [21] published a standard for XML representation of B2B processes: Business Process Specification Schema (BPSS) [22]. BPSS is part of the ebXML series of specifications [23]. As far as known, the specification was not widely implemented.

The ISO/IEC 14662 open-edi reference model [24] describes the components of a system that supports the full interface between the computers of different organizations in an open environment. An important component in the model is the “scenario”. By means of a scenario two business partners reach a certain business goal, by expressing and fulfilling commitments to perform certain activities. An important function of an open-edi system is therefore the ability to monitor commitments. A prerequisite for any standard or agreement for automated inter-organizational collaboration is that it precisely defines the interrelationship between commitments made or fulfilled, information exchanged and steps taken during the execution of the scenario.

ISO/IEC 14662 makes a distinction between the Business Operational View and the Functional Service View on a B2B system. The Business Operational View defines the business requirements, the business protocol and the business information, irrespective of the technology used to support the business collaboration. The technology is defined in the Functional Service View.

A task force within UN/CEFACT combined the concepts of the Open-edi reference model with Object Oriented paradigms and selected the Unified Modelling Language (UML) as the technique for UN/CEFACT use in business process and information modelling. The methodology was named UMM (UN/CEFACT Modelling Methodology) [25]. UMM is not only a methodology, but has been described as a formal UML profile. UMM is now the most widely accepted implementation of the Business Operational View of the open-edi reference model.

It should be noted that UMM is based on the “design” paradigm. It is assumed that in advance of factual use of an e-business system, the system is designed by some

(standardization) body overseeing the trading partnership. UMM does not include mechanisms to negotiate about the semantics of model parts and model features. UMM is a method with which a B2B situation is modelled in a top-down fashion, and that pre-assumes that stakeholders agree on the requirements.

In UMM, the binding between process flow and information semantics is weak. A process is effectively modelled as a sequence of exchanges of messages of a certain type. The contents of those messages have no effect on the message choreography in the process models. Modelling business objects with lifecycles (that would give a stronger process-information binding) was not possible in earlier versions of UMM (it has been specified now in UMM 2.0).

2.8 Business objectives and intentions

As mentioned in section 2.1, B2B communication can be studied from various perspectives. In sections 2.2 through 2.7 perspectives were assessed that focused on one or more layers of the communication stack as pictured in figure 2.1. Another perspective is to regard business communication as a dialog between business people, supported by information systems.

Inter-organizational business processes are in fact conversations between independent peers. Business partners have their own interests and objectives. Unlike the business process within an enterprise, the interests and objectives of independent business partners are not derived from one corporate strategy. Inter-organizational processes therefore are less mechanical than intra organizational processes. Subjective intentions play an important role in business to business negotiations. A theory that studies the structure and semantics of intentions in human conversation is Speech Act theory. Findings from Speech Act theory have never reached the main stream practice of B2B systems. Nevertheless, in order to design flexible mechanisms for establishing B2B systems, intentions of business partners must be taken into account. A few attempts have been made to introduce Speech Act theory into B2B theories and standards.

Moore [26], in co-operation with Kimbrough, has developed FLBC. FLBC is a language, based on Speech Act theory, specifically targeted to B2B communication. The language supports most business interactions needed for normal business processes. An FLBC utterance or statement consists of a propositional content inside an intentional (or, in Speech Act speak: illocutionary) force. The propositional content can be a simple statement (e.g. ‘your product was delivered’), but it may be any logical combination of other FLBC utterances, so the language is recursive. Moore [27] has shown that libraries of EDI messages can be mapped to FLBC statements. FLBC has many advantages over rigid EDI (or XML) messaging. The authors of FLBC however have not described how FLBC clauses can be extracted from or included in legacy applications. Also the binding between FLBC verbs and the desired process flow has not been defined. The FLBC authors have also not made a mapping of the language elements to existing (EDI or XML) libraries. Yet, FLBC is a development that has unjustly been neglected by (at least) the standardisation community.

Jayaweera [28] has mapped speech acts on B2B communication systems as specified using UMM. He proposes to explicitly add an 'Action' and an 'Intention' to each exchange of information between trading partners. Actions include verbs such as Create, Change and Cancel, and define what the receiver is to do with the communicated information on objects. Intentions are Speech Act verbs such as propose, accept, reject, declare, query, reply and assert. The concept of actions and intentions has been implemented in EDI/XML specifications of various communities. OAGIS, an organization that develops specifications for the automotive industry, among others, have introduced a 'verb' in their messages [29]. Verbs combine the roles of actions and intentions (and of more fuzzy concepts). GS1 uses a 'Command' level in its XML messages [30]. The GS1 Command is synonym to Jayaweera's Action. In the (abandoned) UN/CEFACT draft specification on Core Components Message Assembly (CCMA) [31], Actions and Intentions are introduced as well.

Introduction of speech acts in EDI messages suggest that the electronic business conversation may be very flexible. The combination of actions, intentions and a library of business object types allow trading partners in principle to create, change and delete arbitrary business objects on-the-fly.

In chapter 6 the structure of intentional business utterances is described, making use of Speech Act theory.

2.9 Conclusion

The main cause for the lack of adoption of B2B communication is the practice that most EDI or XML connections between computer systems of different organizations need to be manually developed, based on incomplete, ambiguous standards. To improve this situation, and to support computer systems to interconnect on-the-fly, two main components are needed:

- Firstly, a protocol, or meta-protocol[1], must be agreed and standardized, so computer systems may expose their technical, syntactical and semantic capabilities to each other and "shake hands" on a communication protocol that complies with those capabilities.
- Secondly, a technical infrastructure must support such (meta-)protocol. The infrastructure consists of a network and of functions in or adjacent to application systems. These functions may also be provided by external services.

In this thesis we focus on the design of the semantic handshake protocol. The mechanism for connection to legacy software is only roughly described. The way to implement the protocol in application systems, middleware and services is described only for illustration and validation.

[1] A meta protocol is a protocol to agree on a protocol.

References

[1]. European Commission: European Interoperability Framework for European Public Services (EIF) Version 2.0, 2012
[2]. ISO International Organization for Standardization: Information technology -- Open Systems Interconnection -- Basic Reference Model: The Basic Model ISO/IEC 7498-1:1994
[3]. Bussler: B2B Integration Technology Architecture, Proceedings of the 4th IEEE Int'l Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS), 2002
[4]. Hinderer c.s.: openXchange Reference Architecture for Automated Business Process Integration, Proceedings of the 9th International Conference of Concurrent Enterprising, Espoo, Finland, 2003
[5]. Klaver, Verberne: Federatieve databases (in Dutch), Note of the Department of Mathematics and Computer Science, Eindhoven University of technology, 1987
[6]. VeriSign Technical Brief: Building an E-Commerce Trust Infrastructure, 2000
[7]. OASIS ebXML Collaboration Protocol Profile and Agreement Technical Committee: Collaboration-Protocol Profile and Agreement Specification Version 2.0, 2002
[8]. OASIS/ebXML Registry Technical Committee: OASIS/ebXML Registry Information Model v2.0, 2002
[9]. ANSI ASC X12 standard Version release 6020, 2009, available through DISA Falls Church, VA USA
[10]. ISO International Organization for Standardization: Information technology: Electronic data interchange for administration, commerce and transport (UN/EDIFACT) -- Application level syntax rules ISO 9735:2002
[11]. United Nations Economic Commission for Europe (UNECE): UN/EDIFACT Message Design Guidelines, 1993
[12]. ISO International Organization for Standardization: Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation, ISO 8824-1:2008
[13]. ISO International Organization for Standardization: Information processing -- Text and office systems -- Standard Generalized Markup Language (SGML), ISO 8879:1986
[14]. World Wide Web Consortium (W3C): XML Schema 1.0, 2004
[15]. Maler: Schema Rules for UBL and Maybe for You, XML 2002 Conference
[16]. United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT XML Naming and Design Rules Technical Specification, version 3.0, 2010
[17]. Steel: The User's Guide to the Australian Experimental BSR, The University of Melbourne, Dept. of Computer Science, 1996
[18]. UDEF - Universal Data Element Framework, www.udef.com, consulted 2010-04-04
[19]. http://xml.coverpages.org/xmlApplications.html, consulted 2010-04-04
[20] United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT Core Components Technical Specification Version 3.0, 2010
[21]. http://www.oasis-open.org/, consulted 2010-04-04
[22]. OASIS: ebXML Business Process Specification Schema Technical Specification v2.0.4 (BPSS), 2006
[23]. http://www.ebxml.eu.org/about_ebxml.htm, consulted 2010-04-04
[24]. ISO International Organization for Standardization: Information technology -- Open-edi reference model ISO 14662:2010
[25]. United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT's Modelling Methodology (UMM) UMM Meta Model – Foundation Module, 2006
[26]. Moore: KQML & FLBC: Contrasting agent communication languages, International Journal of Electronic Commerce, Volume 5, Issue 1, 2000.
[27]. Moore: Testing Speech Act Theory and its applicability to EDI other computer-processable messages, Proceedings of the 29th Hawaii International Conference on System Sciences, Volume 2: Decision Support and Knowledge-Based Systems, 1996
[28]. Jayaweera: A Unified Framework for e-Commerce Systems Development: Business Process Pattern Perspective, PhD thesis, Department of Computer and Systems Sciences Stockholm University and Royal Institute of Technology, 2004

[29]. Open Applications Group: OAGIS 8.0 Design Document, 2002
[30]. GS1 eCom: GS1 XML Technical Message Guide, 2008
[31]. United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT Core Components Message Assembly (CCMA) Technical Specification draft, rev. 1.10, 2007

3 Requirements

Summary

Business is dynamic. Business partners are independent entities with their own policy and opinions. This context poses different requirements to a B2B system than requirements regular business information systems have. The specific requirements of B2B systems are derived and listed in this chapter.

The following requirements were identified:

1. Business communication uses a language and is a protocol rather than a system
2. The process and the data are not designed but negotiated
3. In a business conversation opinions are exchanged, not facts
4. The business protocol changes over time
5. Data definitions are evolving as processes are
6. Business dynamics is independent from technology dynamics
7. The protocol must support a large variety of business practices
8. Obligations and commitments must be legally enforceable
9. Business communication has no boundaries
10. A business relation may use various communication channels concurrently.
11. Business communication may also be conducted between (potential) business partners, who just discovered each other
12. Business communication forms the interface between private internal business information systems
13. Business communication is controlled by business people
14. A Business communication system must be implementable in an environment with legacy applications and middleware

The B2B architecture that is designed in subsequent chapters is tested against these requirements in chapter 14.

3.1 Research question

Most ingredients of an architecture that supports the business dynamics are known, but they have not been combined in a consistent overall theory or set of standards. The main scientific challenge is to reconcile the various perspectives, such as computer networking, syntax manipulation, service orientation, data modelling, ontology engineering, process and workflow theory, business goal matching and combine them into an overall architecture.

The central research question of this thesis is:

What IT architecture allows enterprises and other organizations to interconnect their business information systems on all levels, taking into account the requirements of flexible and dynamic business processes, without the need for manual configuration and programming each individual connection?

An IT architecture is defined here as the set of (soft- and hardware) components of an information system, with the structure that relates the components to each other, together with the methods and procedures to operate, maintain and enhance components and structure. In this thesis we concentrate on that part of the IT architecture that is used to interconnect organizations. The internal architecture (often called the 'Enterprise Architecture') of the organizations is largely considered as a boundary condition.

The overall perspective we have chosen is the perspective of doing business in general. So we do not depart from a specific technology or technological level.

3.2 Requirements of business communication systems

Business communication systems possess a number of characteristics that are different from, or have different aspects then traditional business information systems. Information Technology to date focuses on the development of business information systems. It is therefore useful to make explicit in which B2B communication systems are different from business information systems and what aspects are dominant.

In this section a number of characteristics of B2B systems are identified. They are derived from the characteristics of a free (though regulated) open market. The identification is based on expert knowledge: an experience of 25 years in industry and in business communication.

The following issues are identified:

1. Business communication uses a language and is a protocol rather than a system
2. The process and the data are not designed but negotiated
3. In a business conversation opinions are exchanged, not facts
4. The business protocol changes over time
5. Data definitions are evolving as processes are
6. Business dynamics is independent from technology dynamics
7. The protocol must support a large variety of business practices
8. Obligations and commitments must be legally enforceable
9. Business communication has no boundaries
10. A business relation may use various communication channels concurrently.
11. Business communication may also be conducted between (potential) business partners, who just discovered each other
12. Business communication forms the interface between private internal business information systems

13. Business communication is controlled by business people
14. A Business communication system must be implementable in an environment with legacy applications and middleware

1. Business communication is a protocol rather than a system

Most business information systems are modelled as a database, where users can add information to, get information from or manipulate information of. In fact the information system functions as the communication system between users. The system bridges the time, the place and the format of the information. A business system within an organization is an asset of that organization. It is designed and configured according to the requirements and policy of that organization.

In an inter-organizational communication system the information that is exchanged between business users is bridged my more than one system. The communication between the systems is a representation of the communication between the users. The borderline between the (responsibilities of) the users does not lie within a system, but between systems. The communication over that borderline is described by means of a protocol.

Inter-organizational business communication is communication between business people, who are acting as independent (economic) agents of their employing organizations. Each user deploys his own system. Installing one central system between the business users is feasible, nor desirable. There exists no central authority that would govern the requirements. Users must be free to choose system providers, independent from each other.

When determining the requirements to the systems and the infrastructure, one should therefore start with modelling the characteristics of the interaction between people. People, also business people, use languages to interact. In a language, signs (sounds, words, sentences) form patterns. The sequences of patterns that may be exchanged form a protocol.

Only after the characteristics of the protocol have been determined, the requirements to the systems and to the infrastructure can be defined.

Req1: Part of the architecture must be a protocol

2. The process and the data are not designed but negotiated

Many authors, who have written about inter-organizational systems, envisage the design and development of such systems as a traditional development process. They take a centralized viewpoint, either from a neutral perspective (assuming that both parties recognize some neutral authority), or from the point view of one of the parties (assuming that it has sufficient market power to force the other party).

A business communication system cannot be regarded as a fixed mechanism, but is rather a social environment in which independent peers constantly negotiate the way they co-operate. Power and trust levels change over time and influence the processes and information requirements. No real distinction exists between "design time" and "run time": the design of the system in fact is adapted and refined during run time.

A business communication system that supports the dynamics, existing in business, therefore needs to be capable of adapting the protocols based on the data exchanged in those protocols. The data and process negotiation process must be part of the protocol.

Req2: The architecture must support negotiation of process flows and data structures

3. In a business conversation opinions are exchanged, not facts

A business information system within an organization lets users store information. Users are authorized to do so. A piece of information, stored by authorized users is regarded as a fact by the organization. Procedures are in place to guard the information validity and the responsibility of the users. The organizations management is the authoritative body that assigns and controls responsibilities and user authorizations.

In inter-organizational communication, however, such authoritative body does not exist. Users are peers that may accept each other's utterances to a certain degree of truth. Utterances are subjective. One business partner may for instance claim that a product has been delivered, while the other may allege that delivery has not taken place.

Utterances may not only bear a different truth value for each partner, they also have a deontic value that may or may not be accepted by the partner. An order, placed by one partner, may be rejected by the other. Requirements (e.g. for a delivery) are subject to negotiation.

Where in most intra-organizational information systems the subject who entered some piece of information is logged in a separate file, only to be accessed in case of irregularities, in inter-organizational systems the subject is an essential part of the information itself, needed to control the business logic. In addition to the subject, his intention or deontic value is also part of the information.

Therefore information, exchanged between business partners should not be regarded as a collection of facts, but of opinions.

Req3: It must be possible to exchange intentions, not only facts

4. The business choreography may change over time

In most business information system development projects, the organization to use the information system is regarded as a deterministic mechanism. Business processes prescribe the tasks and responsibilities of the various users and user groups. The processes are defined by the management of the organization and refined by interviewing domain experts. Management policy and domain expertise are translated into (mechanistic) models that serve as blueprint for the system.

In a market economy however, the market space in which organizations cooperate cannot be envisaged as a mechanism. There is no central authority that may impose its policy. Business partners are autonomous peers, and are not subject to some central co-ordination or authoritative policy. The way supply chains are organized is the result of permanent negotiation. The economy is an ecosystem in which always must be space for innovation. Standardization of innovative business processes kills competitive advantage.

The design and development of inter-organizational processes is therefore a business process in its own right. Not the business processes themselves should be standardized, but the dynamics that may create them. Business process flows and information requirements are not determined in the setting of a systems development and implementation project, but during another business process

Process definitions can be nested. Business processes should be able to define (refined or lower level) processes. Information and Communication Technology should support such an environment.

Determinants of process choreography are factors such as trust, transaction frequency, transaction value, perceived risks, technical product complexity, stability of the trade relationship, logistic control maturity, tradition and legal constraints. Implicitly or explicitly those factors are being translated by business people into profit margins and risk levels. Those are being traded off based on a business strategy that includes elements such as (future) market share, market power, profit, customer satisfaction and loyalty and public image.

At a traditional marketplace entrepreneurs themselves may negotiate, buy and sell, or delegate that to commercial salespeople or procurement officers. In an electronic environment not only price and product quality needs to be negotiated, but process flow as well. Business people are no modelling experts. The possible process configurations therefore need to be derived from strategic options that are stated in business language and translated back into cost, profit and risk. The same is true for the specification of the information that needs to be exchanged.

The process negotiating mechanism in a B2B environment should enable business people to assess the options in these for them relevant concepts. The option consequences should be presented to them in a language they understand.

All determining factors for process choreography and information exchange requirements are dynamic. So also the choreography is dynamic and changes over time.

Req4: The architecture must support adaptation of the process choreography

5. Data definitions are evolving as processes are

Language is dynamic. As business practices evolve and as technology progresses, the language must adapt. New products are introduced that get new names and may possess characteristics which existing products did not have. New business procedures are introduced as logistic control improves and as legislation changes. New procedures often involve new concepts that need to be named and defined. Natural language allows the introduction of new concepts. Natural languages are 'alive'. Language itself is developed by the users of the language.

An infrastructure that supports structured business communication can only be viable, stable and scalable if it also supports dynamic evolution of the business language. Language evolution is a decentralized process. Language innovations are not decided upon by the assemblers of dictionaries. Dictionaries register how the language is evolving; they do not control the evolution. There is no central authority that can prescribe semantics. Semantics are defined by business people in the course of business processes.

As business processes evolve, the requirements of the information to be exchanged in each process step change as well. Translated into messages or transactions, this means that the structure of those messages and transactions change over time.

Req5: The architecture must support adaptation of the data structures

6. Business dynamics is independent from technology dynamics

Of course technology advancement influence business processes, but business processes are not directly dependent on the technological innovation. Business has its own dynamics. Business dynamics depend on market developments, which are heavily influenced by cultural change.

Advancement of information technology is partly an autonomous process (IT as a science) and partly dependent on specific market development (IT as a product). The latter may have more interdependence with the evolution of business processes (e.g. with regard to the adoption rate of IT innovations), but still business process evolution and IT use are relatively autonomous processes.

An architecture for the support of business communication by information technology should therefore be layered, in order to decouple business process evolution and technology innovation. The business requirements layer should be different from the layer in which technology is defined to support those requirements.

Ideally, one should be able to replace the total technology layer without affecting the business functions, or to completely alter business processes without touching the hard- and software of the infrastructure.

Req6: The architecture must be independent from the technological implementation

7. The protocol must support a large variety of business practices

Businesses are very dissimilar. The business scope is not only agriculture, chemicals, retail, steel manufacturing, music, food, apparel, cars and electricity, but also medical services, insurance, governmental licensing, auditing and environmental reporting, among other. Within each of those sectors a large variety of trade procedures, processes and workflows exist. Ultimately, each individual company, to stay on its competitive edge, probably behaves (and therefore communicates) somewhat differently than its competitors.

The context of a business relationship is a hierarchical framework. If trade is conducted with an unknown company in a distant country, rules apply derived from international trade law. Regional and national legislation further regulates trade relationships within its jurisdiction. Customary trade procedures in the applicable industry sector pose additional constraints. For buying and selling soybeans one follows different procedures and conventions than for buying airfreight services. Within those limits specific processes may bilaterally be agreed. When trade is repeating and trust levels increase, the processes may be altered, as processes that keep risk levels low (e.g. trade through documentary credits) are usually not most efficient.

It is not feasible to cast the huge variety of business procedures and processes in rigid standards. It is also undesirable, as it would force businesses to comply which would kill innovation. Business process innovation would be slowed down to the pace of an (international) standardization process. An infrastructure that would recognise the variety by allowing businesses to specialize their process choreographies as they operate in a more specific environment or context would be more feasible and would have a higher adoption.

Req7: The architecture must not be specific for a certain business environment

8. Obligations and commitments must be legally enforceable

The ‘default’ procedures of course should be compatible with international trade law. Especially the legal group of UN/CEFACT has done considerable work to translate lawful trade procedures into e-business processes. Trade law may affect the way transactions (the lowest level information exchange) are managed. Transactions should be monitored and a business level mechanism should be in place to manage them.

Liabilities and commitments must be clear and legally enforceable. The communication track records should be presentable as proof in legal disputes. The communication should be inherent reliable.

Business is used to communicate by means of paper documents for centuries and the enforcement practice has been tailored to that form of communication. Electronic communication in general can be made much more safe and secure than paper communication. The legal system and its enforcement procedures however must adapt to the new technology.

Req8: The architecture should contain instruments to enable enforcement

9. Business communication has no boundaries

Business is not limited to national borders or geographical boundaries. Any national or regional solution fails to support trade that passes the border.

Some theory claims that within a distance of five tiers (a friend of a friend of a friend of a friend of a friend) the world population is everybody's friend. In business the distance is probably smaller. Sector oriented standards assume that trade is being conducted within an industry sector. The truth is that trade is usually conducted *between* sectors. A company's competitors seldom are its clients.

Business moreover interferes with all other disciplines: government, accounting, production technology, medicine, geography, etc. Standards that are specific for business or trading only are not sufficiently open. Business communication systems should be based on generic requirements to human communication, with additional specific business functions and features. They must be extensible to vocabularies and processes in other disciplines.

Req9: The architecture must not be specific for a geographic area

10. A business relation may use various communication channels concurrently.

In many cases some communication channel will only be used for some aspects of the business relation. For instance in many cases for the discovery of trading partners the (not very well structured) World Wide Web is used, catalogues and quotations are exchanged by e-mail or fax, orders are placed by EDI, delivery document come on paper, invoices are typed into an extranet and payments go through the bank channel. Business communication solutions that assume all communication passes the same channel and that fails if it misses a link, is not useful.

The same is true for exception handling. Every process should have escape possibilities. No-one will completely model the course of a law suit in case of breach, if it is even possible, it will certainly not be completely be supported by the e-business system.

Req10: The architecture must be capable to use various technologies concurrently

11. Business communication is also conducted between (potential) business partners, who never did business before

Traditional EDI systems are only implemented by business partners with long standing business relations. Many business partnerships are however incidental and each long lasting partnership once started with an incidental or probing transaction. One might say that EDI was so successful in some industries because it never was applied to incidental business. So it was only successful because it was unsuccessful (for general trade).

A really successful B2B system should solve this 'first trade problem'. No investment (in hardware, software, services or skills) should be needed to use the system, beyond the investments that are needed to participate in business at all.

Business communication systems should support communication between strangers. No previous agreement or even contact should be assumed. When trust levels are low, processes should simply fall back to a level where risks are being managed satisfactory. Technology should be self-adapting to the other peer.

Req11: The architecture should not assume prior agreement between trading partners

12. Business communication forms the interface between private internal business information systems

A B2B system is not just a system to connect people, but to interconnect internal private information systems that process data in a structured way conform private data and process definitions. A B2B architecture should therefore be very tolerant with regard to technology levels that are assumed. Internal business information systems are of very different make, age and architecture. They may consist of large integrated ERP packages, of loosely or manually coupled landscapes of small systems or be totally paper based.

An inter-organizational communication system for business people already exists, it is called 'snail mail' (the postal system) in combination with 'POTS' (the Plain Old Telephone System). B2B communication is to enhance those traditional systems by interconnecting automated information systems of business partners.

Req12: The architecture should be capable to include legacy applications

13 A Business communication system must be implementable in an environment with legacy applications and middleware

No architecture or standard is viable if it requires a total overhaul of the present practices and the legacy infrastructure. It is also not viable if it needs a complete new set of standards and if the components defined are incompatible and inconsistent with existing

ones. A strong requirement to the architecture to develop is therefore that it does not completely replace the legacy, but offers a smooth migration path for existing systems, practices and standards. It should use parts of the legacy where possible.

A global ubiquitous e-business communication system should be an environment where business processes can evolve with the business. It should not be dependent on a specific technology, as technological and business developments have different dynamics. Process choreographies and information definitions (ontologies) should not be designed by some central authority, but innovate with the businesses. However, mechanisms should be in place so they converge where possible (and diverge where needed). Differences should be triggered by business objectives and not by technology or arbitrary modelling decisions. The e-business environment should form an ecosystem where business lives, dies and evolves, not hindered by technical barriers or battles.

Req13: The architecture should be capable to be implemented in a legacy environment

Summary

Req1: Part of the architecture must be a protocol
Req2: The architecture must support negotiation of process flows and data structures
Req3: It must be possible to exchange intentions, not only facts
Req4: The architecture must support adaptation of the process choreography
Req5: The architecture must support adaptation of the data structures
Req6: The architecture must be independent from the technological implementation
Req7: The architecture must not be specific for a certain business environment
Req8: The architecture should contain instruments to enable enforcement
Req9: The architecture must not be specific for a geographic area
Req10: The architecture must be capable to use various technologies concurrently
Req11: The architecture should not assume prior agreement between trading partners
Req12: The architecture should be capable to include legacy applications
Req13: The architecture should be capable to be implemented in a legacy environment

In chapter 14 the architecture, that is designed in the subsequent chapters, is assessed against these requirements.

3.3 Methodology

In chapter 1 the socio-economic problem is stated that B2B communication is not advancing as IT adoption is. With the progress towards a networked economy that phenomenon hinders or even blocks economic development. The reasons of the lack of B2B adoption are analysed in chapter 2, where the present practice of B2B implementation is described. By assessing the methods and technology used for B2B implementation the instrumental problem surfaces. The main instrumental problem appears to be the rigidness of B2B semantic standards, while business is diverse and dynamic. The requirements of an architecture that would solve the problem are listed in section 3.2.

In the subsequent chapters of this thesis that architecture is designed. The requirements form the basis of a reference model of B2B communication. That reference model is presented in chapter 4. First business as a phenomenon is analysed. As business people are humans, the analysis will start with a fundamental investigation how human communication is used in business processes, abstracting from media or IT support.

Information technology developed from the perspective of an individual organization that replaced paper file cabinets and mechanical calculators for automated devices. In the early days of B2B communication, these computers were interconnected at the data level. Business intentions and the process flow were implicitly assumed. To develop an architecture for open B2B communication, such assumptions cannot be made and the business conversation to negotiate a process flow, including subjective business intentions, need to be part of the architecture. Therefore we need to change the perspective from data level interconnection to business conversation.

The model is derived from the theory of semiotics that explains how humans communicate and what the roles are of the communicating subjects, their view or model of the world, signs and a communication channel. This 'semiotic triangle' is applied to B2B communication, where the world models are stored in a business information system and where the communication channel is a B2B system.

When two people communicate, their views of the world must at least partly overlap. This intersection of world models is called their universe of discourse and the information on the universe of discourse is kept in a common or shared virtual knowledge base. Physically, that information is stored in the business information systems the partners deploy. A synchronisation mechanism synchronizes the information systems.

The remainder of chapter 4 and the chapters 5 and 6 are dedicated to the structure of the knowledge base and consequently to the structure of the information that needs to be exchanged in order to synchronize the information systems.

The knowledge base is not static: new concepts and new observations are added in the course of the business process. In section 4.7 therefore the mechanism is designed how to define new concepts. This mechanism is based on ontology engineering and more specifically on conceptual graphs. When a partner defines a new concept, he should use terms that the other partner already is familiar with. A new concept can be defined as a more specific subtype of an existing concept. In order to formalise definitions, concepts must be defined in terms of their properties. New concepts narrow the property space of their super-type. This definition mechanism is recursive and the starting point may be some agreed industry ontology.

In chapter 5 then other utterances that may be exchanged between business partners, such as observations, are analysed. Linguistic theories are used to find the utterance structures. It appears that all utterances have a basic core structure, consisting of a verb and two nouns or concepts that fulfil thematic roles in the utterance. The rule is introduced that

each utterance must be based on a previously exchanged utterance in order to be meaningful and to be understood.

The utterance structure is elaborated in chapter 6. Utterances have a number of attributes that may be represented in a table. The table makes it possible to illustrate the mechanisms described. Attributes include timestamps, uttering partner, intention, cardinality, preconditions and future subtyping.

Chapter 7 describes how process choreographies may be defined and negotiated by means of populating the knowledge base. The process choreography is tightly bound to the information exchanged, and may be guarded by means of informational states of business objects.

In chapter 8 a mapping is shown from the knowledge base structure to existing modelling and exchange languages, such as UML, ORM, SQL and XML. The purpose is to show that the described mechanisms may work in a legacy environment with existing EDI or XML messages and with existing application systems and databases.

The architecture allows business partners to discuss and negotiate at the meta-level about information requirements and availability and about process flow. Such discussion must start from a common basis or generic business ontology. In chapter 9 an ontology, based on Resources, Events and Agents is introduced. That ontology is sufficiently generic and may be specialized for the various business domains and business situations. How business communication and the options to choose match with business strategy is briefly addressed in chapter 10.

Chapters 11 and 12 illustrate the implementation of the architecture in a few example cases. In chapter 13 the architecture is assessed against the requirements in chapter 3.

PART II DESIGN

4 B2B reference model

Summary

Business communication may be viewed from various perspectives. In this thesis business communication is regarded as the enabler for business people to reach a common goal by exchanging information and resources. To make this view explicit, in this chapter a reference model for business communication is presented.

In the model it is assumed that information that is exchanged between business partners creates knowledge that is shared by them. That knowledge may be regarded to be stored in a virtual knowledge base, common to and accessible by both partners. In reality, the information that created the knowledge is stored in the private information systems of the partners, that needs to be kept in sync by means of data exchange.

Partners feed the knowledge base by means of utterances. As the information contained in the utterances must be understandible by both partners, the information elements must be defined beforehand. These definitions are also expressed in utterances. So earlier utterances create a filter for allowing later utterances to be included in the knowledge base.

4.1 Introduction

The main gap in theory and practice for dynamic and flexible B2B systems is the lack of a handshake protocol to agree on the technical, syntactical, semantic and process aspects of the interface between the computers of independent organizations. In this thesis such a protocol is developed and proposed. In order to do so, computerized business communication as a phenomenon is analysed first. This analysis results in a model that serves as a reference model for the design of an abstract protocol. The protocol is described in chapters 5, 6 and 7, taking into account the requirements listed in chapter 3. How the protocol may be implemented in existing modelling and exchange languages is described in chapter 8.

B2B communication may be analysed from various perspectives. One perspective is to start at the bottom of the communication stack, and to describe how data is transferred between computers and operating systems. This perspective is chosen by Bussler [1] among others. Technology is however rapidly outdating such analyses. Moreover, the way data is transferred and formatted is not of concern of the business. Business variation concerns process flows and semantics, not data transmission.

Another perspective is document and data definition. This perspective is chosen in [2]. The focus in this perspective lies on the structuring of the information to be exchanged

between business partners. In traditional EDI (and modern XML) standards, data elements used in messages are being defined by means of human readable definitions. In ontologies such definitions are formalised and made computer interpretable. Ontologies however describe an objective model of reality, not taking into account subjective intentions business partners may have and not addressing process aspects of business relations.

The perspective taken in this chapter is a more fundamental view on business communication. Business is performed by humans, who use their computer systems as communication means. Computer systems are tools for business persons to bridge time, location and representation of the information they exchange. In order to derive the needed handshake protocol and to identify the elements needed in systems that execute such protocol, a reference model is needed that describes such systems and the processes that are being performed by them.

In this chapter, as part of the architecture of open B2B communication, a reference model is presented in which the aspects of a B2B relation are described and positioned in relation to each other. The reference model is based on general principles of human communication and independent from the technology used. The model is used as reference for the design of a dynamic B2B protocol.

The protocol to negotiate and agree upon semantic aspects is in this chapter described on a high level. The protocol will be elaborated and illustrated in subsequent chapters.

The reference model is independent from the technology deployed (Req.10 in section 3.2). A model of inter-organizational business communication processes should not depend on the particular IT applications the communication partners use. It should be the same for companies using a paper based administration or a computerised one. The same model should be applicable to companies using a centralised ERP system and to organizations that have installed a patchwork of interconnected systems.

The reason for technology independence of the reference model is twofold. Firstly companies should be able to do (electronic) business independently of their IT infrastructure. As companies are autonomous, they even should not need to be aware of each other's infrastructure. Secondly, the handshake protocol can only be future proof if it is (largely) technology independent. Technology is rapidly changing, while business relations and trading customs are not.

Therefore the reference model is based on general principles of human communication. The model is focused on rational cooperation and coordination of measurable activities. Non-rational interactions, such as non-verbal communication, utterances of affectionate feelings and poetry are left out of scope. Business communication is about the establishment of legally enforceable agreements and about fulfilling those agreements.

The general communication principles are applied to inter-organizational communication. The involvement of third parties, such as agents, banks, governmental bodies and other outsiders, who influence the communication context, are included in the model.

The role of information systems deployed by the communication partners is also considered and modelled. Such an information system does not need to be automated; it may be paper based or use an arbitrary technology. The reference model includes the systems' functionality, not its technology.

After having modelled the principles of inter-organizational communication and the role of information systems, it is possible to describe how semantic reasoning among business communication partners is functioning. A protocol to negotiate and agree on semantics and business process flow can be based on such a model of inter-organizational communication.

4.2 Business Conversations

Ogden and Richards in 1923 created a model for human communication [3]. The model has the shape of a triangle, of which one of the corners is the human subject who participates in a conversation. Another corner is a thought or an observation by the subject on some real world phenomenon. The thought/observation is symbolized in a symbol or sign, the third corner. The sign can be a sound, a gesture, a picture or a natural language sentence. If computers are used the sign may be a file, a record or part thereof. A sign may be binary encoded. The sign is sent to a recipient over a communication network. The recipient interprets the sign which induces a similar thought in his mind as the thought the sign represented. See figure 4.1. The model is known as the "Semiotic Triangle".

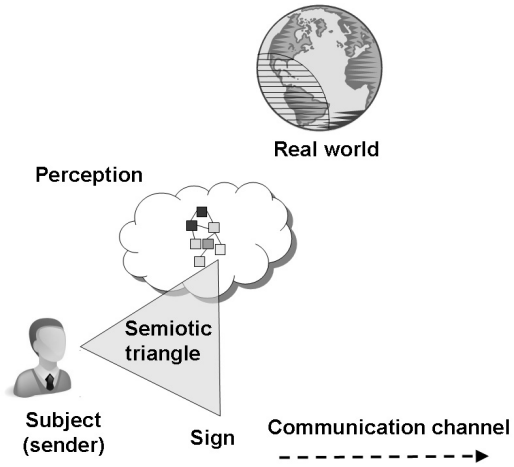


Figure 4.1 Semiotic triangle

Obviously, the recipient can only attach the correct meaning or semantics to the sign, if he has a reference framework that is similar to that of the sender. Both must be able to observe (or at least imagine) the same part of the real world the sign refers to. The part of the real world on which the parties share observations (or imaginations) is called their ‘universe of discourse’. Events that happen in that shared world may affect both parties. Both parties have a perception or view on that world. Meaningful communication is only possible if both parties refer in this communication to the same world and if at least part of the views can be aligned, see figure 4.2.

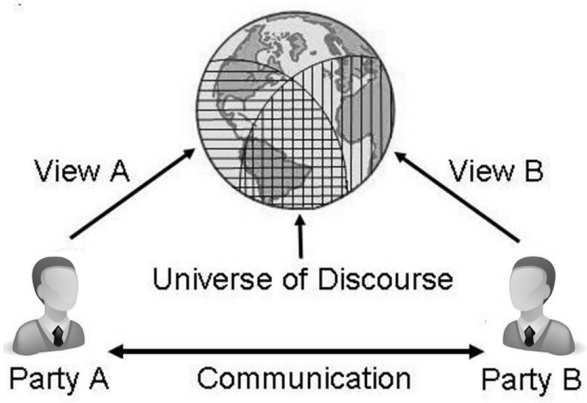


Figure 4.2 Universe of Discourse

The semiotic triangle model has been extended and refined by Saussure [4] and Stamper [5].

Saussure [4] mainly studied the relation between the real world observations and the sign system, especially with relation to natural language: linguistics. An important part of Saussure’s work is dedicated to the relation between an individual sign and the real world phenomenon it represents. Saussure states that while the sign 'may seem to be freely chosen', from the point of view of the linguistic community it is 'imposed rather than freely chosen' because 'a language is always an inheritance from the past' which its users have 'no choice but to accept'. In the context of a business relationship that means that both business partners need to agree on the same sign system in advance of any operational communication and negotiate any extensions to the system before new sign(-combination)s can be used.

Stamper et al [5] propose to use the semiotic model directly for designing and developing information systems. In their view an information system as a model of the real world affords users to act and communicate, and norms implemented in the system constrain those affordances. In fact database-, message-, and process schemas follow these same ideas: they limit the use of signs and sign structures by the users or communication partners. In Stampers terminology, such schemas can be regarded as norms that limit affordances.

In this thesis the semiotic triangle is used as the basis for a B2B reference model. Later the findings of Stamper and Liu [5] are used to implement norm negotiation, thus making B2B systems dynamic and extensible.

4.3 Knowledge Base

The view that a party has on the real world is called here the "knowledge base" of the party with respect to that world. A party's knowledge base consists of all his observations, inferences and decisions. We assume that the knowledge base is structured in some way. In fact we assume that only rational, organized parties can do business with each other. As noted above, irrational, affectionate and poetical aspects are left out of scope.

The knowledge base, which a party has built up, is a model of the real world. It contains observations of real world objects, inferences from those observations (such as generalizations and predictions) and decisions that may have been derived from the party's goals and emotions. An important objective of communication is to align the knowledge base of the sender with the knowledge base of the recipient. Only when the knowledge bases of business partners are aligned and contain similar views on the universe of discourse, the partners can co-ordinate their behaviour, creating synergy to reach their goals.

We can combine the figures 4.1 and 4.2, and extend these figures with the knowledge bases as depicted in figure 4.3.

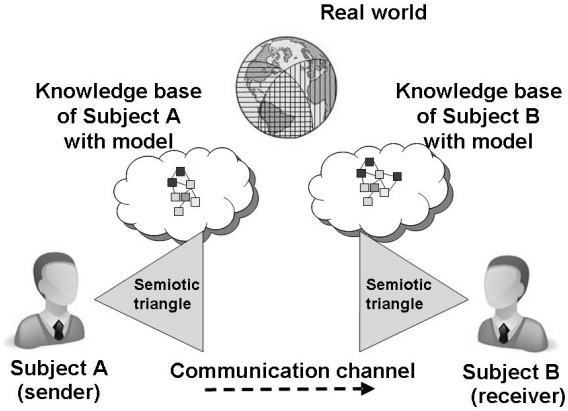


Figure 4.3 B2B Reference model

Now when subject A observes an event in the universe of discourse, he[2] updates his knowledge base accordingly. In order for the business partner, subject B, who may not have observed the event, to react in an adequate way (so both subjects can ultimately reach their business goals) subject A sends the model update to subject B over a

[2] Throughout this thesis, 'he' also may refer to 'she'.

communication channel. In case he does that by using natural language or by using an electronic network, he needs to serialize the updates as a string of alphabetic characters or even (ultimately) bits. On receiving the signs subject A has sent him, subject B updates his knowledge base.

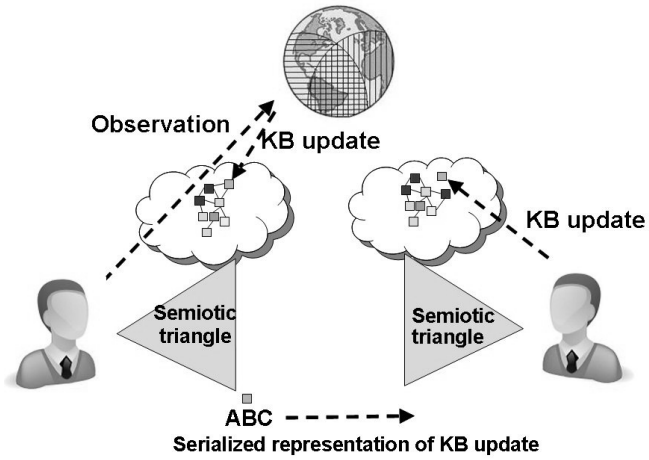


Figure 4.4 Knowledge base update

Both parties build up a common knowledge base by communicating. The common knowledge base is distributed over the information systems (or the minds) of the two parties. The common knowledge base is a model of the universe of discourse. The knowledge contained in the common knowledge base is the intersection of the knowledge in the two private knowledge bases.

Whenever knowledge in a private knowledge base is updated that is of relevance to the collaboration, the update is forwarded over the communication channel. That way the common knowledge of the trading partners stays in sync. Of course for tactical and technical reasons the updating process may be delayed. We assume, however, that trading partners are not deliberately corrupting the information (lying), although they may filter it. The common knowledge should remain a subset of the knowledge hold by the two partners privately,

The updates that are sent over the communication channel can be regarded as utterances that update the common knowledge base. Abstracting from technology and implementation, we can regard a business communication system as one common knowledge base that is updated by means of utterances of the partners.

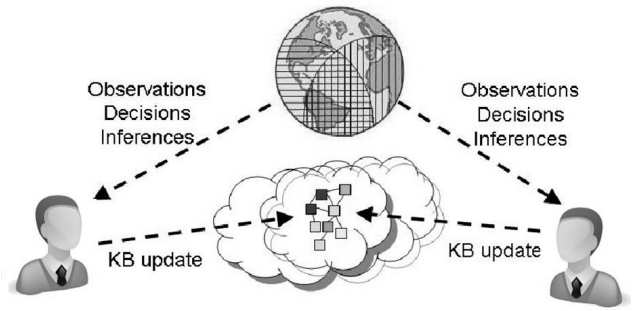


Figure 4.5 Common Knowledge base

A business conversation thus consists of a series of statements or utterances, alternately (but not strictly alternately) uttered by two business partners. Utterances are composed of signs. Utterances relate to real world events that a party has observed (such as the production of goods or the delivery of consignments) or to decisions the party has made (such as the decision to purchase products from the other party). Between the parties there must be some communication system installed that relays the utterances. That system can be as simple as the air (allowing the parties to talk to each other) or be as complex as a network connection between the parties' ERP systems. In both cases the information exchanged in the conversation between the parties has the same meaning, at least commercially and legally.

If the parties' intentions and goals are thought to belong to the universe of discourse as well, in fact the (only) purpose of communication is the alignment of the views as far as that is relevant to the behaviour of the parties towards each other. That behaviour, by the way, is also part of the universe of discourse. In business, the real goals and intentions are of course not always revealed to the trading partner, especially not when the conditions of a transaction are being negotiated. The common knowledge base only contains those intentions and goals that partners wish or need to share.

What in principle happens is:

- A party sets his (long term) goal and stores it in his world model (adds it to his view on the world)
- The party declares his (short term) intentions and desires and adds them to his world model
- The party takes decisions on topics he has power over and adds those decisions to the model
- The party takes actions, changing the world physically and records those in the model
- The party observes changes in the physical world and records those changes in his model
- Parties align their models by means of a communication system.

This process is illustrated in figure 4.6.

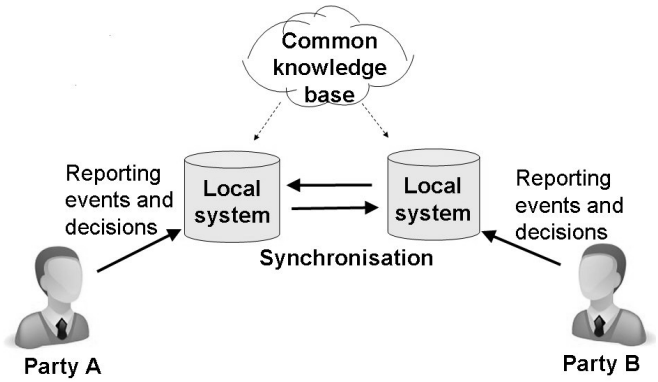


Figure 4.6 B2B communication system

The two views to be aligned are models of the world. These models may be mapped on a scale of structuredness: the knowledge base may be very intuitive, very structured or somewhere in between. In this thesis the focus is on computerized business information systems. If the model is to be manipulated by computer programs, it should be towards the more structured end of the scale.

Note, however, that computers can very well be (and are) used to process unstructured information, such as free text, speech or pictures. Word processors and e-mail programs usually treat documents and messages as large binary objects rather than as structured collections of information. In an ERP system much of the information is not defined exactly. Often the exact meaning of codes, e.g. pricing conditions or production statuses, is known intuitively by employees using the system, while for the system itself it is 'just another code'. The meaning of information elements is usually only described in free text. If the business communication between parties is to be supported by computer systems that (partly or entirely) interpret and process the semantics of the common knowledge base and calculate or derive business actions and consequences, the information and process steps need to be structured and defined formally. In this thesis therefore the main focus is on the communication of highly structured information, although it will be discussed how fine grained that structure should be.

4.4 Metadata and filtering

The knowledge in the common knowledge base must be understandable by both trading partners. If one stores his knowledge in Greek, and the other in Chinese, the knowledge stored is not commonly understood (except when both parties know both languages). The same happens when the knowledge consists of information on concepts that are not mutually understood. If one partner is a chemical factory that stores information on chemical reactions with formulas, and the other is a transport company that is only 'knowledgeable' about logistic procedures, the knowledge is not common as well. The

common knowledge base must contain a mechanism that ensures that both parties understand the common knowledge that is stored.

One of the possible mechanisms is to maintain a metadata schema of the knowledge base. In the metadata schema the definitions of the concepts are stored, with the structure of the information on those concepts. The metadata schema functions as a filter. Only transactions or utterances that obey the schema are stored. An utterance that is not compliant to the schema (e.g. because it refers to a concept that has not been defined, to an association that is new or because it violates cardinality rules) is refused.

This mechanism is used in traditional EDI systems. Prior to actual communication, a standard messaging scheme (or an adapted standard messaging scheme) is agreed upon. Messages that do not obey the scheme are refused by the Value Added Network or by the middleware of the receiving partner. Of course the sending partner is informed about the refusal.

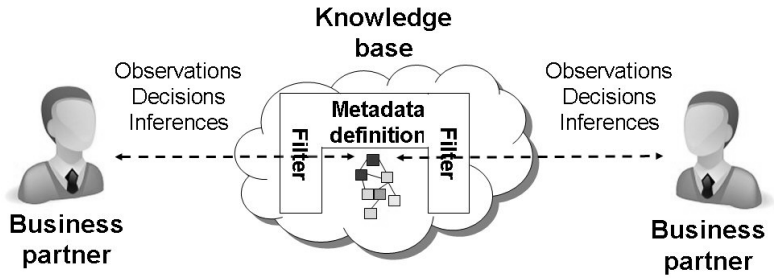


Figure 4.7 Metadata based filtering

However, in a dynamic B2B system the metadata must be extendable and negotiable. A business partnership must be capable of changing the trade procedures and of defining new concepts. The chemical factory must for instance be able to agree with the transport company what transport conditions and precautions are needed for which chemical substance. Hence the factory has to be able to define the properties of substances in some proper manner.

The metadata must allow for manipulation by the business partners. Manipulation needs to be done in the course of a business process. When the logistics manager of the chemical factory sits around the table with the account manager of the transport company, they may agree on logistic procedures (planning, frequency, equipment, legally required safety precautions) to be followed during operations. They agree on the metadata of those operations. A computerized B2B system should support such negotiations.

In most systems the manipulation of the metadata takes place at another ‘level’ as compared with the manipulation of data. In the OMG MOF, data manipulation is performed on layer M0; obeying rules that are stored at layer M1. Metadata manipulation

is done at layer M1, according to rules at layer M2. In most modelling and database languages different language constructs are used for the different layers.

In chapter 5 we propose a mechanism by which the metadata in a knowledge base is defined by the data itself. In practice the metadata definitions may be mapped to the M1 level of the modelling language used. Such mapping is described in chapter 8.

4.5 Inter-organizational communication

Abstracting from technology, an inter-organizational business process is a business conversation between two parties. These parties are autonomous organizations. In this thesis the focus is on how these organizations behave towards each other. How parties are organized internally is outside the scope of the thesis.

Internal organization includes the way, in which tasks and responsibilities are delegated to employees, subcontractors or external agents. In most cases, a party's internal organization is not of concern to the other party. Sometimes the other party is not even allowed to know who is the agent (employee or subcontractor) who is acting on behalf of his business partner or opponent. Here, therefore, parties are considered to behave towards each other as single, coherent entities without internal structure.

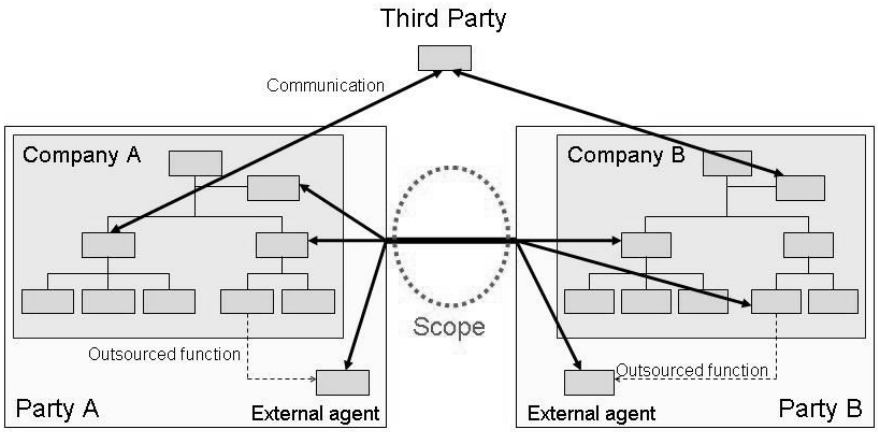


Figure 4.8 Scope of a B2B communication system

The inter-organizational process must be connected to the companies' internal processes. The requirements for automated support of the inter-organizational process are derived from the internal processes and the implementation of those requirements is negotiated between partners. The negotiation space is determined by the policies and capabilities of the negotiating partners. This is elaborated in chapter 10.

The reference model describes the communication process between two parties. One of the characteristics of a market is that companies have relative freedom in shaping the

trade relationship with each of their trading parties independently. Legal obligations always exist between two trading parties. Many business processes however involve more than two parties. Such multiparty collaborations are composed of multiple binary communication processes. Each pair may negotiate a separate process with separate requirements.

An intermediary such as a bank, a carrier and an auctioneer acts as an agent of one of the trading partners. Sometimes they simultaneously act as an agent of both, but that does not change the relationship between the trading partners. Governmental agencies like customs or tax authorities enforce or permit activities to be fulfilled by each trading partner individually. From the perspective of the inter-party relationship, such fulfilment can be seen as an internal process of a party.

In the reference model the existence of multi-party collaborations (multi>2) is not ignored, but always mapped to bilateral ones. This way the flexibility and dynamics of the business processes are maximized.

A system to support inter-organizational business processes is different from traditional business information systems that are used by individual organizations. In such a business information system, the organization to use the information system is regarded as a deterministic, or at least a hierarchically coordinated, mechanism. Business processes within an organization are defined by the management and prescribe the tasks and responsibilities of the various departments and employees. When a business information system is developed, management policy and domain expertise are gathered and translated into (mechanistic) models that serve as a blueprint for the system. In an inter-organizational system the communicating parties are autonomous peers without a hierarchical relation. The communication process is negotiated among those peers instead of having been defined by a higher authority. The peers' freedom of movement however may be limited by legislation or by previous (multilateral) agreements, such as industry sector convenants.

The negotiation process is in itself a communication process. An organization may have many trading partners, up to several thousands. A system to support inter-organizational business communication therefore must be tolerant to many different variations of the communication processes and support the continuous evolution and alteration of the processes.

The knowledge base of a party contains more data than is relevant for the collaboration. A party's knowledge base also contains private data and data that is only relevant to the collaboration with other partners. A bilateral common knowledge base is a filtered view on the private knowledge bases of the communicating parties.

The common knowledge base, which contains the model of the part of the world that is shared between the communicating parties, may be regarded as a virtual database. That database is the abstract representation of and a view on the synchronized databases of the

parties. Abstracting from the synchronization process, the communication acts or utterances of the parties can then be regarded as transactions to that database.

The history of the conversation is significant for the commercial and legal relationship and consequently for the future course of the conversation. Therefore utterances are remembered by the parties or, in other words, (virtually) stored in the knowledge base that is common to the parties. Each utterance may be significant, also an utterance that deletes an entity, that removes a concept or that changes a property of a concept. Transactions only add facts and opinions to the knowledge base. Knowledge is never changed or deleted. Property values of course can change, but a changed value is then the 'latest value added'. Properties may even have multiple values simultaneously, as the different parties may differ in their opinion about the value.

One may delete a tuple from a database, but not command a trading partner to 'forget' events that have happened. A B2B knowledge base is not a data base, however it may be mapped to a database.

Business knowledge, as added to the common knowledge base of two business partners, is knowledge about concepts and entities.

4.6 Knowledge Base structure

The real world consists of things, objects or entities. We see these terms as synonyms and shall use 'entities' in the sequel. Entities can be tangible (e.g. a stone) or abstract (e.g. a number). Entities of the same type (entities that have something in common) are grouped and form a 'concept'. In section 5.3 the 'concept' concept is described in more detail. For now it suffices to regard a concept as the set of all (potential) entities that meet certain criteria. Criteria are defined on the entities' properties. All entities that have the same types of properties may be defined as belonging to the same concept.

The shared or common knowledge of the world is stored in the knowledge bases of the parties involved. A party's knowledge base may have been integrated in the database of his Enterprise Resource Planning system, but more often it is scattered over several information systems. So technically that "knowledge base" may be implemented in the respective private information systems or ERP-systems of the parties or (additionally) reside in the file cabinets in the parties' offices, where paper documents are being stored. In case the conversation is conducted by telephone or during a business lunch, part of the knowledge base may even only be stored in the personal memories of employees or agents.

Since the party's internal organization of his information is not relevant to the other party, the knowledge base may be regarded as one coherent (virtual) database. It is the party's responsibility to keep that database coherent in his communication with the outside world.

In many cases automated information systems have a rigid structure: their data models, rules and operations cannot easily be changed by users. Apart from the information system, business parties have a specific informational view on the business relation. Such a view is related to the companies' policy and working procedures. In order to do electronic business, both the data model of the information system and the informational view (as far they are relevant to the trading partner) must be represented in the knowledge base. Parties see the knowledge base "through" their information systems.

4.7 Definition of new concepts

As Saussure already concluded, the sign (symbol) that represents some real world concept has been defined by a linguistic community beforehand. If that is true, new concepts, represented by new signs cannot be introduced in a bilateral conversation, certainly not if the sign system is all they have. If the two partners would meet physically, they could use the real entities as examples, e.g. by pointing to them. Separated by a communication channel, that is not possible.

Yet it is necessary in a dynamic B2B relationship to be able to define new concepts, e.g. new products, services or conditions. If the introduction in the conversation of entirely new concepts is not possible, another mechanism must be found. The mechanism proposed in this thesis is inspired by the work of Wierzbicka.

Wierzbicka [6] raised the question why definitions of concepts in (ordinary) dictionaries are circular. She discovered that concepts in dictionaries are defined in terms of other concepts that indirectly and ultimately have the original concepts in their definition. She considered this fact as unsatisfactory and conjectured that there are some concepts that are axiomatic in human cultures. Axiomatic concepts cannot be defined to any deeper level but are intuitively understood, and accepted. To prove this theory she analysed a great number of languages, modern ones but also languages of native people that have been living isolated for centuries. Indeed she found 40 concepts that all cultures seem to have in common and that cannot be derived from other concepts.

She then designed a definition language to express all other concepts in terms of the 40 basic ones. As a cultural anthropologist she based the language on emotional value of concepts. Her definitions are based on the intersection of multiple supersets of the defined concept. For instance, a Mule would be defined as a Horse-Donkey.

Wierzbicka's 'bootstrap' approach seems attractive for use in an architecture where business partners are enabled to build forward on a standardized ontology for their bilateral collaboration, by introducing refined concepts based on existing ones. In chapter 5 this approach is elaborated on. In chapter 8 an extensible business ontology is proposed, that may play the role of the 40 fundamental concepts of Wierzbicka, but then in the context of a trade relation.

The mechanism that enables parties to introduce new concepts is based on refinement. New concepts are described as special subsets of already known concepts, by narrowing their properties.

As an illustration, suppose the concept of "vehicle" is already known in the common knowledge base of two communication partners. A Vehicle has a number of wheels. A "Bicycle" may then be defined as a Vehicle with the number of wheels being 2. This mechanism is described and elaborated in chapter 5. In chapter 9 a basic trade model or ontology is presented that can be used as the basis for refinement.

4.8 Knowledge base structure negotiation

Information systems (automated or not) are updated by means of transactions. A single decision or event in the real world usually brings about a multitude of changes. Each change is reported in a knowledge base update. The updates resulting from one event or decision have only meaning together. Such set of combined updates is regarded as a transaction. Transactions are ACID [7]: Atomic (if one part of the transaction fails, the entire transaction fails), Consistent (it leaves the knowledge base in a valid state), Isolated (it is not dependent of other transactions) and Durable (it is permanent and can only be reversed by means of a new transaction).

Transactions are atomic information update chunks. In B2B systems, part of the updates comes directly from the trading partners' information system. In fact each update of the partners' system is, if relevant for the trade relation, 'forwarded' to the other system. As described in previous sections, B2B communication is basically the same as synchronizing parts of information systems.

As the communication consists of transactions, the allowed transaction types must be defined and stored in the knowledge base. Allowed transactions are transactions that can be accepted by the information systems *and* that obey to the parties' policies. Both partners need to define the transactions they can perform and the transactions they can accept. If and when the transaction sets of both partners match, electronic trading can commence. In fact the negotiation about which transaction types may be added to the knowledge base is already part of the trading.

The knowledge base is populated with definitions of allowed transaction types and with the transactions themselves. Transaction definitions include the definition of common (mutually understood) concepts; the transactions populate the concepts with information on individual entities. Transactions can only be performed if their definitions have been accepted by both parties.

The negotiation process to determine the transactions that are allowed in a specific trade relation is elaborated in chapter 10.

Transactions represent decisions of trading parties and their observations of real events. In a transaction new instances of concepts may be defined, but also new concepts may be

introduced. Previously defined concepts may be extended with new property types, or properties of previously introduced instances may be added or changed. As stated in the previous section, concepts are defined as specialisations of more generic concepts; e.g. a Car is defined as a certain type of Vehicle. In a definition of a new concept the property types and/or the property value ranges of the more generic concept are limited in meaning, value range or cardinality.

Not only concepts and transactions need to be defined and agreed among the trading partners, the sequence in which transactions may occur in the course of the business process must be defined and agreed as well. A mechanism to do so is elaborated in chapter 7. The mechanism is based on the dependencies between transactions. Transactions may only be allowed under certain conditions. E.g. Delivery may only take place after an Order being placed. Conditions are defined on the contents of the knowledge base. Conditions may be negotiated much the same way as concept definitions are.

Parties can define transactions to be initiated by themselves or by the other party. In the former case the definition can be regarded as a *capability* of the party to provide certain information. In the latter case the definition is an information *requirement*. Of course capabilities and requirements do not need to match right away. In the conversation between the parties (through the B2B system) it must be determined whether a match can be reached.

Concept definitions and transactions are proposed or accepted through the respective information systems of the parties. The mapping to the local information systems is made at the time of proposal or acceptance of a transaction type. This mechanism is explained in chapter 10. The mechanism can be performed as a native function of the information system or by specialised middleware. If a party doesn't deploy a business information system at all, it can use a form based solution as described in chapter 13.

4.9 Intentions

Traditional information systems support the storage of "facts". Employees are authorized to enter the facts in the system. Employees are regarded as agents of the same entity: the organization they work for. In an inter-organizational business process the users are agents of different entities. The information an agent provides needs not to be recognized as a fact by the agent of his business partner. The information that is communicated consists of *opinions* rather than of facts. A supporting system needs to recognize this and support mechanisms to assess the level of agreement on the opinions that are stored.

Transactions, exchanged in a B2B process do not result from neutral observations. The party that initiates a transaction has some intention doing so. These intentions are relevant to the B2B relationship and are to be stored in the knowledge base as well. An intention can be a proposal, an acceptance, a commitment, a statement, a request, etc. A transaction is in fact a subjective utterance of one of the parties.

So a transaction has two aspects:

- It is a proposition, assigning properties or predicates to entities and concepts
- It has an intention and it changes the relationship between the communicating partners.

4.10 Conclusion

In this chapter a reference model was sketched for B2B communication that serves as a basis for the architecture and protocols to be defined in subsequent chapters. The reference model is based on general principles of human communication, but focused on more structured communication and applied to communication between organizations. The core of the reference model is a virtual knowledge base that is filled by the communicating partners. The knowledge base is implemented in the individual information systems of the partners. This implementation is a distributed implementation: the two information systems are kept in sync with regard to the common knowledge. The synchronization is implemented as a message exchange protocol. Each message contains one or more transactions. A transaction satisfies the ACID [7] rules.

The knowledge base contains knowledge about individual entities and about concepts. Information on an individual entity can only be added if the concept has been defined to which the entity belongs. New concepts may be defined as subsets of more generic concepts.

Knowledge is created when partners do observations on a universe of discourse: the part of the world that is relevant for their relationship. Knowledge can also be created when partners derive inferences from the observations and when they make decisions. Observations, inferences and decisions are added to the knowledge base with some intention. The intentions are part of the transactions.

In chapters 5 and 6 the structure of the knowledge base is designed. In chapter 7 the protocols are described to agree on new business processes. In chapter 8 a core business ontology is proposed that may serve as a basis for refinement in specific business contexts. Chapter 11 describes possible implementation of the protocols in business information systems and middleware.

References

[1]. Bussler: B2B Integration, concepts and architecture. Springer, 2003.
[2]. Glushko and McGrath: Document Engineering. MIT Press, 2005.
[3]. Ogden and Richards: The Meaning of Meaning: A Study of the Influence of Language Upon Thought and of the Science of Symbolism. Routledge & Kegan Paul, 1923.
[4]. Saussure: Course in General Linguistics (trans. Roy Harris). Duckworth, 1983.
[5]. Stamper et al.: Semiotic Methods for Enterprise Design and IT Applications. Proceedings of the 7th International Workshop on Organisational Semiotics, 2004.
[6]. Wierzbicka: Lexicography and Conceptual Analysis. Karoma Pub, 1984.
[7]. Gray: The Transaction Concept: Virtues and Limitations. Proceedings of the 7th International Conference on Very Large Databases, 1981.

5 Structure of a B2B knowledge base

Summary

Business people are human beings. Human beings communicate by using a natural language and natural languages possess some structure. In order to find and design the structure of business utterances that populate a B2B knowledge base, therefore first the structure of natural language is inspected. Business utterances need however not to possess the richness that natural languages usually possess.

In this chapter the structural aspects of natural language that is relevant to business communication is modelled. It is shown that such business utterances may be mapped to both ontology languages and information modelling languages.

Concepts are defined by genus and differences, stating the properties that distinguishes them from other concepts belonging to the genus. Properties are associations with other (defined) concepts. As the definitions must be machine interpretable, concepts are defined using their structured properties.

As business partners are independent, they are not merely exchanging facts, but do so with an intention. These intentions are modelled as speech acts.

It appears that seven utterance types exist: Definitions, Expansions, Restrictions, Instantiations, Observations, Perceptions and States. Definitions define concepts that may be contained in future utterances. Expensions and Restrictions modify definitions. Instantiations, Observations, Perceptions and States are used to exchange information on instances of the defined concepts.

5.1 Introduction

In the previous chapter a high-level reference model for dynamic B2B communication was introduced. The model includes a virtual knowledge base that is populated by the communicating parties, as the business partnership develops. The knowledge base is implemented in the information systems of the partners. Knowledge is synchronized among the systems by means of a communication protocol over a communication channel.

In this chapter the structure of the B2B knowledge base is further analysed and designed. An abstract language is developed to specify inter-organizational business processes. The language can be represented in a computer interpretable format to enable automatic control of the integrity and status of the business processes.

In this chapter we abstract from implementation aspects and focus on the structure of the information stored in the knowledge base and on the structure of the transactions updating this information. In chapter 8 it is shown how the structure may be expressed in

languages used to design systems and interfaces. Real implementation issues are tackled in chapter 11.

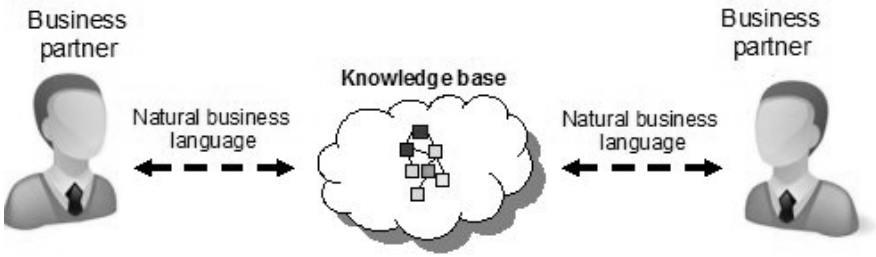


Figure 5.1 Conceptual reference model

Abstracting from implementation, business partners communicate with each other using natural business language, filling a knowledge base, as illustrated in figure 5.1. At this level the business concepts and the business process are agreed upon and shared information is created. The communication and the behaviour of the partners towards each other are constrained by a legal and economical context.

The knowledge base is filled with the observations, inferences and decisions of business people. Business people, like all humans, use natural language for communication of their observations, inferences and decisions. Therefore in section 5.2 natural language is analysed to find patterns that help structuring the knowledge base. Section 5.2 discusses the structure of sentences in natural language. The focus will be on business language. Business language is slightly more structured than natural language in general, as a result of the legal and economical context business is conducted in.

Ontology engineering claims to be able to capture and formalize knowledge, which is the main function of the knowledge base. Within ontology engineering a number of languages have been developed for knowledge representation. Knowledge representation languages relate (meta-)concepts to each other by means of logic. In section 5.3 is described how knowledge collected in the B2B Knowledge base can be modelled by ontology languages, such as Conceptual Graphs.

In section 5.4 FLBC is described, the Formal Language of Business Communication. This language was designed specifically to support structured business communication. It however lacks one important aspect: the capability to define new concepts on the fly.

In natural language reference is made to real world phenomena by means of terms or signs. In referring to the real world, distinction can be made between concepts, entities (instances of those concepts) and observations on entities. In static languages (and in rigid information systems) concepts (and sometimes even instances) are defined beforehand. Communication then only consists of observations and the instantiation of new entities. Dynamic languages allow for the definition of new concepts as well.

As explained in chapter 4 the business partnership to be supported by a B2B knowledge base is dynamic. Partners may define new concepts and negotiate new business process flows during their conversation. B2B knowledge is extensible. The knowledge base is not only used to store information on instances of predefined concepts, but also on the concepts themselves. New concepts may be created or existing concepts may change. In a traditional information system the structure of the information has been programmed beforehand while information on instances of pre-defined concepts is added during operations. In a B2B conversation, however, it must be possible to define new concepts "on the fly". Section 5.5 discusses definitions of concepts used in B2B and how such definitions are represented in the B2B knowledge base.

In section 5.6 a number of aspects of a business language are inspected, such as the properties of concepts, cardinality and speech acts. The main difference between B2B communication and private information systems is the perspective. Information systems are constructed from an objective, neutral perspective. In B2B communication the partners each have a subjective, intentional, perspective. Not only do they have a different viewpoint, they also have private intentions that may not always concur. This issue is treated in section 5.6. In that Section, Speech Act theory is introduced as a way to allow subjective intentions to be included in the communication. Speech Act theory leads to a few additional meta-concepts. Cardinality and properties as features are also elaborated in section 5.6.

In order to support discussions about concepts alongside discussions about instances, in a B2B knowledge base meta-data is stored alongside with data. The separation between meta-data and data, as it exists in information systems and exchange protocols, is made less strict to support the dynamics. In information systems, the meta-data is usually stored in programming code, system catalogues or schemata. In the B2B knowledge base that is designed in this thesis, meta-data is partly stored as data in the knowledge base. Concepts may be added to the knowledge base just as instances may be added. While in traditional systems newly added data is only validated against the meta-data schema, in a B2B knowledge base new data must also be validated against (meta-)data added previously. This feature is elaborated in section 5.7.

As mentioned, natural language sentences or utterances refer to concepts and entities. Both concepts and entities need to be defined and altered as a result of observations, inferences and decisions. This leads to five types of assertions, each with their own pattern: Definitions, Expansions, Restrictions, Instantiations and Observations. These types and patterns are described in section 5.8. The five patterns suffice if the purpose of the business partners would simply be to build a world model or ontology together. B2B communication however is geared towards the co-ordination of (economic) activities. The first three allow for introduction and manipulation of concepts, the latter two for introduction and manipulation of instances.

Activities must be undertaken in a certain sequence that needs to be agreed on among the partners. Two more patterns are defined to support definition and control of B2B processes. A sixth pattern, called "perception" is added to define the information on

instances that is needed to be exchanged in some business process step. This type of assertion is equivalent to the definition of Business Documents or message types in a traditional B2B environment. In such an environment, a process is defined as subsequent exchange of bundles of information (e.g., Quote, Order, Delivery, Invoice). These bundles may be defined as perceptions, as they are in fact views on the information in the common knowledge base.

A seventh pattern, “State”, defines the information on a concept instance that must be present in the knowledge base as a precondition to a process step. B2B process definition and control are elaborated in chapter 7.

In chapter 6 the basic structure of the utterances or transactions of the business partners that update the knowledge base is elaborated on, based on the findings in the previous sections. In section 6.2 utterances will be represented in a tabular structure. That structure is used to illustrate the validation and integrity rules that apply when the knowledge base is populated. The tabular structure also facilitates the wording of conditions that may be needed to enforce business processes. Conditions are elaborated on in section 6.3.

The utterance structure is presented in a meta-model in section 6.4.

When implemented, utterances must be represented in concrete structured languages such as XML. An important aspect of such languages is data typing and especially the hierarchy that exists in the stack from abstract types to bit level representation. This aspect is elaborated in section 6.5.

In an example in section 6.6 is shown that the described patterns suffice for a knowledge base in a typical B2B partnership. In chapter 12 more complex examples prove the completeness of the patterns for practical B2B contexts.

5.2 Natural business language

In this section natural language patterns are analysed. The B2B knowledge base is populated with assertions about observations, inferences and decisions of the business partners that participate in the business process. In order to be able to store those assertions in information systems as structured information, that information needs to be extracted from the assertions. Identifying natural language patterns is a first step towards that extraction. Extracted information is structured, so it may be automatically processed. In chapter 8 the patterns are mapped to modelling languages to enable the storage and processing of the information in computer systems.

Business is conducted by means of human conversation. Human conversation makes use of natural language. A natural language as meant in this thesis is a language that can be spoken and written by humans. Natural languages include English and Dutch, but also Esperanto, which is artificially constructed. We exclude body language and iconographic ‘languages’ that use standalone symbols and colours. Such languages are used e.g. in

traffic control to direct travellers or in marketing, to transfer feelings or to manipulate perceptual attitudes.

In this analysis only very basic language constructs are assessed. This is a design choice. Natural language can be very rich and may contain very complex structures. For the purpose of formalizing trade conversations in order to support them with computer systems a small subset of language constructs suffices. Business is not only being conducted in face-to-face meetings, but also by means of structured trade documents and (dedicated, customised) computer communication. Those communication channels are very limited in language complexity.

We make use of the findings of scholars who have attempted to analyse natural language in order to extract logical assertions, intentions and knowledge. Chomsky [1] has analysed language grammars to find logical patters such as Markov chains and Turing machines. Montague [2] mapped natural language to logical formulas and described how semantics and pragmatics may be added. Van Benthem [3] positions logic in natural language in the context of conversations and describes the dynamics of utterance sequences. These findings have laid the foundation of semantic language analysis by Fillmore [4] and of Conceptual Graphs [20]. Both are described below.

The analysis of natural language for extraction of knowledge and intentions has different objectives than this thesis has. Here the objective is to construct a structured language, based on natural language that allows business people to do business, supported by computer systems. It is a design exercise rather than an inductive study. It resembles the efforts of Zamenhof [5], who designed Esperanto, as a general purpose language. However, the language needed to do business (even including the specification of product technology and logistic control) may be less rich than a general purpose language that should also be capable of expressing emotions and allowing poetry.

Moreover, while Zamenhof constructed an entire vocabulary for Esperanto, here we present a mechanism to allow the business partners themselves to extend their vocabulary, which makes our task much easier. It is only necessary to define a bootstrap mechanism and/or a small basic extensible ontology. The latter is described in chapter 9.

Fillmore [4] developed a theory to analyse the syntactic structure of sentences by studying the combination of ‘deep cases’ or semantic roles (Agent, Theme, Beneficiary, Location or Instrument) which are required by a specific verb. For instance, the verb "give" in English requires an Agent, a Theme and a Beneficiary. E.g. "(Agent) Smith gave a (Theme) painting to the (Beneficiary) university”. Semantic roles can be mapped on syntactic roles, such as Subject and Object, but they are not the same.

An assertion about the real world has the structure of a (natural language) sentence or verb phrase. A sentence has a verb and may have several ‘slots’ that can be filled with noun phrases. As the focus of this analysis is on business language, the analysis is limited to complete sentences or verb phrases. Incomplete sentences, such as exclamations, are kept out of scope. It is assumed that a sentence at least contains a verb.

Sowa states that a verb represents an event, activity or process in the real world [6]. Bennett [7] distinguishes verbs that represent an action fulminating into some end point (e.g. 'Arrive') and verbs that refer to the action itself (e.g. 'Travel'). Another category of verbs represent a state (e.g. 'Like' as in 'George likes chocolate' – the state refers to the agent/subject). Here we shall treat these state representing verbs similar to the verbs that refer to an event or action. We interpret the verb as referring to the event when the state was established or (if that moment is not known or relevant) to the event when it was noticed.

The languages' grammar defines the functions or (syntactical) roles of the words or phrases that fill the slots in their relation to the verb and the subject. As a slot can be filled by another verb phrase, in general the structure of a sentence is recursive. Here such complex sentences are assumed to be decomposed in separate, non-recursive, sentences.

Slots represent syntactic roles that depend on the verb. Examples of syntactical roles are Object and Attributive Adjunct. Different languages have different mechanisms to syntactically indicate the syntactical roles. Some of those mechanisms are prepositions, cases and word ordering within sentences.

For example, in the sentence "The carrier delivered the package of Jansen to the house", the positions of the "The carrier" and "the package" indicate subject and object respectively and the prepositions "of" and "to" indicate adjuncts. In German the sentence would read: "Der Frachtführer liefert Jansens Paket zu dem Haus". Here the cases, indicated by articles (der, dem) and suffixes (Jansen**s**) indicate the syntactic roles.

Gruber introduced Thematic Roles (or Theta roles) of phrases as opposed to syntactic roles [8]. In the two sentences: "John wrote the paper" and "The paper was written by John", syntactically John and the paper are the Subject respectively. In both sentences however John plays the same semantic role: he is the "Agent" and the paper is in both sentences the "Theme" or the "Patient". In this analysis we are interested in semantics, rather than in syntax. Therefore the focus will be on Thematic (or semantic) roles of phrases and terms rather than on syntactic roles.

As stated by Fillmore [4] and Gruber [8] each Verb can be assigned a "Frame" with a number of "Slots", each to be filled by a semantic or thematic role. A number of initiatives have attempted to define the frames for the most common verbs. Verbnet [9] has defined frames and identified role slots for some 3500 verbs. The verbs are grouped according to a classification made by Levin [10]. Other initiatives to define semantic frames for verbs include Framenet [11] and Propbank [12]. Each initiative has a slightly different perspective and theoretical background. Attempts are made to combine the definitions from these multiple initiatives [13].

The major difference between the VerbNet "thematic" roles and the FrameNet "semantic" roles is that the thematic roles are generic and global with respect to language, while the semantic roles are local and specific only to their frame. Verbnet lists 22 thematic roles. Framenet defines considerably more semantic roles, but the definitions are

frame-specific and some roles are even named after the verb (e.g. Employer and Employee as roles for Employ). For the purpose of structuring a B2B knowledge base we prefer the Verbnet approach. In Verbnet generic roles are being reused across verbs and they may be specialized.

Verbnet identifies the following thematic roles [14]:

- **Actor**: used for some communication classes (e.g., Marry, Meet) when both arguments can be considered symmetrical
- **Agent**: deliberately performs the action (e.g. **Bill** ate his soup quietly)
- **Asset**: used for the Sum of Money Alternation, present in classes such as Build, Get and Obtain with `currency' as a selectional restriction
- **Attribute**: attribute of Patient/Theme refers to a quality of something that is being changed, as in "The price(attribute) of oil soared"
- **Beneficiary**: the entity for whose benefit the action occurs (e.g. I baked **Reggie** a cake)
- **Cause**: used mostly by classes involving Psychological Verbs and Verbs Involving the Body.
- **Destination**: end point of the motion, or direction towards which the motion is directed
- **Experiencer**: receives sensory or emotional input (e.g. The smell of lilies filled **Jennifer's** nostrils).
- **Goal**: what the action is directed towards (e.g. The caravan continued on **toward the distant oasis**).
- **Instrument**: used to carry out the action (e.g. Jamie cut the ribbon **with a pair of scissors**).
- **Location**: where the action occurs (e.g. Johnny and Linda played carelessly **in the park**).
- **Material**: start point of transformation
- **Natural Cause**: mindlessly performs the action (e.g. **An avalanche** destroyed the ancient temple).
- **Patient**: undergoes the action and has its state changed (e.g. The falling rocks crushed **the car**) (Sometimes used interchangeably with theme)
- **Predicate**: used for classes with a predicative complement.
- **Product**: end point of transformation
- **Recipient**: a special kind of goal associated with verbs expressing a change in ownership, possession. (e.g. I sent **John** the letter)
- **Source**: where the action originated (e.g. The rocket was launched **from Central Command**).
- **Stimulus**: events or objects that elicit some response from an experiencer
- **Theme**: undergoes the action but does not change its state (e.g. Bill kissed **Mary**). (Sometimes used interchangeably with patient)
- **Time**: the time at which the action occurs (e.g. The rocket was launched **yesterday**)
- **Topic**: the theme/topic of the conversation or transfer of message

Note that the terms to indicate thematic roles are deliberately different from syntactical roles that include “direct object” and “subject”.

It should be noted that the objective of the initiatives such as Verbnet, Framenet and Propbank is to automatically interpret natural language. Much effort is made to convert syntax into semantics. However, the objective in this thesis is not to automatically interpret the utterances of business people, but to structure their language in order to process the exchanged information automatically with 100% accuracy. This structuring should preserve the necessary flexibility and commercial freedom. So while the process for analysing natural language starts with a set of sentences, interpreting the syntax and then deriving the semantics, here we start with the semantics and construct a syntax to convey those semantics in a processable way.

Sowa [15] designed a structure with semantic roles, taking an ontological view. His aim is not to automatically interpret natural language but to use roles in designing an ontology.

He defines four fundamental roles:

Initiator: determines the direction of the process from the beginning
Goal: determines the direction of the process from the end
Resource: must be present at the beginning of the process, but does not actively control what happens
Essence: must be present at the end of the process

Depending on the frame or the type of verb or event, these roles may be specialized:

Type of frame	Initiator	Resource	Goal	Essence
Action	Agent, Effector	Instrument	Result, Recipient	Patient, Theme
Process	Agent, Origin	Matter	Result, Recipient	Patient, Theme
Transfer	Agent, Origin	Instrument, Medium	Experiencer, Recipient	Theme
Spatial	Origin	Path	Destination	Location
Temporal	Start	Duration	Completion	PointInTime
Ambient	Origin	Instrument, Matter	Result	Theme

Table 5.1 Thematic roles (from Sowa [15])

These roles may further be specialized, e.g. to the frame specific level of Framenet semantic frames. This approach brings structure in the roles and allows trading partners to define new context-specific roles if needed.

Verbs denote the relation between the entities that fulfil the thematic roles in a sentence. Some slots may be mandatory, other slots optional. An example, showing the core frame of “Deliver” is: An Agent delivers a Theme to a Beneficiary at a Location.

More roles may exist, such as Source, Time and Instrument (e.g. the vehicle used) or Speed (as in “We deliver your package overnight”). Roles may need to be specialized. For example the Instrument of a Delivery may be a vehicle, but also a Pallet. So the Instrument may need to be specialized into a Transport Means_ Instrument and a Transport Equipment_ Instrument.

Sowa's fundamental thematic roles may serve as a root for developing or negotiating a business ontology in a B2B knowledge base. The Verbnet thematic roles may be defined as specialisations of the fundamental thematic roles. In this thesis, however, these specialisations are not detailed.

Summarizing, a natural language sentence refers to an event, activity or process that involves the entities that fulfil thematic roles. The sentence defines the relations between those entities and the event, activity or process that is denoted by the verb. The entities filling the role slots are predicates or properties of the event. The thematic role names may serve as property terms.

For instance, in the sentence “Consignment 123 is delivered to Jones in Amsterdam” a delivery event is defined. Consignment 123, Jones and Amsterdam each play a role in that event and thus have a relation with the event. So the sentence defines a delivery event with Theme Consignment 123, Beneficiary Jones and Location Amsterdam.

5.3 Ontologies

Ontology engineering claims to be able to capture and formalize knowledge, which is the main function of a B2B knowledge base. Within ontology engineering a number of languages have been developed for knowledge representation. Knowledge representation languages relate concepts to each other by means of logic.

In this section the features of the most well-known ontology languages are assessed against the patterns found in business communication, keeping in mind the requirements of B2B systems, as defined in chapter 3.

Ontology engineering fills the gap between natural language and information systems. Ontologies are a formal description of the knowledge within a domain. Such a domain may be the context of a particular business relationship. Ontology languages therefore should be capable of defining the structure and the content of a B2B knowledge base. As an ontology (the result of such definition) is formal, mapping of the knowledge to an information system should be possible. Here we inspect ontology languages, such as RDF, OWL, SBVR and Conceptual graphs on their capability to describe the knowledge that is created by the sentences as described in the previous sections.

At the moment the main application area for ontology engineering is probably the Semantic Web [16]. The Semantic Web initiative attempts to make the World Wide Web (which is essentially a collection of interlinked multimedia documents) semantically searchable. The ontological perspective on the World Wide Web is somewhat different from the B2B perspective. In the B2B perspective, business partners communicate about real world phenomena (products, services, intentions). The semantic web has the perspective of annotating “resources”, existing on the web, so they become semantically meaningful. Resources are mainly documents or part thereof.

RDF

The Resource Description Framework (RDF, [17]) is a means to express properties of a resource and to assign values to these properties. Expressions or statements in RDF are represented as triples, consisting of a subject, a predicate (also called a property) and an object. An RDF statement denotes the relationship between two resources. Both concepts (classes) and entities (instantiations) are resources. The relations between subclasses and classes ('is a') and between entities and concepts ('is instantiation of') are expressed as RDF statements as well. RDF is expressed in XML.

OWL

The Ontology Web Language (OWL, [18]) can be regarded as an extension to RDF, adding cardinality, data types and arbitrary constraints. OWL is based on description logic. Cardinality expresses the multitude of objects that may be of some relation to a subject. E.g., in a language with cardinality it is possible to express that a car may only have one owner, but that it must have one or more components. It is also possible to express whether all cars must have an owner or that only some may have one. In logic this feature is called quantification. Although the main application area of OWL is the semantic web, the language is sufficiently rich to also be used for the description of business ontologies [19].

Conceptual Graphs

Sowa [20] has developed a language to describe the semantic structures of sentences, abstracting from their syntax. The language, Conceptual Graphs, has a graphical representation but also a textual one. Conceptual Graphs are a methodology to translate natural language assertions into an abstract language, grounded in Common Logic [21]. In the graphical representation, the Verb in a Verb Phrase is positioned in the centre of the graph, and the clauses of the phrase are positioned around it, with an indication of the thematic roles of those clauses.

Verbs and Nouns (or Noun Phrases) are represented by rectangles, and (thematic) Roles are represented by circles.

For example in the sentence "John is going to Boston by bus" four objects are identified: three noun-clauses (John, Boston and bus) play a role towards the verb (go). The sentence can be represented in a conceptual graph, in which the three nouns are connected to the verb by three properties of the verb. Objects in Conceptual Graphs can be typed. In this case John can be an instance of the type "Person" and Boston can be an instance of the type "City".

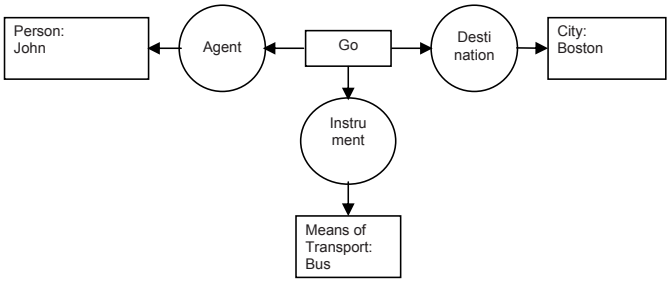


Figure 5.2 Conceptual Graph

Conceptual Graphs offer a perfect language to represent the sentence structure as introduced in section 5.2. Verb frames can be constructed by means of Conceptual Graphs by replacing a Noun by a 'Concept': a type of noun. A rectangle that contains a Concept rather than a particular instance then becomes a slot and the Conceptual Graph becomes a Verb frame. Additionally Conceptual Graphs introduce Typing of entities. Slots of verbs may only be filled by entities that belong to a certain type. In this case the Agent of Go must be a "Person" and the Instrument must be a "Means of Transport".

The Conceptual Graph can be represented as illustrated in figure 5.2 or as the string:

```
[Go]->
   (Agent)->[Person: John]
   (Destination)->[City: Boston]
   (Instrument)->[Means of Transport: Bus].
```

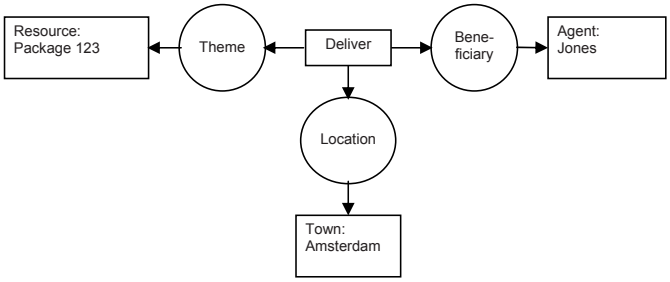


Figure 5.3 Verb frame

In figure 5.3 the (core) frame of the verb Deliver is represented as a conceptual graph. The Theme slot of the verb Deliver is filled by the Resource concept. In this case the entity Package 123 is an instance of the resource. The Location role is fulfilled by the Town concept and the Beneficiary role by the Agent concept.

Conceptual graphs offer a formal notation of natural language sentences as described in section 5.2. Rectangles represent nouns and verbs, circles thematic roles.

SBVR

Semantics of Business Vocabulary and Business Rules (SBVR) is a specification of OMG [22]. The specification defines a meta-model for vocabularies and business rule sets. The meta-model maps to a representation of business vocabularies in structured English. It is grounded in predicate logic. As such SBVR can, just like Conceptual Graphs, be regarded as another way to represent facts in a business context. SBVR also supports business rules. Business rules are different from facts in that deontics are involved. For each statement is indicated whether the statement is necessarily true (a fact) or *should* be true (a business rule).

The ontology languages described above offer the following concepts and features as ingredient for the structuring of a B2B knowledge base:

- Core statement as triple Subject-Property-Object
  Each utterance to be added to a B2B knowledge base has a core, consisting of a Subject, a Property and an Object. Subjects and Objects may be concepts (classes) or entities (instances).
- Cardinality
  It must be possible to specify cardinality or quantification.
- Data typing
  Ultimately, the concepts and entities must be represented by signs in computers and over communication channels. So a data typing mechanism is needed.
- Constraints
  Not all constraints that apply to an ontology can be expressed in core statements and cardinality. Arbitrary additional constraints may apply to utterances.
- Verbs and Roles
  Property terms are in fact derived from the names of verbs and names of (thematic) roles.
- Deontics
  B2B utterances are not objective facts, but include intentions of business partners, just like the deontics that are part of business rules.

Together with the Verbs and the Thematic Roles in section 6.2 these concepts and features are used as ingredients for the structure of the B2B knowledge base. Ontology engineering has introduced the concepts and features in order to capture human knowledge. The requirements, stated in chapter 3 include the capturing of knowledge, expressed by business people during a business conversation into a knowledge base that is implemented in their respective information systems. Ontology engineering solves part of that problem.

An important feature that is still missing is a mechanism by which new concepts may be introduced. Such a mechanism is described in section 5.5.

Other concepts and features that need further inspection are quantification, deontics and the reconciliation of the ontological Property concept with Thematic Roles in natural language. These are analysed in section 5.6.

5.4 FLBC

A language that claims to fulfil the requirements of a structured business language was developed by Moore and Kimbrough. They developed a Formal Language for Business Communication (FLBC) [23]. FLBC structures natural language (including thematic roles) in first order predicate logic and adds Speech Acts (see section 5.6). The authors claim that the language can be used in dynamic and flexible B2B relationships.

The core of a clause in FLBC is a proposition in predicate logic. For instance “Delivery city is Groningen”. The propositions “P” are packed in speech acts “F(P)”, such as “Request(Delivery city is Groningen)”. The speech acts can be nested and combined, e.g. Inform(Request(Delivery city is Groningen) and Commit(Payment is made through ING-Bank)).

FLBC is a very flexible and precise language. FLBC allows capturing business communication in a processable way. The main strength of FLBC is the identification within B2B utterances of propositional content *and* intention or illocutionary force.

FLBC has however no internal mechanism to define new concepts. The FLBC vocabulary is extensible, but extensions must be defined outside the language.

Weigand et al [24] have extended FLBC and represented it in proper XML. They call the resulting language XLBC (Extensible Language for Business Communication). The main extension is the reference from the language to a controlled vocabulary that defines the concepts and terms used. XLBC also does not have an internal mechanism to extend that vocabulary. XLBC separates semantics from (XML-)syntax.

No mapping is provided from XLBC or FLBC to legacy EDI and XML messages. Moore [25] has shown that the function of most EDI messages can be fulfilled by FLBC. The structures of FLBC and XLBC are however different from traditional electronic messages, mainly because the speech act verbs and the propositional logic form explicit parts of the language grammar.

In constructing the structure of a B2B knowledge base we build forward on the structures as proposed by Moore et al. In chapter 8 the structure is mapped to ‘legacy’ XML structures. We however do not use the nesting of propositions. Nesting would make the language unnecessarily complex.

5.5 Definitions

As mentioned in section 5.3 in a dynamic B2B knowledge base a mechanism is needed to introduce new concepts, in order to fulfil requirement #5 in chapter 3. Such a mechanism is described in this section.

In section 5.2 natural language assertions on individual entities (Package 123, Amsterdam) were analysed. Observations and decisions in business relationships however may also concern sets of entities. Such sets may be exclusively defined by their members (the extension of the set), but its membership may also depend on predefined criteria on the entities and the membership may change over time. In other words, the set may be defined by a comprehension scheme (in a logical formula: $\{x \epsilon A | F(x)\}$, where $F(x)$ is the set of criteria that determines if some instance x is a member of concept A). In the latter case we speak about a type, class or concept. A concept is defined here as the (potential) set of entities that meet certain criteria. In this section the language patterns to manipulate concepts, membership criteria and membership ("instantiation") are inspected.

Observations refer to concepts, entities and properties of those entities. Concepts and properties are referred to by means of terms or signs (see section 4.2). Before a term can be used in an observation, there must be an agreement among the partners what concept or property the term represents. Definitions enable the partners to relate terms and real world phenomena to each other. Definitions make it possible to introduce a new concept, based on a concept the partners agreed upon previously, and assign a term to the new concept.

Wieringa [26] states that concepts may be defined by genus and difference. He also introduces operational definitions (a procedure that can be followed to determine whether a term has been correctly assigned to an entity), but definition by genus and difference fits best for knowledge base structures. A concept inherits its definition from a more general, wider concept (the genus) and adds a set of constraints to the properties of the general concept to differentiate itself from other specializations or subtypes (and consequently subsets) of the more general concept. Entities that are instances of the newly defined concept are also instances of the more general concept. The 'differences' are defined by constraining the properties. Properties can only be constrained if they have been defined. In order to define a specialisation therefore the properties that are to be constrained must have been defined on the level of the more general concept. As the instances of the specialised concept (that have the constrained properties) also belong to the more general concept, at least some instances of the more general concept have these properties.

New concepts are defined based on existing concepts, so the basic semantics of concepts and properties are already known. Those semantics are refined by further constraining the properties of the concept members, in other words, the 'membership' criteria are made

stricter. In most cases this is established by stating the ‘rigid’ properties of the new concept instantiations: the properties all instances must possess.

A concept is defined by means of stating the criteria entities must fulfil to be a member of the concept. A concept as a set can have a potential extension (all possible entities that in some point in time may fulfil the criteria) and an actual extension (the entities that have been identified in the knowledge base as fulfilling the criteria now). At the time of definition of the concept the actual extension is empty: the concept may only be instantiated after it has been defined. It is however possible that the more generic concept already was instantiated. Existing instances may appear to belong to a newly defined concept, based on their properties.

As an illustration, we take again the example of the “Vehicle” that has been previously defined having a (variable) number of wheels. A “Bicycle” may then be defined as a Vehicle with the number of wheels being 2. Expressed in a sentence: “A bicycle is a vehicle with 2 wheels”. Note that in database representations the number of wheels of a bicycle is usually not an attribute: as all bicycles have 2 wheels, storing the number of wheels for each bicycle would be redundant. The name ‘Bicycle’ already implies that the number of wheels is 2. For bicycles the numbers of wheels being 2 is a rigid property.

Another example: a “tomato”, according to the Pocket Oxford Dictionary, is a “glossy, red or yellow, pulpy edible fruit”. To define the “tomato” concept as a subset of the “fruit” concept, the fruit concept needs to have the property types “appearance”, “colour”, “consistency” and “edibility”. The tomato concept would then consist of all fruits with the property values “appearance=glossy, colour=red|yellow, consistency=pulpy and edibility=true”. This means that “fruit” needs to possess the property types that are being narrowed in the definition of “tomato”. The Oxford Dictionary at least implicitly assumes that the properties are meaningful in a ‘fruit’ context (in practice, when defining an ontology, one must frequently add property types to higher level concepts that initially were not allocated to them).

In order to define a tomato in the knowledge base, the concept ‘Fruit’ needs to be defined first.

[#0] Define Concept with Attribute

[#1] Define Fruit, based on #0, with Is Vegetable_ Attribute = ‘True’
[#2] Expand Fruit, based on #0, with Appearance_ Attribute
[#3] Expand Fruit, based on #0, with Colour_ Attribute
[#4] Expand Fruit, based on #0, with Consistency_ Attribute
[#5] Expand Fruit, based on #0, with Edibility_ Attribute

[#6] Define Tomato, based on #2, with Appearance_ Attribute = ‘Glossy’
[#7] Define Tomato, based on #3, with Colour_ Attribute = {red, yellow}
[#8] Define Tomato, based on #4, with Consistency_ Attribute = ‘Pulpy’
[#9] Define Tomato, based on #5, with Edibility_ Attribute = ‘True’

The Definition pattern can be described as:

A [new concept name] is a [existing concept name] with [property1 = concept1|valuespace1, property2 = concept2|valuespace2, …].

The properties mentioned in a definition are the properties of the instances of the defined concept that are essential to categorize those instances as belonging to that concept. Often those properties are fixed values (such as the number of wheels of a bicycle) or reduced value spaces (such as the colour of a tomato). Properties that are not unique to the instances of the defined concept, that may belong to instances of other concepts as well, are not mentioned in the definition. A bicycle may have a weight (that varies from bicycle to bicycle), just like many other types of objects. The weight is not part of the bicycle's definition.

Restriction of properties

Business sentences are assertions about entities and/or concepts. An assertion assigns properties or predicates to an entity or to (instances of) a concept. As explained in section 5.2, the property names can be derived from the verb and the thematic roles in the sentence. For B2B communication verb names, property terms and entity identifiers (that fill the slots) cannot be chosen freely. For computers, terms are simply arbitrary strings of characters. The semantics, allocated to the term must be defined explicitly.

Properties are associations of entities with other entities through events, see section 5.2. Other entities may be objects or values. In section 6.5 data type or value systems are elaborated upon. Values may be regarded as entities. Properties of Concepts are associations with other concepts (which may have an extension size of 1) or with value spaces.

The property terms mentioned in the definition must already have been defined as being property terms of the existing concept. The slots (target concepts) of the properties of the newly defined concept must belong to the target concepts of the properties of the existing concept.

Except target concepts that are narrowed, also the property terms (role and verb names) may be narrowed. E.g. if the verb of the existing property is "Pay", the verb of the new property may be "Cash_ Pay".

In Conceptual Graphs a mechanism has been defined named "Restriction". Exactly this mechanism is used for the definition of new concepts. In figure 5.4 a slightly adapted example is shown from the Conceptual Graphs standard [20]. It is assumed in this example that "Chasing" is a special kind of "Hunting". In this case a Cat might be defined as a Predator that Chases Mice (while predators in general hunt animals).

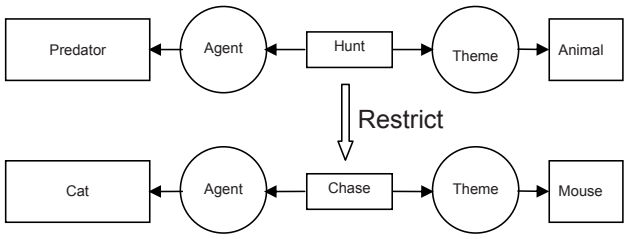


Figure 5.4 CG restriction

The properties in the definition must have been defined to be properties of the existing concept. Later in this section conventions are proposed for the naming of concepts. Also a mechanism is presented to specialize properties. The property "colour" may be a specialization of the property "look", which may be a specialization of the property "feature". Specialization of properties as concepts get more specific, avoids the need to define hundreds of property types for the higher level (more generic) concepts. In fact it is feasible to depart from a small set of thematic role names.

Note that the properties mentioned are in fact not the properties of the concept (which is a set), but they are properties of the concept instances, projected on the concept. We say that a tomato 'is' red or yellow, and we mean that the instances (members) of the tomato concept (set) are red or yellow.

Before entities are defined, the concepts they belong to should be defined first. Entities are instantiated from the concepts. Only then information systems may handle information on entities in an organized way.

The mechanism to define entities can be illustrated to compare it to the definition of concepts in a traditional information system. In such traditional systems, the system designer has pre-defined the concepts and has assigned property types to each concept. A system user then may instantiate the concept by giving values to the (mandatory) properties of entities. Later a user may update property values of entities.

Not only concepts need to be defined, also properties need definitions. As has been shown in section 5.2 property terms are derived from verbs that represent events, activities or processes. An event is an entity and its properties are the concepts that fill the semantic slots of the verb. A new property inherits the frame from the property it is based on, while the number of slots and the concepts that may fill the slots are further constrained.

Naming

When a new concept is defined it is given a name. The name in fact is a code or sign that represents the definition, or better, represents the set of criteria that must be met by

instances in order to belong to the concept. The name is also a (special) property of the concept. As the criteria themselves also need to be defined, the concept name is in fact a derived attribute, dependent of the set of criteria.

Concept names are here considered to be name spaces that can be filled with the names of sub concepts. Concepts inherit the properties of the concept they are based on, including the name. This mechanism is similar to that of a family name that is inherited by family members. Namespaces however can have arbitrary levels, whereas in many cultures personal names have only two (Family name and Christian name). In some cultures also family names have arbitrary levels (e.g. "Ayaan, daughter of Hirsi, son of Magan, son of Isse, son of Guleid, son of Ali, son of Wai'ays, son of Muhammad, son of Ali, son of Umar, son of Mahamud" is the way family naming is expressed in Somalia). Within the namespace or scope of a “Family name”, for the B2B knowledge base we demand that “Christian names” are unique (which is not always the case in human families). When a defined concept is further specialized, the Christian name of the concept serves, in combination with its Family name, as the family name of its specializations.

As an example: “Fruit” may have been defined as specialization of the more generic concept “Food”. The term “Fruit” then must be unique within the “Food” namespace (in another namespace however, the term “Fruit” may be used for something entirely different, e.g. the result of some economic venture). An “Apple” then can be defined within the ”Fruit” namespace. “Jonathan” can be a further specialization of “Apple”. One could name the Jonathan apple race in a more globally unique way by also mentioning its namespaces: Jonathan_ Apple_ Fruit_ Food.

Concepts may not be assigned other properties than the ones they inherit: the set of properties may only be restricted. If a property value is outside the range that was defined for the original concept, the particular instance cannot be a member of that original concept. This would be in violation to the sub-setting rule. This principle seems in contradiction with the Object Oriented inheritance mechanism. In fact it is not. In Object Oriented models properties of lower level classes are usually not shown on a higher level, but some instances of the higher level class do possess the properties (namely instances of the lower level class). The difference between Object Oriented inheritance and the Based On mechanism in B2B knowledge bases is therefore only a representational issue.

By considering a concept name as a name space, no exception on the (semantic) restriction rule for properties is based on relations needs to be made for the name. The inherited name space is further restricted with a term that distinguishes the newly defined concept from other concepts that may be defined based on the existing concept. This way the name can be considered as a ‘normal’ property.

So the bicycle mentioned before will be named “Bicycle_ Vehicle”, and the tomato will be named “Tomato_ Fruit”. The former is a restriction of the namespace “Vehicle”, the latter a restriction of the namespace “Fruit”.

In many of the examples in this thesis, for brevity and clarity not the fully qualified names are shown (walking along the branches of the based-on tree all the way to the top term “Concept”). Often only the last term is mentioned, sometimes the last term and its direct parent. So instead of “Jonathan_ Apple_ Fruit_ Food_ Concept” we say “Jonathan” or “Jonathan_ Apple”.

The name of a concept serves as the namespace of the names of concepts that are based on it and of the names of the entities that are instances of the concept.

A practical implementation of this mechanism is described in ISO 15000-5 or CCTS (Core Component Technical Specification) [27]. In this specification the names of concepts (object class terms) and properties (property terms) are separated from their namespace by means of a special separator (underscore-space). Names are called ‘qualifiers’ in CCTS. CCTS does not explicitly support (yet) the fact that a property term should consist of a verb name and one or two role names.

A property relation consists of three parts: an object class term, a property term and a representation term. All three consist of strings of qualifiers that denotes the based on relation tree. This structure is also described in ISO 11179 [28].

For example, a Chocolate_ Food_ Product. Urban_ Reseller_ Customer. Franchise_ Retail_ Party is a property of the concept Chocolate_ Food_ Product. That concept is based on the concept Food_ Product, which is based on the more generic concept Product. In the same way the concept Franchise_ Retail_ Party is based on Retail_ Party which is based on the high level concept Party. In the same way, the property Urban_ Reseller_ Customer is based on the property Reseller_ Customer, which is defined as a property of either Food_ Product or Product. The property is based on the higher level Customer property (of Product). See figure 5.5.

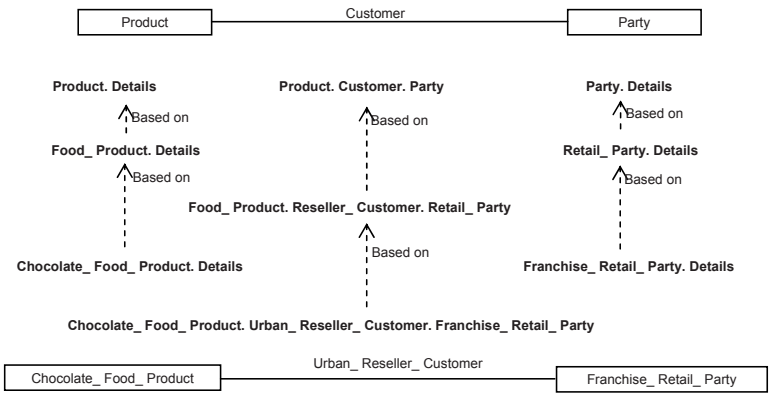


Figure 5.5

Expansion

We make a distinction between properties that define the concept and other properties. Properties that are not needed for the definition (e.g. properties that need not be present for all instances or properties that have the same value space as the more general concept has) will be defined by means of an "expansion" instead of a "definition".

Definitions distinguish concepts by properties that make them different from other concepts. Usually entities have (many) more properties than the properties that define them to belong to a concept. Communication should also be possible about those non-discriminating properties. These properties are assigned to (previously defined) concepts by means of expansions.

When the set of properties of an existing concept is extended, we speak of an expansion. The new properties must also be or have been included in the property set of the concept on which the existing concept based (etc.). The extension of the concept is not changed by this assertion type.

The format of an expansion is:

A [concept name] has [property1 = concept1|valuespace1, property2 = concept2|valuespace2, …].

Example:
A car has colour = colour code.

Restriction

Instead of expanded, a concept also may be restricted. For all future utterances that are based on the definition, then the restrictions apply. Restrictions may for instance limit the cardinality, the allowed uttering party, the allowed stereotype or the allowed intention of the utterances that are based on it. By means of restriction a concept may be locked by not allowing any more definitions based on it. A Restriction may also lock further instantiation or even observation.

The format of a restriction is:
The [meta-attribute] of concept [concept name] is restricted to be [value space]

As concepts are defined, based on more generic concepts and as this mechanism is recursive, all 'based-on' levels are present in the knowledge base, including the highest level concept (named "Concept"), which represents the set of all possible entities and concepts (except itself of course). The mechanism to define new concepts allows trading partners to define any new concept based on very high level concepts, even on that "Concept" concept. In a business relation however, especially when automated

information systems are used, one needs to “freeze” parts of the already agreed (M1) model.

It must be possible to freeze part of the agreed model or ontology to a certain level (from the top down). In the construction industry it may for example be allowed to define “gas concrete brick” based on the already known “brick”, but to define a “Grasshopper” based on the (rather high level) concept “Living Creature” would be out of scope. Construction industry information systems would not be able to handle grasshopper related messages. In that case the “Living Creature” concept is to be locked: no new concepts may directly be based on it.

Note that in traditional automated B2B (EDI or XML) systems, all (M1) metadata is defined at design time, and therefore frozen at run time. Metadata is designed during design time and then rigidly built in information systems and interfaces. To allow the freedom to over and over again “start from scratch” building an ontology in an (operational) B2B knowledge base is the other end of the scale and often undesirable, both from a technology and from a business point of view. Business relations need a relatively stable environment, although there must exist some room for improvement, innovation and creativity.

To avoid unwanted freedom some kind of locking mechanism is needed to stop definition of concepts based on too high level super concepts. Locking can be illustrated with the following example. Suppose some business supports two payment methods: bank transfers and cash payments, but not credit card payments. The initial knowledge base has a content as pictured in figure 5.6.

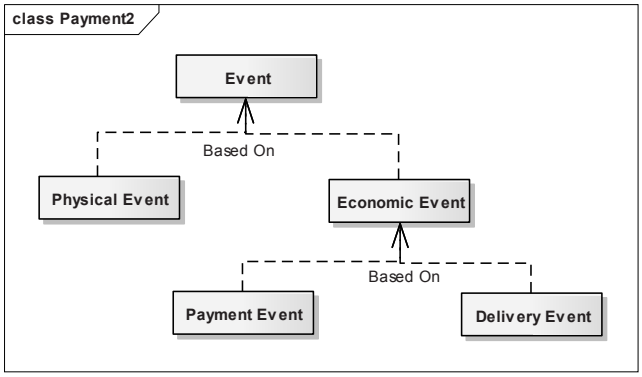


Figure 5.6 Initial KB content

In its offers to its business partners the business specializes the Payment Event with Bank Transfers and Cash payments. To prevent some business partner to add more payment event types, such as Credit Card payment, the Payment Event is restricted by specifying

the ‘Allowed Stereotype’ to be Instantiation and Observation, but not Definition (figure 5.7).

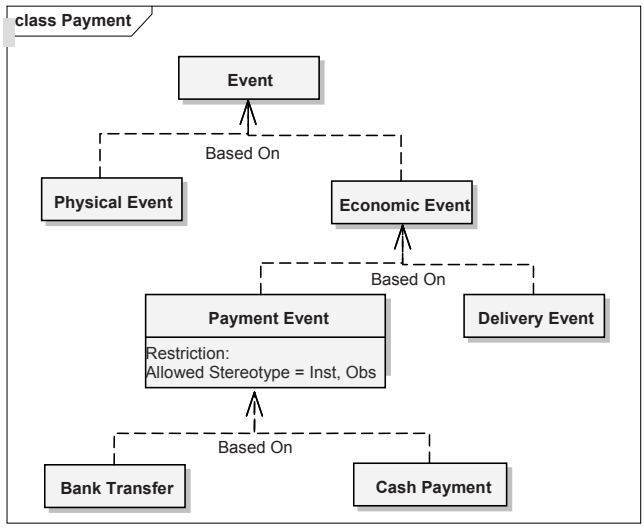


Figure 5.7 Locking

Note that exhaustive subtyping is a known concept, supported by languages such as ORM.

Summary

Summarizing, in a B2B knowledge base entities are introduced and defined as instances of previously defined concepts. Concepts are defined as subtypes or specializations of more general concepts. Definitions are represented as constraints on the properties of the more general concept. Each property must have been defined on the higher level concept, and is limited or narrowed (in quantification or specialization of target concepts or entities).

Concepts are identified by names that form namespaces for their sub-concepts and instances. Entities may have multiple identification schemes, each consisting of multiple identifying properties. Properties of entities may be rigid and thus may not change during the entity’s life cycle.

5.6 B2B Knowledge base features

In this section some needed features of a B2B knowledge base, such as Identification of entities, Quantification and the Property concept are further inspected.

Properties

In section 5.2 thematic roles were introduced to qualify the relation between noun phrases that represent concepts and entities, and the event that is represented by a Verb. In section 5.3 is shown that the core of assertions in ontology languages relates concepts and entities directly to each other, 'bypassing' the event, by means of a predicate or property term.

So it is needed to 'shortcut' the verb and to denote relations between the entities directly. Entities have relations with other entities through events. The relations are meaningful for the knowledge about the entities. Entities are even known through their relations, as the characteristics that identify them are also relations (e.g. a fingerprint or a license plate) and concepts are defined by their properties as was shown in section 5.5. A language should be capable of capturing relations between entities mutually and not only between entities and events.

Let us take as an example the delivery of a specific consignment (Consignment 123). We might for instance be interested in the delivery date of Consignment 123, instead of in the delivery event itself. We would like to express the delivery date as a predicate of Consignment 123, and 'shortcut' the event that causes the predicate to be true. The verb name and the names of the thematic roles then together define the relation between the entities that fill the slots in the frame, as a result of the event, action or process that is defined by the sentence. July 10 is the Deliver Time (or Delivery Date) of Theme Consignment 123.

The requirement to express relations as predicates of entities has an important practical cause. It is derived directly from the Implementability requirement in section 3.10. A B2B language must be representable by modelling languages, ontology languages, database schemes and message syntax structures. Most such languages do not capture sentences or event structures, but relations or associations between entities.

So a relation between two entities is denoted by three terms: a verb (that defines the event) and the role names of the two semantic slots of the related entities. Together they define the relation between the two entities. If one of those entities is the event itself, only two terms are needed: the verb and the role name of the other entity (even the verb then is redundant as it is implied by the event name).

The role of both related entities is meaningful, as the following example shows. In the sentence "Package123 is delivered by Smith to Jones in Amsterdam" we may be interested in the relation between Smith and Package123. Package123 is the Theme, the verb is "Deliver". But without the role of Smith (Agent) we do not know whether Smith received or delivered the package.

The four fundamental roles Initiator, Goal, Resource and Essence can be specialised if that is useful to clarify the semantics or to distinguish between different roles of the same fundamental type.

In practice, when modelling an information system, property terms are often chosen rather informally. In some modelling languages (such as UML) practice is to define properties as roles (nouns) of entities played in relation to each other. The verb is then implied (as in the "Carrier" as property of a "Consignment"). In other modelling languages (such as ORM) verbs are used to define relations (as in "Carrier transports Consignment") and the roles are implied. The terms (nouns or verbs) are often chosen intuitively and arbitrary. In a system in which semantics need to be negotiated in an automatic fashion it is required to bring more structure in the selection and composition of terms.

In the sequel we model the relation between two entities as a property of one of the entities. The property term then exists of the verb and the semantic slot names of the 'source' and the 'target' entity. Represented as property, the relation is asymmetrical. It has a direction (from source to target). The relation needs to be specified twice if the other direction is meaningful as well.

For instance, in the sentence "The package was delivered in Amsterdam", Amsterdam is the "Theme-Deliver-Location" (Role-Verb-Role) of the package, while the package is the "Location-Deliver-Theme" of Amsterdam. Amsterdam is the "Deliver-Location" of the Delivery event and the package is the "Deliver-Theme" of that event.

In table 5.2 some example verbs are listed, cross referencing verb names, event names and thematic roles. The fundamental role names, as proposed by Sowa have been specialised in more meaningful names.

Verb	Event	Thematic Role
Deliver	Delivery	Theme_Essence
		Beneficiary_Goal
		Time_Goal
		Agent_Initiator
		Location_Goal
Buy	Ownership transfer	Patient_Essence
		Beneficiary_Goal
		Time_Resource
		Agent_Initiator
Have	Obtainment	Goal
		Theme_Essence
		Agent_Initiator
		Time_Resource
		Location_Resource

Table 5.2 Example Verbs, Events and Roles

Verbnet identifies 270 verb classes. Verbs united in a class are semantically similar and share a core frame. Each class is named after a stereotypical verb. Specific verbs are specializations of that stereotype. For instance the "Put" class has as members arrange, immerse, implant, lodge, mount, position, situate, sling, station and superimpose. In chapter 9 a few classes will be selected for use in business communication.

In order to be able to specialise verb sentences, a hierarchy of verbs is created. At the top of the hierarchy the generic verb “Do” is situated. Immediately under “Do” the verbs “Be” and “Have” are situated. “Be” is used to indicate the genus in definitions (“A car IS a vehicle that....”). “Have” is used for possessive relations or properties. These three verbs form the top of an extensible tree with all other verbs.

Summarizing: a B2B knowledge base is filled by means of sentences that represent observations, inferences and decisions of business partners. The information in a sentence can be represented as a property relation between an event, activity or process (as indicated by the verb) and each identified entity. The entities fill slots in a frame that is associated with the verb in the sentence. The verb may be shortcut and the sentence may define property relations between the entities directly. Language analysis, aimed at automated interpretation of natural language, thus helps to define structures suited for a business language that conveys information that can be stored in database systems.

Names of events can be derived by objectification of the Verb that defines the event. Property terms can be derived from the names of the thematic roles that fit in the slots of the verb frame.

The sentence structure analysed in this section was applied to individual entities. In subsequent sections this core structure will be also applied to concepts and will be extended with quantifiers (cardinality), naming conventions and identifiers. Also the way in which new concepts and entities may be introduced and defined will be elaborated. In chapter 8 this structure is mapped to data modelling and exchange languages.

Identification

Instances are individually identified within the scope of the concepts they belong to. The concept name can therefore be regarded as the namespace of the identification scheme(s) of the instances.

Instance identification can be distributed over multiple properties. An address, for instance, may be identified by its country, city, street and building number. Those properties should be labelled as being identifying. Sometimes multiple sets of identification schemes exist and are defined in parallel. As example: a Car may be identified by its License plate number (= the combination of Issuing Country and number) or by its Chassis number (= the combination of Manufacturer, Type and Number).

In natural language the phrase “Is identified by” or “Is identified by the combination of” can be used, as in “An address is identified by the combination of Country, City, Street and Building Number”.

Within the scope of the business conversation, an instance can be referred to by means of an arbitrary name that is unique within that conversation. This mechanism resembles the

way in which in SQL reference is made to instances, by assigning a (temporary) arbitrary identifier to them.

We take as an example an Address, with three properties: Postcode, House Number and Floor. Assume that the first two properties define an Address instance. Suppose it is needed to give a value to the Floor property of some Address instance in a conversation. One may then refer to the address as "This". "This" is a nick name that refers to the instance within the scope of the conversation:

This_ Address. Post Code = 8017KJ
This_ Address. House Number = 208
This_ Address. Floor = 5

The first two utterances identify the instance; the third one assigns a new value to a property Floor. Instead of "This", any arbitrary string could have been used, such as "123" or "XYZ".

By specifying the identifying properties it is sure (by definition) that the resulting entity is uniquely addressed or better, that the extension of the resulting set is 1. Any other property restriction (e.g. all Addresses with house numbers 208 that are positioned on the 5th floor) may lead to multiple entities. In that case a new concept may be defined with its own identification scheme.

During a conversation instances may also be referred to by an arbitrary name or number, within the namespace of the concept they belong to. A group of observations of an instance may also have a name assigned. The same instance may even get multiple names as the conversation develops. Names are not determining the identity of the instance. Identification schemes do. At instance level, naming is merely a (temporary) referencing tool.

Quantification

Constraints that define concepts apply in principle to *all* instances of the concept. Constraints which define new concepts via specialization of existing concepts, may however only apply to *some* of the instances of the original concept. This "quantification" is part of the definition. For instance, all Rivets have a head, but only Blind Rivets have a Mandrel. So the property Head is always present for rivets, while only some rivets possess the property Mandrel. All Bind Rivets however have a Mandrel.

In natural sentence languages quantification can be indicated by the words "Each" or "Some", as in "Each rivet has a Head" and "Some rivets have a Mandrel".

When defining a new concept based on an existing one the properties of the existing properties are constrained. As said before, no properties are introduced for the new concept that do not apply to the existing concept. The instances of the new concept are also instances of the concept on which the new concept is based. So at least some of the

instances of the existing concept possess the properties of the new concept. This must have been defined for the existing concept (or must be defined at the same time as the new concept is defined). This is also the case for all higher level concepts the concept under definition is based on. Note however that properties may be more generic at the level of the existing concept. For instance, a “Fastener” may have a property named “Physical Measurement”, while a “Rivet_ Fastener” may have “Length_ Physical Measurement” as one of its properties.

Another quantification is the multiplication of a property. Some properties (e.g. an identifier) may only occur once, while other properties (e.g. the parts of a car) may occur multiple times. In natural language this quantification is stated as “Zero or One”, “One or More”, “Exactly two”, etc. For instance: “A Car has One or More Parts”, “A Bicycle has Exactly two Wheels”.

Both Conceptual Graphs and SBVR support quantifiers, such as “Every” (universal quantifier, ∀) or “At least one” (existential quantifier, ∃). SBVR also supports more sophisticated quantifiers, such as “Exactly 3”. Quantifiers translate into cardinality when ontologies are being represented in modelling or database languages.

Speech acts

Deontics in Business Rules imply the existence of some body (government or corporate board) that has the authority to issue laws and rules. B2B relationships are symmetric. Business partners do not have the authority to enforce rules upon each other. They may instead negotiate mutually agreed rules, based on their interests. Mutual agreement is expressed in commitments of the trading partners to comply with the agreed rules.

B2B deontics therefore are more subtle than deontics issued by corporate management. The SBVR deontic vocabulary which includes phrases such as “Is necessarily” and “Should” is to be extended with personal ‘deontics’ or intentions, such as “Wish”, “Commit”, “Declare”. In fact for B2B relations deontics need to be extended to speech acts. The partners need to be able to express their opinions, wishes, requirements, etc., which are subject to negotiation in the conversation.

The structure to add deontics is very similar to the structure needed to add speech acts to facts. In a deontic sentence is stated: “It is [necessary|obligated] that [fact]” (e.g.: “It is obligated that each order is confirmed”). A Speech act has the structure “I [speech act verb] that [fact]” (e.g. “I wish that you confirm my orders”).

The partners in a business communication relationship are independent peers. Different from employees who operate the information system of their company, business partners are not hierarchical subordinate to the same boss. They have different interests and different viewpoints. For utterances that are exchanged within such a relationship, it is important to know who uttered the assertion and why (s)he did so. The why however often is not be revealed. What can be revealed is the decisiveness of the speaker. It is

important for the course of the business process whether an assertion is made, a question raised, a wish uttered, a commitment, request or proposal is made.

Assertions by trading partners are not neutral facts or observations, but subjective opinions that are coloured by the partners' interest and pragmatic intentions. Business people communicate by means of natural language. Using natural language they exchange information, but also opinions, directives and feelings. Utterances that express those opinions (with the objective to bring something about in the real world) are called speech acts, as explained in section 2.10. Speech Act verbs need to be added to the assertions that are stored in the B2B knowledge base.

We follow the taxonomy of Searle [29] in categorizing speech acts:
Assertives commit the speaker to something being the case (e.g., stating: "the cat is red")
Directives try to get the hearer to do something (e.g. ordering: "get me a red cat")
Commissives commit the speaker to some future course of action (e.g. promising: "Tomorrow I'll find you a red cat")
Declaratives bring about a (new) state of affairs by merely declaring it (e.g. declaring: "Let the cat be red!")
Expressives express the speaker's attitudes or feelings (e.g. "What a beautiful red cat!").

Each type can be present in a business context. When some consumer or procurement officer communicates to the market some desire to procure specific goods or services, he utters an expressive. When some company responds with an offer, it issues a commissive. The buying party may place an order by means of a directive. The seller may confirm having consigned the goods with an assertive. Finally, when the customer has paid, the receipt of the payee may be regarded as a declarative that the deal has been settled.

An utterance, as a consequence, is not only a proposition that assigns property values to concepts or entities. Each utterance carries additionally the identification of the sender organization and his intention. The latter can be indicated by a speech act verb.

Within each of the five categories, Speech Act verbs may be sequenced in the order of 'strength' of the intention. Searle calls this strength 'illocutionary force'. For example a Commitment (in the Commissive category) is stronger than an Intention. In the Assertive category, a Statement is stronger than a Believe or a Prediction. The strength of the illocutionary force may have strong legal and commercial consequences. If some trading partner has stated that he will Attempt to perform a certain action, and he fails to do so, the failure has far less consequences that when he had Promised it.

In this thesis we do not present a mechanism to define new Speech Act verbs on-the-fly. We assume that the verbs to use (with their legal consequences) have been defined in advance of establishing the trading relationship. When such mechanism is to be defined however, it should define a way to position a new Speech Act verb on a specific place at the scale of illocutionary force.

A B2B knowledge base is populated by means of utterances of the communicating parties. Each utterance assigns some predicate (a property or property value) to a concept or entity. So an utterance is logically a proposition. But it is a proposition, uttered with some intention by a subjective party. The intention is expressed by means of a speech act verb.

The intentions with which utterances are communicated, are significant for the semantics of concepts and entities (an offered delivery is different from a realized delivery) and for the process flow. In many cases a seller will not perform a delivery unless a buyer has ordered it.

Intentions therefore will be used both for defining the core business ontology in chapter 9 and for the negotiation of business processes in chapters 7 and 10. Speech act verbs, that identify the intention, form an essential part of an utterance. Therefore the Speech act verb is part of the utterance structure that is proposed in section 6.2.

5.7 Meta layers

An important design principle that we shall follow is the principle to treat concepts and entities in a similar way. Business partners need to manipulate both. In business utterances observations concerning existing objects may be expressed, but also definitions of new concepts. So the utterance structure must cater for both.

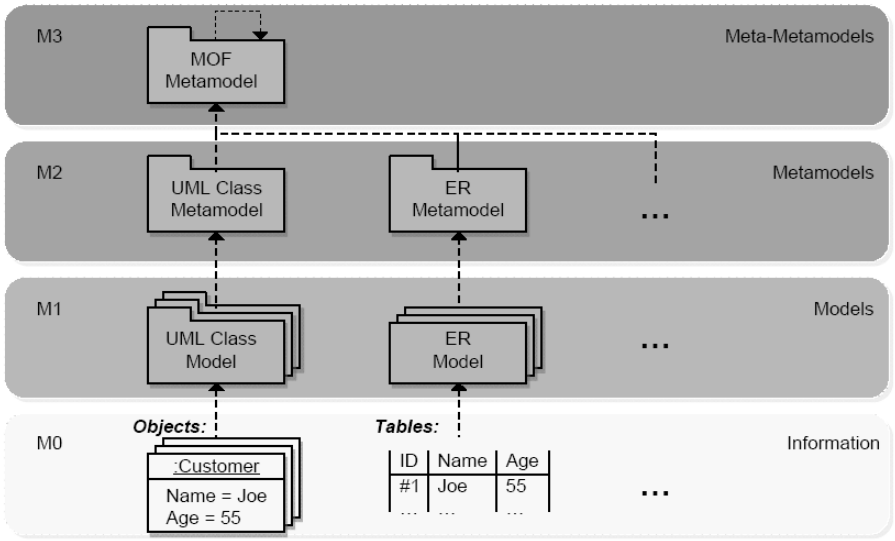


Figure 5.8 Classical MOF

In this thesis we follow the meta-levelling approach as it is defined in the classical version of the Meta Object Facility of OMG [30][3]. At the lowest level (MOF M0) the information on individual entities or instances is positioned. At level M1 the model is positioned that contains the concepts and their associations. At level M2 meta-concepts are defined, such as the "Concept" and the "Property" concept and the features of the particular language. See figure 5.8.

The features presented ('concept', 'entity', 'property', 'event', 'role', 'quantification', 'value space', 'namespace', 'based on') reside at M2 level in MOF. Definition of new concepts resides at M1 level. Instantiation and observation of entities are manipulations at M0 level. In chapter 11 it is shown how M1 level manipulation may be implemented in an (M0 level) operational system that enables businesses to make their B2B systems dynamic: they may automatically negotiate semantics and introduce new concepts without having to reprogram or reconfigure their information systems.

In order to manipulate the knowledge base data on both model- or concept-level (M1) and on instance- or information-level (M0), the knowledge base is designed as a dynamic model in the MOF M1 layer. In the M1 layer a mechanism is defined to specialize generic concepts to more specific concepts. If that mechanism is applied recursively, it is possible to define a super-concept that is the most generic and that collects all instances of all specialized concepts. When specializing concepts, ultimately instances may be regarded as deep specializations of a concept. Going further, even observations may be regarded as specializations of (observations on) instances. See figure 5.9.

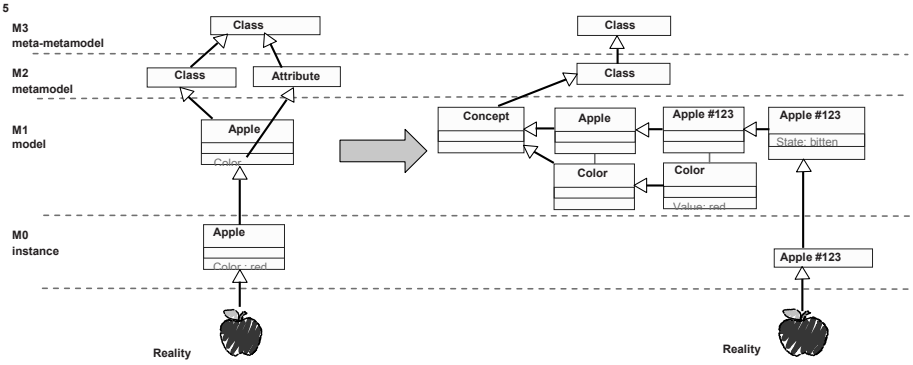


Figure 5.9 MOF

The left part of figure 5.9 shows the traditional modelling practice. Instances are kept at a separate level (M0) from concepts (M1) and meta-concepts (M2). In a B2B knowledge base we keep information on instances at M1 level as well (in modelling languages such as UML and ORM this is supported). We also place a generic concept called "Concept"

[3] in the newest version of MOF, meta modelling is defined recursively, allowing an arbitrary number of meta levels

in the M1 layer. All other concepts at M1 are then based on that generic “Concept” concept. Thus it represents the “Class” class at M2 (meta-model) level.

As a B2B knowledge base is to be mapped on multiple modelling, ontology and exchange languages, the number of model elements at M2 level should be as small as possible. A small number of M2 elements also make negotiation on model level (M1 and M0) easier. As explained above, in a B2B knowledge base the separation between levels M0 and M1 is not as strict as it usually is in an information system. In section 6.3 an M2 meta-model for the B2B knowledge base is presented.

5.8 Functional types of assertions

In the previous sections a few types of sentences that are used in B2B relationships were identified, such as Definitions of new concepts, Expansions and Restrictions. In this section a few more types of sentences are introduced that are needed for business communication, such as Observations, Perceptions and States. At the end of this section an overview is given of the sentence patterns found, with their function for a B2B relationship.

In a traditional operational EDI system, where all metadata has been defined beforehand, only two types of utterances are being exchanged (although the difference between the two is not always explicitly indicated): Instantiations and Observations. Both reside at MOF M0. In a dynamic B2B system it should be possible to do manipulations at M1 as well, leading to two more types. Moreover, in order to be able to define information requirements, preconditions and process flows, two more utterance types are needed.

Concepts and instances are added to and changed in the knowledge base by means of assertions. Five basic assertion types can be identified: Definitions, Expansions and Restrictions (to add and change concepts), Instantiations and Observations (to add and change instances). In addition two more assertion types are introduced in order to define and monitor business process flows: States (to define process choreography) and Perceptions (to define information requirements).

Observations

The most basic assertion is the assertion about an observation of a real world phenomenon. As said in chapter 4, both communicating parties observe that part of the real world that is relevant to their business relation. Their observations are made from their respective perspectives. They need to communicate the observations in order to synchronize their knowledge and to be able to do business.

Observations do not only need to apply to phenomena that happen beyond their control. They include “observations” of their own behaviour, but also of their decisions and inferences.

An observation assigns one or more property values to some instance that is already known. The instance needs to be defined or declared in advance, the properties that are observed too. Observations on instances contain values for two kinds of properties: identifying properties and observed properties. The identifying properties are stated to reference the particular instance. Each observation assertion needs to cite the identifying property values. See section 5.6 for naming and identifying instances. The observed properties contain the newly observed information relevant to an instance.

In fact two ways exist to reference an existing entity or instance (‘existing’ means: existing in the knowledge base). The instance may be referred to by name, as an instance may have been assigned a unique name (within the namespace of the name of the concept it instantiates). In practice however, most instances are not named, but identified by means of one of their identification schemes. The name is then assigned as a temporary ID for use during the transaction only.

For instance, when a Car is instantiated, it is assigned a name (‘My_ Car’), and the identifying properties (e.g. License Plate and Issuing Country) are assigned values. The name is used to fill other remaining properties, such as colour.

[#0] Define Concept with Attribute

[#1] Define Vehicle, based on #0, with Identifier_ Attribute
[#2] Expand Vehicle, based on #0, with Characteristic_ Attribute

[#3] Define Car, based on #1, with License plate_ Identifier
[#4] Define Car, based on #1, with Issuing Country_ Identifier
[#5] Expand Car, based on #2, with Colour_ Characteristic
[#6] Expand Car, based on #2, with Weight_ Characteristic

[#7] Instantiate My_ Car, based on #3, with License Plate = ‘JZ8845’
[#8] Instantiate My_ Car, based on #3, with Issuing Country = ‘NL’
[#9] Observe My_ Car, based on #5, having Colour = ‘Blue’

In later observations, reference may be made to ‘My_ Car’, or to the Car with the applicable identification.

[#10] Observe My_ Car, based on #6, having Weight = ‘750kg’

Or, alternatively:

[#10] Observe This_ Car, based on #3, with License Plate = ‘JZ8845’
[#11] Observe This_ Car, based on #4, with Issuing Country = ‘NL’
[#12] Observe This_ Car, based on #6, having Weight = ‘750kg’

The fact that the stereotype of #10 is Observation, and not Instantiation, implies that an instance with the stated identifying properties must already exist in the knowledge base.

The format of an observation is therefore:

The [concept name] with [ID property1 = value1, ID property2 = value2, …] has [property3 = value3].

In fact this observation is a bundle (transaction) of several utterances: two utterances identifying the entity, and one utterance that set the property values.

Perception

Definitions and expansions together define the full structure or content model of concepts. Not all phenomena affect each property. When business partners negotiate a business process, they are defining the sequence of phenomena that may happen during an instance of the process. Each phenomena instance is represented in the (future) business conversation as a transaction: an atomic exchange of information.

To be able to define the content structures of individual transactions (sets of assertions that describe the observation of individual phenomena), sub-structures of the content model of concepts need to be defined. It must be possible to control which values may be assigned in a specific observation in a certain business process context. So it must be possible to define *perceptions* on entities belonging to a certain concept that contain a subset of the properties of the entities.

A perception is a selection of properties of a concept, in order to exchange information about future instances of that concept. Perceptions resemble View definitions in database languages such as SQL. Database views however can also offer a selection of instances, constrained by a “Where” clause (cf. SQL). Perceptions only select properties (the SQL “Select” and “From” clauses).

A perception is therefore a derivation of the structure of properties of a concept. Whenever information concerning an instance is updated, a perception on its structure is used. In general, the perception always contains at least one set of properties that identifies the instance. Perceptions leave out properties because they are not relevant to the communication in a certain phase of the business process, and not because those properties don’t exist.

A perception is named within the name space of the perceived concept.

The format of a perception is:

The perception on concept [concept name] with name [perception name] includes [property1]

Example:
The perception on concept Car with name Car Colour Report includes Body Colour.

States

In business processes, many allowed steps or transactions are conditional. Payment is often conditional w.r.t. delivery. A call-off transaction is conditional w.r.t. the existence of a contract. Conditions are defined as States. States have the scope of an entity, but may include properties of other entities that are associated with the entity by means of property relations.

A State defines a subset of the extension of an entity type at a certain moment. A State narrows down the value space of the (non-rigid) value spaces of the entity type. At a given moment, a State has a one-to-one relation with the collection of entities that are in that State. States will be defined more precisely in chapter 7, as the state concept is needed to define processes. The name of a state is defined within the namespace of the concept the state is defined upon.

When states are used as conditions for other utterances, the concept names in the state definition refer to the concept instances in the other utterance. They are used as placeholders. E.g. if the state is defined as “Car is red”, this does not mean that all cars need be red if the state is to be evaluated true, but only the individual car in the observation that has the state as its precondition.

The format:
The state with name [state name1] of concept [concept name] has [property1=valuespace1] and has preconditions [state name2].

Instantiations

Often, in the course of a business process new entities need to be introduced as instantiations of previously defined concepts. It must be possible to define new economic events, products, services, locations, etc.

An instantiation defines or declares the existence of a new instance of a concept. It needs to set the values of at least one set of identifying properties. Additionally other property values may be set as well by means of observations.

Format:
An instance of concept with name [concept name] has [property1=value1]

Example:
An instance of concept with name Car has license plate NL/JZ-88-45.

Summary

In this section seven utterance types have been defined: definitions, expansions, restrictions (to manipulate concepts), instantiations, observations (to manipulate entities), states and perceptions (to define process flows). In the examples, e.g. in chapter 12, it is shown that all types are needed and that they are sufficient for most inter-organizational trade processes.

Utterance type	Utterance structure
Definition	A [new concept name] is a [existing concept name] with [property1 = valuespace1]
Expansion	A [concept name] has [property1=valuespace1]
Restriction	The [meta-attribute] of concept [concept name] is restricted to be [value space]
Instantiation	An instance of concept with name [concept name] has [property1=value1]
Observation	The [concept name] with [ID property1 = value1, ID property2 = value2, …] has [property3 = value3]
Perception	The perception with name [perception name] on concept [concept name] includes [property1]
State	The state with name [state name1] of concept [concept name] has [property1=valuespace1] and has preconditions [state name2]

Table 5.3 Utterance types

All utterance types have the general pattern that reference is made to a more generic concept for defining a specific concept, state, perception, instantiation or observation. Only for expansions the reference is implicitly made because the concept that is expanded is named within the namespace of the generic concept. For the other utterance types the reference is explicit.

The seven types of utterances may be used to define and monitor a B2B relationship. Each utterance adds knowledge to the knowledge base. The formats of the utterance types have been shown in an informal way. In section 6.2 the utterance types are presented in a tabular format and in section 6.3 in a meta-model. In chapter 8 the utterance structure and the required structural features and attributes are mapped on existing modelling languages.

6 Structure of utterances

Summary

An utterance may be structured as a row in a table. The row consists of 5 parts, each containing some utterance attributes:

- The Core Proposition states a (property-)relation between two concepts
- The Cardinality states the allowed repetition of that relation
- The Definition states (a.o.) the party who is allowed to utter the defined concept
- The Intention specifies the speech act
- The Identification contains a timestamp, ID number of the utterance and the ID of the parent utterance it is based on.

Each utterance must be based on a previously entered utterance. This way utterances act as a filter for future utterances. The table and the 'based on' rule may be implemented in a (middleware) system, that checks the utterances in a B2B conversation for consistency. It may also be implemented in a reactive system that acts on behalf of a business partner. Such system must have access to the business partner's information system(s) and to the business partners himself for policy decisions.

Except in a table format, utterances may also be modelled more extensively. Such a model is also presented in this chapter and is explained in the Annex.

Data types may be defined and refined in utterances as well. This is shown in section 6.5.

6.1 Introduction

Utterances, such as definitions and observations are exchanged by means of some communication protocol over a communication channel and are used to update a business information system. In a traditional business relationship the communication channel is the (‘plain old’) telephone or fax network, or the oscillating air pressure (speech) in a face to face meeting. The utterances are then represented in the form of natural language or semi-structured business forms. The utterances (and the B2B knowledge base) are stored in business peoples’ minds and in paper archives. All aspects as introduced in this chapter are part of such a traditional system, although they are usually not explicitly indicated.

When business partners use computerised business information systems and interconnect them they probably make use of a computer network (e.g. the Internet) and some structured syntax (such as XML) to structure the data to be exchanged. In most present standard EDI and XML systems many aspects as mentioned in chapter 5 (notably speech acts and M1 manipulation) are not explicitly supported. Speech acts are usually implicitly defined in the functional definition of the messages. Some protocols, such as FLBC [23], OAG’s BODs [31] and UN/CEFACT CCMA (Core Component Message Assembly, an

abandoned draft) [32] support explicit speech acts, but this mechanism is rarely implemented, if at all. M1 manipulation, to our knowledge, is not supported at all

In chapters 8 and 11 ways are described how to use the aspects in standard B2B messaging protocols. In order to make such mappings the structure of B2B utterances must first be defined in a more precise way. This is elaborated in this section. It must be stressed that the structure as presented in this section is not meant to be directly used as communication syntax or protocol. It is an abstract description of the aspects that are to be present in dynamic B2B communication. The aspects must be mapped on practical (standardised) message structures and protocols, preferably those that are already widely implemented. As the exchanged business information must be processed by business information systems, mapping must also be made to languages that model such systems. Such mapping is also described in chapter 8.

A B2B knowledge base stores the knowledge that is shared by business partners for some business relationship. That knowledge is exchanged between the business partners by means of (speech act) utterances. The structure of the knowledge base and the structure of the utterances are closely related: basically, the knowledge base stores the utterances. In this section the utterance structure is further discussed and placed in a tabular format. In chapter 8 the tabular structure is mapped on some existing modelling languages.

As shown in section 5.8 an utterance may be an observation, the definition of a new concept, an expansion, a restriction, a perception, a state or an instantiation. The core structure of each of those types is similar: they assign properties or property values to concepts or entities.

Not all utterances are meaningful or allowed in a business conversation. Parties must agree which utterances *are* allowed, and the communication system should filter out the rest. The filtering may be as strict as the present practice is, only allowing utterances in the form of messages that comply with static schemas, or more flexible. If more flexibility is needed than static schemas can offer, the definition of allowed utterances must be structured differently than in message schemas. We propose a simple but effective way, that will allow not only more flexibility (e.g. the possibility to negotiate about the meta-data) but also to define the process flow, as is shown in chapter 7.

The simple rule we propose to introduce in B2B knowledge bases is:

Each utterance (except the initial utterance) must be based on an utterance defined previously in the conversation.

The core of an utterance is the triple concept-property-concept. The first part is the name of a (source) concept. The name is defined within the namespace of a wider concept and forms the namespace of narrower concepts. The second part is a term that denotes the event and the roles another (target) concept and the source concept plays with regard to the event. That term is a combination of a verb and the thematic roles as explained in section 6.2. The third part is the name of the target concept. Concepts include data or

value types. The concept name may be a value (= the name of a value type). Values and value types are analysed in section 6.5.

This core structure
Source concept - Source Role - Verb - Target Role - Target concept
is valid for each of the seven utterance types:

- For *observations* the source concept denotes the identification of an already known entity, the property (source role-verb-target role) a previously defined property of the entity and the target concept a value or the identifier of a second entity.
  example:
  JZ8845_ Car – Theme - Arrive - Location – Zwolle_ Town
  ("The car with ID JZ-88-45 arrived in Zwolle": the particular car had already been instantiated and the arrival location is updated)
- For *definitions* the source concept is the name of a new concept within the name space of an existing concept, the property an existing property of that existing concept or a specialization of such property and the target concept a value space or the name of a known concept
  example:
  Tomato_ Fruit – Goal - Measure - Color_ Measurement_ Essence – Red
  ("A Tomato is a Fruit with a red color": Fruit had been defined before, Tomato is a specialization of Fruit)
- For *expansions* the source concept is an existing concept, the property an existing property or a specialization of such property of the concept the existing concept is based on and the target concept another existing concept or a value space
  example:
  Tomato_ Fruit - Theme - Sell - Best Before_ Date_ Time - Date
  ("A Tomato has a 'Best Before' selling date": A Tomato had been defined before and is assigned a new attribute: 'Best Before' selling date)
- For *restrictions* the source concept is an existing concept. The property and target concept are properties/targets already defined for the concept.
  example
  Tomato_ Fruit - Theme - Sell - Best Before_ Date_ Time - Date; Max rep=1
  Restrictions are used to limit meta-attributes, such as repetition factors and preconditions for existing concepts.
- For *perceptions* the source concept is a specialization of an existing concept. The property and target concept are properties/targets already defined for the concept.
  example
  Label_ Tomato_ Fruit - Theme - Sell - Best Before_ Date_ Time - Date
  ("The Best Before selling date is to be printed on the label of the tomato")
- For *states* the source concept is the specialization of an existing concept, the property an already defined property and the target concept a subset of the concept that forms the property target
  example
  Rotten_ Tomato_ Fruit - Theme - Sell - Best Before_ Date_ Time - Obsolete_ Date
  ("A tomato is rotten if its Best Before selling date is obsolete")

- For *instantiations* the source concept is the name of the instance as specialization of a concept, the property an (identifying) property and the target concept the value of the property.
  example
  `JZ8845_ Car - Goal - Register - License Plate_ Essence - NL/JZ8845`
  ("A car exists with 'NL/JZ-88-45' as license plate")

Each of the three parts must have been defined previously in the knowledge base, possibly at a more generic level.

So utterances have the characteristics of a Russian doll, see figure 6.1. Each of the three parts of an utterance must fit into the (semantic) space created by a previous utterance. An utterance also creates space for future utterances.

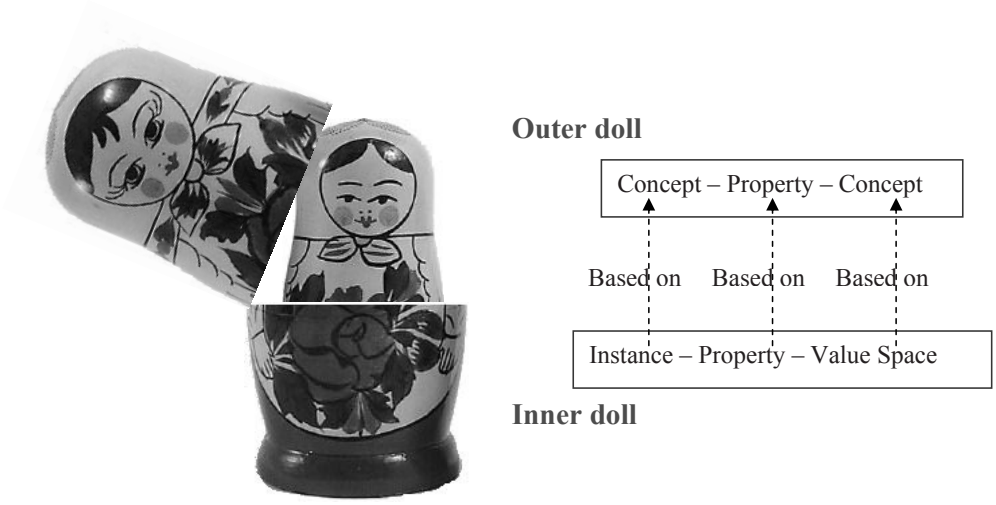


Figure 6.1 Utterances as a Russian doll

The 'based on' relation between concepts/instances and properties is an ontological relation. A concept that is based on another concept must have an intension (set of criteria) that is more restrictive than the intension of the other concept. Its extension (set of possible instances) must be a subset of the extension of the other concept. A property that is based on another property must have a meaning that is narrower than the meaning of the other property (like Length is narrower than Dimension) and the set of values of the property (the "target" concept of the property) must be a subset of the set of values of the other property.

The 'based on' relation is also defined lexically. The name of a concept that is based on another concept is defined within the name space of the other concept. The name of a property is defined within the name space of the property the property is based on.

6.2 Table format

In this section the structure of the knowledge base is represented as a table. This approach is inspired by the work of Renssen [33], who represented product ontologies as tables. The B2B knowledge base table is built up part by part.

Core

A representation of the core of an utterance is:

Source Concept Name	Property Term	Target Concept Name

Table 6.1 Core utterance

An utterance defines in its core a property of a concept. A concept may be the type of an entity or an instance. The property may have as its target another entity type or instance, or the value space of a data type. The value space may be narrowed to a single value.

The Property term that relates the source concept to the target concept consists of a Verb (defining the event that relates the two concepts), a Source Role and a Target Role.

Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name

Table 6.2 Verb and roles

For instance the fact that a fastener fastens artefacts can be represented as shown in table 6.3.

Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name
Fastener	Instrument	Fasten	Theme	Artefact

Table 6.3 Roles example

Intentions

As explained in section 5.6, the agent who uttered the utterance and his intention for doing so is relevant for the semantics and for the pragmatics. Utterances are subjective. So in addition to the tuple <source concept-role-Verb-role-target concept>, the party ID and the speech act verb must be part of the utterance.

Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name

Table 6.4 Party and intention

Most business conversations are conversations between a buyer and a seller. For generic ontology and process definitions therefore 'B' and 'S' are used as identifiers of the participating parties. Other roles in business conversations may exist, such as government agencies, information providers (who do not need to sell the information), etc. In actual knowledge bases parties will be identified uniquely among all business partners that may potentially participate in a business conversation.

Intentions are chosen from a (small) list of applicable speech act verbs. Some of those verbs have already been mentioned, such as Desire, Assert, Commit, Order, Claim and Declare. The set may be extended, but it is expected that the number of business speech act verbs will stay limited. It is important that the legal and economic impact of each speech act verb is known and accepted by the participants of a conversation. Speech act verbs are therefore subject to standardization.

Sequence

For the business process it is essential to know in what sequence the utterances were (and are to be) added to the knowledge base and sometimes the exact date and time of an utterance is crucial. Utterances are bundled in transactions and all utterances in the same transaction have the same time stamp. The timestamp is also an important attribute of each utterance.

Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name

Table 6.5 Sequence and timestamp

Multiple utterances may have the same timestamp. Utterances must therefore be identified, the time stamp does not suffice. This can be done by a simple sequence number. The sequence number is to be unique within the scope of the conversation between two partners. Utterances that are part of a Propose/Accept or Propose/Reject pattern (see later) however bear the same utterance number. To uniquely identify them both the utterance number and the timestamp are needed.

Based On relationship

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name

Table 6.6 Based on relation

Structure of utterances

Each utterance in the knowledge base (except the root utterance) must be based on a (one) more generic previous utterance. This rule is valid for all utterance types. Definitions must be based on Definitions or Expansions of more generic concepts. Perceptions, Expansions, Restrictions, States, Instantiations and Observations must be based on Definitions or Expansions. Utterances are identified by means of their sequence number. The sequence number is therefore used to base an utterance on a previous utterance.

The Source Concept, Verb, Role and Target Concept names of an utterance are defined within the namespaces set by the Source Concept, Verb, Role and Target Concept names of the utterance it is based on. Other restrictions apply when basing an utterance on an earlier utterance, such as the allowed Recording Party and Intention. These restrictions are explained later in this section.

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name
1	0	00:01	B	Assert	Fastener	Instrument	Fasten	Patient	Artefact
2	1	00:02	S	Assert	Rivet_ Fastener	Instrument	Permanently_ Fasten	Patient	Plate_ Artefact

Table 6.7 'Based on' example

In the examples of table 6.7 not the full name of the concepts, verbs and roles is shown. The full name consists of the names of all concepts a concept is based on, until the root. So for instance Rivet_ Fastener_ Artefact_ Object_ Entity. In most example cases the naming tree is made up of two levels (e.g. Rivet_ Fastener). If the knowledge base is implemented in a data base only one level (Rivet) suffices. The full name can be derived by following the based-on trace.

Quantification

As mentioned in section 5.6, quantification is an important part of an assertion on concepts. It must be possible to state that each instance (the logical ∀) of a concept has the property, or only some instance (logical ∃). It is also significant to know how many properties of the same type an instance may have. E.g. an instance may only have one name, but many parts. Both quantifications can be represented by a "repetition factor", a mechanism that is used in the modelling language UML, but also by ERD. The Repetition factor indicates the minimum and maximum repetition of the properties of a certain property type of an instance. If the minimum repetition is one, all instances must have (at least one instance of) the property.

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Min Repetition	Max Repetition

Table 6.8 Quantification

The repetition factors must obey the repetition factors stated in the Based on Utterance. If the utterance, the utterance is based on, had stated that the minimum repetition is 1, the minimum repetition of the new utterance may not be 0. If the maximum repetition of the Based On utterance is 5, the maximum repetition must be equal to or lower than 5.

However, data base implementations need not always to include properties that all instances of a concept possess. If a tomato for instance is defined as a "fruit" with colour "red", then it would be redundant to enter the colour as an attribute of each tomato instance. Here the difference is hit between an ontology and a data model. Definitions are usually included in an ontology but not in a data model, and certainly not in a data base.

Therefore it is not necessary to repeat mandatory properties of concepts for each concept that is based on it. It is then assumed that the instances of the child concept have the property, without having to specify it. As the property is mandatory (minimum repetition = 1 at the parent concept), the property may be assumed. Note that a non-mandatory property that is not repeated is non-existent or not relevant for a child concept.

Process flow definition

The future course of the conversation and the business process need to be controlled. This is done by specifying the party or parties and their intentions of utterances that are based on the utterance. A 'To utter by' attribute specifies the party or parties that may utter utterances based on this utterance and the With Allowed Intention specifies his intention or intentions.

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Min Rep	Max Rep	To utter by Party	With allowed Intention

Table 6.9 Allowed parties and intentions

These two utterance elements restrict the utterances that are based on the utterance in which they are present. So an utterance with the "To utter by Party" element set on "S" (Seller) may only be the basis for future utterances of the Seller.

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Min Rep	Max Rep	To utter by Party	With allowed Intention
1	0	00:01	B	Order	S	Agent	Deliver	Patient	Consignment	1	n	S	Commit
2	1	00:02	S	Commit	S	Agent	Deliver	Patient	123_ Consignment				

Table 6.10 Allowed utterance example

So restrictions apply when basing utterances on other utterances. Except that the concepts, names and roles must be more restrictive than those of the utterance the utterance is based on, the recording party and the intention must fall within the restrictions that are specified with the higher level utterance.

Preconditions

For the control of future communication the 'based on' relation (plus the future party and intention) is not sufficient. It must be possible to control future utterances in a more refined way. As is shown in chapter 7 in many cases it is needed to add to the utterance structure a precondition. A precondition is a set of states (or any other utterances) that must be present or absent when some subsequent utterance is based on the utterance that is being defined. Implicitly the utterance itself is always a precondition for subsequent utterances that are based on it.

A Precondition refers to previously recorded utterances. Utterances are numbered in order to make such reference. Preconditions are Boolean AND and OR clauses of sets of utterances.

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Min Repetition	Max Repetition	To utter by Party	With allowed Intention	Precondition #

Table 6.11 Preconditions

Preconditions in fact do not refer to States themselves, but to instantiations and observations that obey the State rules (manifestation of States). This means that one or more instances must exist that are in that state in order to obey the precondition. Instances are not always explicitly defined to be in a state. They may be defined being the instance of a concept, and observations assign values to their properties. At the time of evaluating the validity of utterances based on the utterance in which the condition is referred to, the condition will be evaluated.

Preconditions, defined for an utterance, also apply to the utterances that are based on the utterance. Preconditions may be combined in a Boolean expression. That way arbitrary processes may be defined with serial and parallel paths (see chapter 7). Preconditions do not need to be represented as States. Other stereotyped utterances, such as definitions and expansions may also serve as the basis of preconditions. So the utterance of an order may be a precondition for an invoice.

Transactions

A single event may trigger multiple utterances. The utterances belong together in a transaction and will have the same timestamp. To define that the utterances are part of the same atomic transaction they can each have as condition that one of the other utterances must be present as well. The condition is a Boolean expression of other utterances. A new utterance may only be based on the utterance for which the "Transaction with #" cell is filled, if it has the same timestamp as utterances that are based on the utterances referred to in that cell.

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Min Rep	Max Rep	To utter by Party	With allowed Intention	Pre-condition #	Transaction with #

Table 6.12 Transactions

Example:

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Min Rep	Max Rep	To utter by Party	With allowed Intention	Pre-condition #	Transaction with #
1	0	00:10	B	Order	S	Agent	Own	Patient	Consignment	1	n	S	Assert		
2	0	00:10	B	Order	S	Agent	Deliver	Patient	Consignment	1	n	S	Commit		1
3	1	00:20	S	Assert	S	Agent	Own	Patient	123_ Consignment	1	n				
4	2	00:20	S	Assert	S	Agent	Deliver	Patient	123_ Consignment	1	n				

Table 6.13 Transaction example

In the example in table 6.13 in utterance #2 is defined that the Seller may commit the delivery of a Consignment (only) together with the assertion that the Seller owns it (utterance #1). In utterance #3 the Seller states his ownership of 123_ Consignment and at the same time utters his commitment to deliver in utterance #4, which is based on utterance #2. Utterances #3 and #4 together form a transaction.

Identification

It should be possible to define a certain property as identifying the instance. As was shown in section 5.6 instances may have multiple identification schemes, each consisting of multiple properties. The schemes may be numbered and the identification indicator may be an integer identifying the scheme. The properties belonging to the same scheme get the same number. Together they identify the instance. E.g. if a car may be identified by its license plate in combination with its registration country or by its manufacturer and its chassis number, the license plate and registration country both get the 'Part of ID' value of 1 and the manufacturer and the chassis number get the 'Part of ID' value of 2.

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Intention	Precondition #	Transaction with #

Table 6.14 Identification

Example:

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Intention	Precondition #	Transaction with #
101	10	00:10	B	Assert	Car	Theme	Have	Registration Country	Country_ Code	1	0	1	S	Assert		
102	10	00:10	B	Assert	Car	Theme	Have	License Plate	Text	1	0	1	S	Assert		101
103	10	00:10	B	Assert	Car	Theme	Have	Manufacturer	Text	2	1	1	S	Assert		
104	10	00:10	B	Assert	Car	Theme	Have	Chassis Nr	Numeric	2	1	1	S	Assert		103
201	103	01:00	S	Assert	123_ Car	Theme	Have	Manufacturer	VW							
202	104	01:00	S	Assert	123_ Car	Theme	Have	Chassis Nr	533789							

Table 6.15 Identification example

In table 6.15 it is shown how two identification schemes may be defined, each consisting of two properties. The instance shown is identified by one of the two schemes. Note that although the License Plate is identifying the Car, it is not always present, e.g. when the Car has not been registered yet.

Stereotypes

Whether an utterance is a definition, expansion, restriction, perception, instantiation, state or observation is coded in the Stereotype column. The 'With allowed Stereotype' column controls whether the utterance may serve as the basis for further definition or only for instantiation and observation. In combination with the 'Restriction' stereotype this can be used to 'lock' a level of conceptualization. In the complete knowledge base all levels are in principle available for further specialization, including the most upper levels. This would allow trading partners to propose very generic concepts. In a real business community (and certainly in a bilateral relationship) it is needed to fix or lock an ontology, only allowing further refinement but not redefinition of generic concepts.

Source and Target Concept names refer to the concept or the type of instances they are naming. A Definition will specialize a concept, so the name is different from the name of the concept it is based on. However, to allow references to instances in conditions and states, a concept name in a Perception refers to an individual instance of the concept. This way it is possible to formulate complex conditions, without having to introduce new names. An example, illustrating this mechanism is elaborated in section 6.6.

Table 6.16 shows which stereotyped utterances may be based on which stereotypes.

	May be based on	Refers to
Definition	Root, Definition, Expansion	Concept
Expansion	Root, Definition, Expansion	Concept property
Restriction	Expansion	Concept property
Perception	Definition, Expansion	Transaction
State	Definition, Expansion	Entity state
Instantiation	Definition, Expansion, Perception	Entity
Observation	Instantiation	Entity property

Table 6.16 Stereotypes

In principle it could be allowed to base Observations on other Observations. One Observation then creates the affordance to add other Observations. For instance an Order (Instance or Observation) could be the basis of a Delivery (Instance/Observation). In practice however this pattern is difficult to map on legacy systems. In traditional information systems a strict distinction exists between metadata and data, so a Delivery Instance can only be based on a Delivery Definition, Expansion or Perception, not on another (e.g., Order) Instance.

A concept is locked, after having defined refined concepts based on it, by adding a 'restriction' utterance with the 'With allowed stereotype' column being empty. Then no future utterances may be based on the concept. If the 'With allowed stereotype' column contains only Definition, Expansion and/or Restriction, the concept may not be instantiated directly. It then is to be regarded as an abstract concept (in terms of UML).

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #

Table 6.17 Stereotype

Action

Additionally it is needed to state the Action (add/propose/accept/withdraw). All utterances will remain in the knowledge base, but some may be not be valid any more. The Action mutates the state of the utterance. This utterance state is another concept than the State stereotype: it governs the validity of the utterance in the knowledge base.

Utterances that add definitions, expansions and perceptions (M1 level utterances) need to be confirmed by the counterpart. Only if a party has the ability to process the new or expanded concept in the information system (s)he will accept it. So for definitions, expansions and perceptions a propose/accept pattern exists. Propose and Accept are Actions. Definitions of new concepts are first proposed to a trading partner (Action = Propose) and must be accepted by the counterpart (Action = Accept). Only after acceptation the definition may be used to base utterances on it.

Structure of utterances

The Action element is also used for corrections. Utterances may be withdrawn.

Utterance #	Based on Utterance #	Time stamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #

Table 6.18 Action

An utterance with an Action of Accept, Reject or Withdraw must be a complete copy of the utterance it is accepting, rejecting or withdrawing, with the exception of the Timestamp and the Uttering Party. It has the same utterance number as the originally proposed utterance.

Summary

Identification				Intention			Core Proposition					Cardinality			Definition				
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #

Table 6.19 Utterance sections

The structure of the B2B knowledge base is presented in this section as a table. The table format allows the implementation of the knowledge base by for instance a spreadsheet, with integrity and validation rules. In section 6.4 a meta-model of the knowledge base is presented. The structure of the knowledge base may be mapped to modelling languages, database schemes and message syntaxes, which is shown in chapter 8.

In general one can say that the first four columns of the table identify the utterance, columns 5 through 7 denote the intention of the utterance (action, stereotype, intention of the proposition), columns 8 through 12 define the proposition, columns 13 through 15 set the cardinality and columns 16 through 20 regulate future utterance that may be based on this one.

In chapter 8 the knowledge base structure, as presented in the table, is mapped to a number of modelling and exchange languages. This allows business partners each to use their own modelling language while still complying with the agreed structure. In chapter 8 the structure is also mapped to UN/CEFACT Core Components. Core Component structures are used to define the semantic structure of the information that is exchanged between the partners. They can be mapped on XML structures in a standardized way. That mapping is defined in the UN/CEFACT Naming and Design Rules.

6.3 Conditions

States may be used to define conditions: value constraints, preconditions, post-conditions and dynamic constraints. The ability to define conditions is an important mechanism: it is used to define processes as is elaborated in chapter 7. In this section it is shown how to define various conditions in the knowledge base structure.

Definition of states

States are defined by filtering property values of a concept. This can be illustrated by defining three States of a Delivery: Requested Delivery, Planned Delivery and Actual Delivery.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
0	0	0:00	B	Add	Def	Assert	Concept	Role	Verb	Role	Concept		0	n	B, S	Def	any		
11	0	00:10	B	Add	Def	Assert	Delivery		Deliver	Requested_ Time	Date		1	1	B	Sta	Assert		10
12	0	00:10	B	Add	Exp	Assert	Delivery		Deliver	Planned_ Time	Date		0	1	B	Sta	Assert		10
13	0	00:10	B	Add	Exp	Assert	Delivery		Deliver	Actual_ Time	Date		0	1	B	Sta	Assert		10

Table 6.20 Delivery definition

In Utt# 11 through 13 of table 6.20 a Delivery event is defined with three dates. For this example the definitions only afford the Buyer (col. 16) to assert States (col. 17) based on the definitions.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotyne	With allowed Intention	Precondition #	Transaction with #
15	11	00:30	B	Add	Sta	Assert	Requested_ Delivery		Deliver	Requested	Date		1	1					
16	12	00:30	B	Add	Sta	Assert	Requested_ Delivery		Deliver	Planned	Date		0	0					
17	13	00:30	B	Add	Sta	Assert	Requested_ Delivery		Deliver	Actual	Date		0	0					
20	12	00:31	B	Add	Sta	Assert	Planned_ Delivery		Deliver	Planned	Date		1	1				15	
21	13	00:31	B	Add	Sta	Assert	Planned_ Delivery		Deliver	Actual	Date		0	0					
25	13	00:32	B	Add	Sta	Assert	Actual_ Delivery		Deliver	Actual	Date		1	1				15 OR 20	

Table 6.21 States

In Utt# 15 through 17 the Requested Delivery State is defined: it has a Requested Date, but no Planned and Actual Dates (their cardinality is set to 0 in columns 14 and 15). In Utt# 20, #21 and #25 the Planned and Actual Delivery States are defined. Both have as precondition the existence of a Requested Delivery State (column 19).

The state machine defined in this way can be represented as a state chart as pictured in figure 6.2.

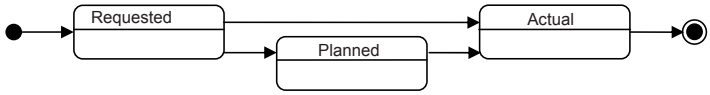


Figure 6.2 State machine

Value types

Value types may be constrained by using formulas as target concept. Such formula may restrict the value space of a data type by comparing the content to certain values, using operators such as =, <, >, <=, >=, <>, Between, Not between, In {a,b,c}, Like pattern (using regular expressions), AND, and OR.

Value types may also be constrained by relating the values of several properties. The set of operators is extended with operators such as +, -, *, /, ^, SQR() (and other mathematical functions) to combine several properties, and Sum{}, Avg{}, Max{} and other statistical and set functions. Reference to other attributes is made using the (source and target, concept, verb and role) names, traversing property paths, in the same way OCL [34] defines paths.

Constraints, as defined by means of states, are by default post-conditions. A state can serve as precondition by referring to it in column 19 from another utterance. Such another utterance may be the basis for another state (to build a state machine like the one in figure 6.2) but it also may be the basis for observations or instantiations.

6.4 Meta-model

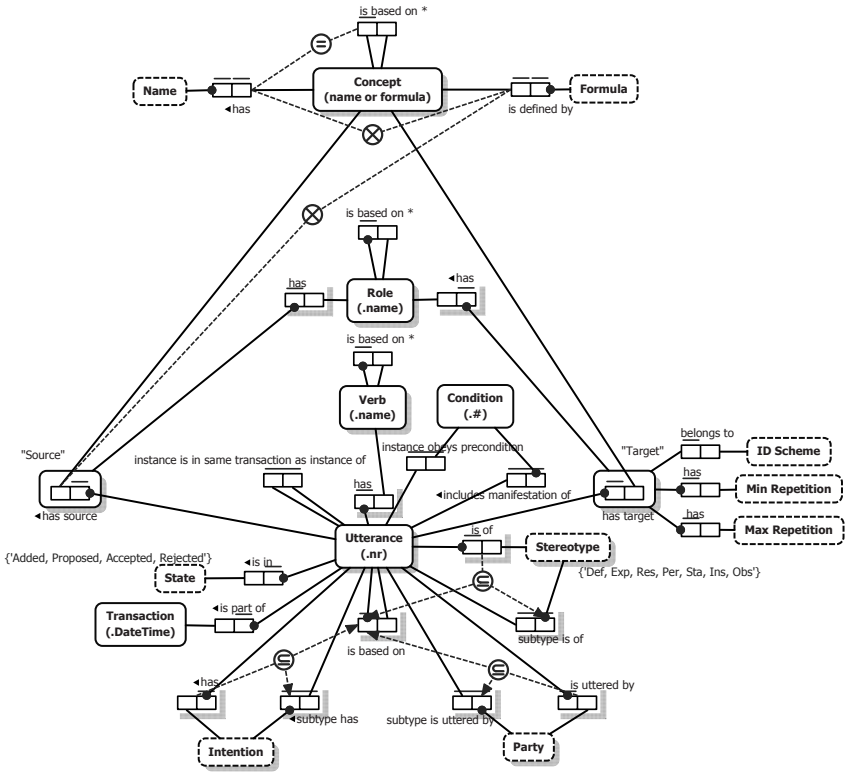


Figure 6.3 Meta-model

In figure 6.3 a meta-model of the knowledge base is represented in ORM. The meta-model is explained in the Annex.

Utterances may pose facts or opinions on concepts (entity types) or on entities (instances). Whether concepts or entities are addressed is determined by the stereotype. Definitions, Expansions, Restrictions, Perceptions and States address concepts, while Instantiations and Observations address entities. Both are subtypes of the Utterance in general. We represent Instances as subtypes of Concepts: an Instance is a Concept with one member. Definitions and Expansions can be the basis of Definitions and Expansions of more specialized concepts. They also may be the basis for Instantiations and Observations. Restrictions, Perceptions and States are always based on Definitions or Expansions (see table 6.16).

The ‘based on’ rules as expressed in table 6.16 also may be represented in a meta-model, as figure 6.4 shows.

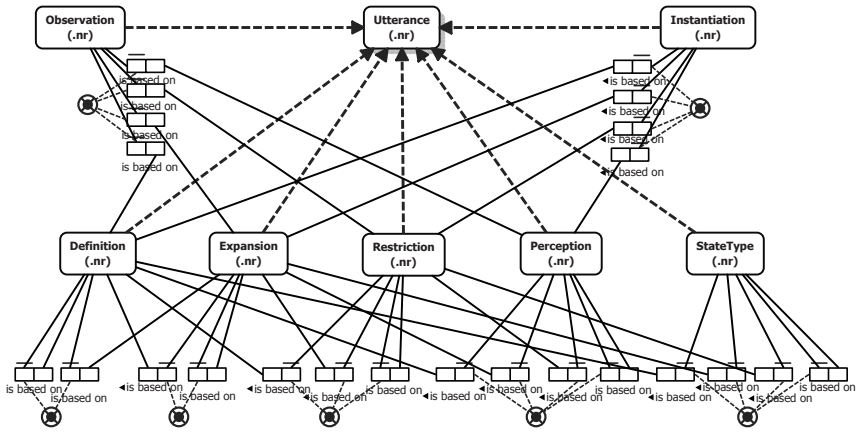


Figure 6.4 ‘Based’ on rules

6.5 Data-typing

In the previous sections a structure is proposed for utterances that add knowledge on concepts and entities to a B2B knowledge base. The knowledge consists of representations of events that result in properties of the concepts and entities. Those properties are roles fulfilled by other concepts and entities. In the presented table structure, concepts and entities (and events and roles) are represented by names. Names are assigned to concepts and entities by means of definitions. The definition pattern has been designed in section 5.5. Definition of concepts (entity types) results in unique configurations of properties. Each named concept has a unique set of properties.

Names may be chosen arbitrarily, with as restriction that they are to be unique within the namespace that is set by the concept the named concept is based on. In practice it is wise to choose names that have some meaning in a natural language (preferably English), to facilitate intuitive interpretation by human trading partners. The natural language meaning is, however, not determining the interpretation by automated systems that process the information: the property configuration is.

Hence, a concept can be referred to by a name and is represented by a set of properties. Each property has a target concept, which - in turn - has a name and a unique set of properties, etc. Each concept forms the root of a tree, where branches and sub branches are concepts as well, that have been defined in the knowledge base. Ultimately the branches end in leaves that have no other concepts than their properties. These leaves are data types.

Entities (concept instances) are identified by one or more sets of identifying properties. These identifiers, like all properties, have other entities as their target. Data type instances, however, are identified by their values. Data type values do not have concepts that are defined elsewhere in the knowledge base as target. The scales on which data type values are defined must be standardized beforehand. Data type values are the basic information elements or building blocks of the utterances that are exchanged between trading partners.

For example, a numeric data type has a number as the value of an instance. A number represents a point on the mathematical numeric scale. A textual data type has as instance values strings of characters of some alphabet. Usually such text has some meaning in a natural language.

Just like names of concepts, entities, Verbs and roles, values of data type instances must be represented in communication messages and in information stores using some sign system. The utterances as represented in the table system in section 6.2 must be supported by the sign systems' syntax. If computer systems are used for communication and storage of information, names and syntax need to be represented in bit patterns.

In this section the structure and representation of data types and data type values are inspected.

The scales on which data type values are projected may be abstract mathematical scales (such as the numeric scale) but may also be physical scales, such as the scale of geographical locations or the time scale. Data type scales need not to be one-dimensional and the different dimensions of a scale may have very different semantics. For example a 'measurement' may consist of a (one-dimensional) value and a measure unit that is defined on the 'measure unit scale' (e.g. the SI system).

It seems not possible to define entirely new data type scales by means of a structured business conversation. Scales must be defined, standardized or agreed beforehand. It is however possible to define subsets on existing scales. This is done by means of 'facets'. Facets may limit the length of the scale, the precision or may define specific value patterns.

Data typing includes the mapping of semantic units or ontological constructs to sign systems. To enable the processing of signs by computers, signs and sign constructs are encoded in binary systems. In fact binary systems are a special kind of sign systems, using bits as signs. Other sign systems include printed text, icons, sounds, etc.

Names and values might be directly represented in bit patterns. However, different computer languages, operating systems and storage technologies use different bit representations for the same functional content. In order to be technology-independent and to be able to specify communication of B2B utterances between computers that use different languages and operating systems, the data type system to use for B2B communication should be layered. In the higher layers of the stack, semantic structures

can be specified, while in the lower layers, mappings can be realized to specific technologies.

In databases, computer programming languages and exchange languages entities are represented as structures of data fields. Data fields are ultimately represented in bits, but have (at M1 level) a certain type. The type defines the semantics of the field (its scale, e.g. the fact that the field represents a number), but also the operations that are allowed on the field and the syntax of the field (allowed bit-patterns).

In information- and knowledge specifications that abstract from technology representation, it would be sufficient to state only the semantics of the data elements involved. At implementation level a generic mapping can then be made between the data type semantics and the bit representation.

One of the differences between data types and other concepts is that data types have a not-too-complex internal structure. It must be noted that some modelling languages, e.g. Express, allow for (very) complex data type structures. For B2B purposes, as mentioned, semantics of all data structures involved should be specified explicitly. This cannot be accomplished by complex data type structures that are void from internal semantics. When defining data types for B2B purposes one should be reluctant in designing complex structures.

The Core Component Technical Specification (CCTS) [27] offers a language to describe data models and messages for B2B relations in a technology- and syntax-neutral way. CCTS 3.0 makes a distinction between Core Data Types (CDTs) and Business Data Types (BDTs). BDTs specialize CDTs: the domain value of a CDT is restricted for a BDT. CDTs (and therefore also BDTs) have a not-too-complex internal structure, but they are not scalars. In the sequel of this section the CCTS data type system is taken as the basis for open B2B communication.

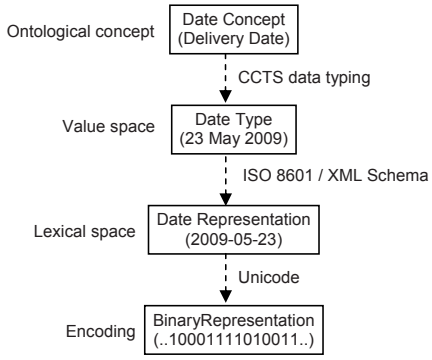


Figure 6.5 Encoding of information

A more generic data type system is described in the XML Schema specification [35]. The XML schema specification makes a distinction between the value space of a data type

and the lexical space of a data type. The value space denotes the semantic scale of the represented information, such as Date Time, Numeric, or Text. The lexical space defines the way in which values in that dimension are represented in Unicode (for XML messaging). ASN.1 [36] follows a similar approach. X.680-ISO/IEC8824 defines how message parts and protocol elements are constructed in an abstract way; X.690-ISO/IEC8825 defines how they may be represented in a binary system.

CCTS data types are not one-dimensional scalars. CCTS defines data types consisting of a Content Components and one or more Supplementary Components that further specify the semantics of the content. E.g. an Amount has a number as content and a currency code as Supplementary Component to specify the currency. This mechanism resembles closely the "Semantic Values" as proposed by Sciore [37] and Lee [38].

Business Entities are identified from the top down: the most generic "Business Concept" (representing the set of all business entities) forms the basis and all specific business concepts and business entities are derived by narrowing its property space. Data types however are defined bottom-up. At the bottom a number of (pre-standardized) scales are defined, such as the set of numbers, the set of texts and the set of date-time combinations. By constraining these scales by means of facets and by combining them, the specific data types are being defined.

Facets are constraints that are specific to the scale. Some scales are ordered (such as the numeric and the date-time scale); others are not (such as the text scale). Only for ordered scales minimum and maximum values can be specified. Precision may be specified for numerics in a straightforward way, for dates and times precision is fairly complex.

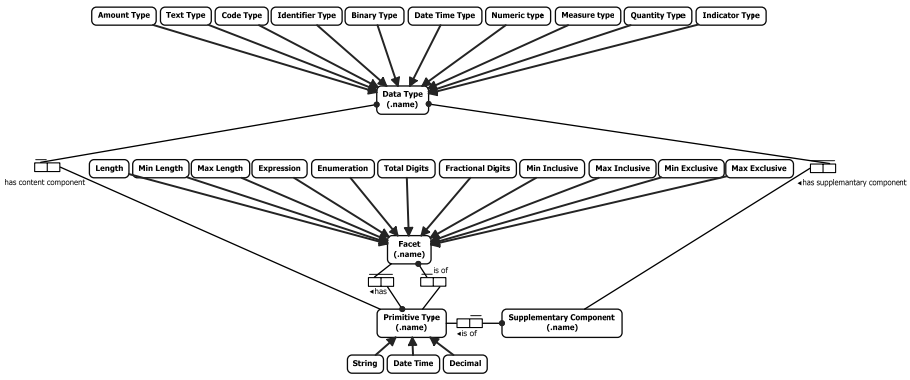


Figure 6.6 Data-type meta-model

The basic scales that are needed for the CCTS v.2.01 data types are the textual scale, numeric scale and the date-time scale. These scales map (not coincidental) neatly to the data type system of XML Schema. The basic scales are called "Primitive types" in CCTS.

Primitive Type	Format Restrictions or Facets	Definition
String	Expression	Defines the set of characters that can be used at a particular position in a string.
	Length	Defines the required length of the string.
	Minimum Length	Defines the minimum length of the string.
	Maximum Length	Defines the maximum length of the string.
	Enumeration	Defines the exhaustive list of allowed values.
Decimal	Total Digits	Defines the maximum number of digits to be used.
	Fractional Digits	Defines the maximum number of fractional digits to be used.
	Minimum Inclusive	Defines the lower limit of the range of allowed values. The lower limit is also an allowed value.
	Maximum Inclusive	Defines the upper limit of the range of allowed values. The upper limit is also an allowed value.
	Minimum Exclusive	Defines the lower limit of the range of allowed values. The lower limit is no allowed value.
	Maximum Exclusive	Defines the upper limit of the range of allowed values. The upper limit is no allowed value.
Date-Time	Minimum Inclusive	Defines the lower limit of the range of allowed dates. The lower limit is also an allowed date.
	Maximum Inclusive	Defines the upper limit of the range of allowed dates. The upper limit is also an allowed date.
	Minimum Exclusive	Defines the lower limit of the range of allowed dates. The lower limit is no allowed date.
	Maximum Exclusive	Defines the upper limit of the range of allowed dates. The upper limit is no allowed date.

Table 6.22 Primitive types

Based on the three Primitive Types ten Core Data Types are defined in CCTS. This set is however extensible and in CCTS version 3.0 the maintenance of the Core Data Type list is assigned to a special committee within UN/CEFACT.

CCT Dictionary Entry Name	Definition	CCT Components
Amount. Type	A number of monetary units specified in a currency where the unit of currency is explicit or implied.	Amount. Content (Decimal) Amount Currency. Identifier (String) Amount Currency. Code List Version. Identifier (String)
Binary Object. Type	A set of finite-length sequences of binary octets.	Binary Object. Content (String) Binary Object. Format. Text (String) Binary Object. Mime. Code (String) Binary Object. Encoding. Code (String) Binary Object. Character Set. Code (String) Binary Object. Uniform Resource. Identifier (String) Binary Object. Filename. Text (String)
Code. Type	A character string (letters, figures or symbols) that for brevity and/or language independence may be used to represent or replace a definitive value or text of an *Attribute* together with relevant supplementary information.	Code. Content (String) Code List. Identifier (String) Code List. Agency. Identifier (String) Code List. Agency Name. Text (String) Code List. Name. Text (String) Code List. Version. Identifier (String) Code. Name. Text (String) Language. Identifier (String) Code List. Uniform Resource. Identifier (String) Code List Scheme. Uniform Resource. Identifier (String)
Date Time. Type	A particular point in the progression of time together with relevant supplementary information.	Date Time. Content (Date-Time) Date Time. Format. Text (String)
Identifier. Type	A character string to identify and distinguish uniquely, one instance of an object in an identification scheme from all other objects in the same scheme together with relevant supplementary information.	Identifier. Content (String) Identification Scheme. Identifier (String) Identification Scheme. Name. Text (String) Identification Scheme Agency. Identifier (String) Identification Scheme. Agency Name. Text (String) Identification Scheme. Version. Identifier (String) Identification Scheme Data. Uniform Resource. Identifier (String) Identification Scheme. Uniform Resource. Identifier (String)
Indicator. Type	A list of two mutually exclusive Boolean values that express the only possible states of a *Property*.	Indicator. Content (String) Indicator. Format. Text (String)
Measure. Type	A numeric value determined by measuring an object along with the specified unit of measure.	Measure. Content (Decimal) Measure Unit. Code (String) Measure Unit. Code List Version. Identifier (String)
Numeric. Type	Numeric information that is assigned or is determined by calculation, counting, or sequencing. It does not require a unit of quantity or unit of measure.	Numeric. Content (Decimal) Numeric. Format. Text (String)
Quantity. Type	A counted number of non-monetary units possibly including fractions.	Quantity. Content (Decimal) Quantity. Unit. Code (String) Quantity Unit. Code List. Identifier (String) Quantity Unit. Code List Agency. Identifier (String) Quantity Unit. Code List Agency Name. Text (String)
Text. Type	A character string (i.e. a finite set of characters) generally in the form of words of a language.	Text. Content (String) Language. Identifier (String) Language. Locale. Identifier (String)

Table 6.23 Core Data Types

In fact many more scales (Core Data Types or even Primitive Data Types) could be defined, such as a colour scale, a location scale, a taste scale, etc. For e.g. locations, many different representations exist (postal, geographical, official, etc.). Each of them has a different structure (number-pairs, codes, and text blocks). It seems therefore more

feasible for an open B2B system to build forward on the three mentioned Primitive Types (possibly extended with a “Binary Large Object” or “BLOB”), by defining Core Data Types and specialize those in Business Data Types.

A special remark must be made on the ‘Code’ data type. A code is in fact some kind of shorthand to indicate a more complex concept or entity. By referring to a concept or entity with a code one does not use the structured definition in the form of property structures. Definitions for code values are made off-line in only human readable (not machine readable) form. Codes are very suitable for very generic concepts, such as countries, measure units and time zones. Code lists are however not extensible in the way as described in this thesis. For concepts that need to be customizable or extensible, such as delivery conditions, container sizes or transport means it would be better to construct full definitions that result in (extensible) property sets.

Summarizing, the information stored in a B2B knowledge base is to be represented in information systems (automated or not), documents (electronic or not) and in human minds. Representation of complex concepts and entities is not made directly. Entities are represented by their atomic properties or attributes. Attributes are projected on some scale or value space. Depending on the technical context, the value space is coded in a character based system and ultimately in a binary system.

The CCTS specification offers a sound base for data typing of business concepts. The CCTS data type system is layered and rich: supplementary components offer additional semantics to the data content. Facets allow for format restrictions as semantics are narrowed.

Sometimes it is needed to include formulas instead of values in the definition of the value space of a data type. A formula may be a Regular Expression, but it may also refer to the contents of other data fields.

6.6 Example

The tabular format can be illustrated by means of a simple example in which various features of the knowledge base are shown. The example shows a basic trade transaction. In the example, shown in tables 6.24 through 6.30 is illustrated how commercial negotiation may be supported by a B2B knowledge base.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
0	0	0:00	B	Add	Def	Assert	Concept	Role	Verb	Role	Concept		0	n	B, S	Def	any		
20	0	0:10	B	Prop	Def	Assert	Delivery		Deliver	Product_ Theme	ID	1	1	1	B, S	Per	Request, Offer		
21	0	0:10	B	Prop	Def	Assert	Delivery		Deliver	End_ Time	Date	1	1	1	B, S	Per	Request, Offer		
22	0	0:10	B	Prop	Exp	Assert	Delivery		Deliver	Start_ Time	Date		0	1	B, S	Per	Request, Offer		
23	0	0:10	B	Prop	Exp	Assert	Delivery		Deliver	Price_ Condition	Amount		0	1	B, S	Per	Request, Offer		
24	0	0:10	B	Prop	Def	Assert	Delivery		Deliver	Quantity_ Theme	Measure		0	1	B, S	Per	Request, Offer		
25	0	0:10	B	Prop	Exp	Assert	Delivery		Deliver	Actual_ Time	Date		0	1	B, S	Per	Assert		

Table 6.24 Initial definitions

First, in Utterance #20 to #25 the Buyer (col. 4) proposes (col. 5) to define (col. 6) a Delivery concept (col. 8). The Delivery concept is defined as a concept (col. 2) that has a certain quantity of a product delivered in a specified period. The Delivery is identified (col. 13) with a Product and an End date in which the delivery must take place.

The Source Role (col. 9) is left empty, as the Delivery concept represents the event that is directly represented by the Verb (col. 10). The Target Roles (col. 11) are specializations of the Verbnet roles, listed in section 5.2, with the addition of a Condition role (Utt. #23).

Utterances #22, #23 and #25 are stereotyped as Expansion rather than as Definition (col. 6). The start date of the delivery period, the actual delivery date and the price are not part of the definition of a Delivery concept. They are additional and conditional (col. 14) properties of a Delivery.

In table 6.25 the Seller (col. 4) accepts (col. 5) the proposed definitions. After acceptance the definitions are valid and may be used in the sequel of the conversation.

Structure of utterances

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
20	0	0:20	S	Acc	Def	Assert	Delivery		Deliver	Product_ Theme	ID	1	1	1	B, S	Per	Request, Offer		
21	0	0:20	S	Acc	Def	Assert	Delivery		Deliver	End_ Time	Date	1	1	1	B, S	Per	Request, Offer		
22	0	0:20	S	Acc	Exp	Assert	Delivery		Deliver	Start_ Time	Date		0	1	B, S	Per	Request, Offer		
23	0	0:20	S	Acc	Exp	Assert	Delivery		Deliver	Price_ Condition	Amount		0	1	B, S	Per	Request, Offer		
24	0	0:20	S	Acc	Def	Assert	Delivery		Deliver	Quantity_ Theme	Measure		0	1	B, S	Per	Request, Offer		
25	0	0:20	S	Acc	Exp	Assert	Delivery		Deliver	Actual_ Time	Date		0	1	B, S	Per	Assert		

Table 6.25 Acceptance of definitions

In Utterance #30 to #34 a Perception (col. 6) is defined in which the Buyer (col. 16) may express his Request for the Delivery. A Perception is the blueprint of a transaction (or business document, in this case an order) that is exchanged in the course of the business process. Utterances #30 to #34 are based on the definitions and expansions in Utterances #20 to #24 (col. 2). Utterances #20 to #25 allow (only) Perceptions to be based on them (col. 17). In Utterances #35 to #39 the potential reaction of a Seller, the offer or order response, is defined as a Perception as well.

Both perceptions are defined as transactions (col. 20): utterances, based on Utt. #31 to #34 may only exist in combination with an utterance based on #30 with the same timestamp. The same is true for utterances, based on #36 to #39 in combination with an utterance based on #35.

Utterances, based on Utt. #30 and #31 may be stereotyped as Instantiation (col. 17). Utt. #30 and #31 are based on Utt. #20 and #21, which are identifying attributes of a Delivery. After instantiation of a Delivery, Observations may be based on Utt. #30 and #31. Utterances #32 to #34 only allow Observations to be based on them (col. 17). The Observations form one transaction with the instantiation (col. 20). Note that in such transaction, utterances, based on Utt.#30, #31 and #34 always must be present. Utterances based on Utt.#32 and #33 are not always required.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Sourc e Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
30	20	0:30	B	Prop	Per	Assert	Delivery		Deliver	Product_ Theme	ID	1	1	1	B	Inst, Obs	Request		31
31	21	0:30	B	Prop	Per	Assert	Delivery		Deliver	End_ Time	Date	1	1	1	B	Inst, Obs	Request		34
32	22	0:30	B	Prop	Per	Assert	Delivery		Deliver	Start_ Time	Date		0	1	B	Obs	Request		31
33	23	0:30	B	Prop	Per	Assert	Delivery		Deliver	Price_ Condition	Amount		0	1	B	Obs	Request		31
34	24	0:30	B	Prop	Per	Assert	Delivery		Deliver	Quantity_ Theme	Measure		1	1	B	Obs	Request		30
35	20	0:30	B	Prop	Per	Assert	Delivery		Deliver	Product_ Theme	ID	1	1	1	S	Obs	Offer	30	36
36	21	0:30	B	Prop	Per	Assert	Delivery		Deliver	End_ Time	Date	1	1	1	S	Obs	Offer	31	39
37	22	0:30	B	Prop	Per	Assert	Delivery		Deliver	Start_ Time	Date		0	1	S	Obs	Offer		36
38	23	0:30	B	Prop	Per	Assert	Delivery		Deliver	Price_ Condition	Amount		0	1	S	Obs	Offer		36
39	24	0:30	B	Prop	Per	Assert	Delivery		Deliver	Quantity_ Theme	Measure		1	1	S	Obs	Offer		35

Table 6.26 Perceptions

An offered Delivery (Utt #35 to #39) may not be instantiated, but must be an Observation on a previously instantiated requested Delivery (col. 17). An Observation must follow an Instantiation, so the Seller may not offer unsolicited. The Seller may propose to change some of the delivery data, such as the start date of the delivery period, the quantity and the price of a requested delivery (Utt. #37 to #39). Utterances #30 to #34 allow the Buyer to react on the Sellers offer by changing his request: column 17 contains not only Ins(tantiation), but also Obs(ervation) as allowed stereotype for Delivery attributes.

The Seller then accepts the Buyers' proposals ('Acc' in col. 5, similar to Utt. #20 to #25), but for brevity this is not shown in the table.

Structure of utterances

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
40	33	0:50	B	Add	Sta	Assert	Agreed_ Delivery		Deliver	Price_ Condition	R_ Amount		1	1	B	Obs	Request	41	
41	38	0:50	B	Add	Sta	Assert	Agreed_ Delivery		Deliver	Price_ Condition	<= R_ Amount		1	1	S	Obs	Offer	40, 43	
42	32	0:50	B	Add	Sta	Assert	Agreed_ Delivery		Deliver	Start_ Time	RS_ Date		1	1	B	Obs	Request	43	
43	37	0:50	B	Add	Sta	Assert	Agreed_ Delivery		Deliver	Start_ Time	>= RS_ Date		1	1	S	Obs	Offer	42, 45	
44	34	0:50	B	Add	Sta	Assert	Agreed_ Delivery		Deliver	Quantity_ Theme	R_ Measure		1	1	B	Obs	Request	45	
45	39	0:50	B	Add	Def	Assert	Agreed_ Delivery		Deliver	Quantity_ Theme	R_ Measure		1	1	S	Obs	Offer	44, 41	

Table 6.27 States

In Utt. #40 to #45 a State (col. 6) of a Delivery (Agreed_ Delivery) is defined. Utt. #40 to #45 serve as each other's precondition (col. 19), so they together can be regarded as one State of a Delivery (col. 8). Utt. #40 states the existence of a Delivery (col. 8) instance that is Requested (col. 18) by the Buyer (col. 16), with a Price_ Condition (col. 11) that is called R_ Amount (col. 12). Utt. #41 states the existence of an Offered (col. 18) Delivery of the Seller, of which the Price_ Condition (col. 11) is lower or equal to the price, requested by the Buyer (col. 12). A Delivery is in the Agreed state if and only if the offered price is lower than or equal to the requested price (Utt. #41), the offered delivery period falls within the requested period (Utt. #43; note that the End_ Time is part of the Delivery ID and thus fixed) and the offered quantity is equal to the requested quantity (Utt. #45). Utterances #40 to #45 together contain the conditions that change the state of the delivery from In Negotiation into Agreed (col. 8).

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
50	0	1:00	B	Prop	Def	Assert	Payment		Pay	Amount	Amount		1	1	B	Per	Assert		
51	0	1:00	B	Prop	Def	Assert	Payment		Pay	Beneficiary	Party	1	1	1	B	Per	Assert		
52	0	1:00	B	Prop	Exp	Assert	Payment		Pay	Ultimate_ Time	Date		0	1	B	Per	Assert		
53	0	1:00	B	Prop	Exp	Assert	Payment		Pay	Delivery_ Condition	Delivery	1	1	1	B	Per	Assert		
54	0	1:00	B	Prop	Exp	Assert	Payment		Pay	Actual_ Time	Date		0	1	B	Per	Assert		
60	50	1:05	B	Prop	Per	Assert	Payment		Pay	Amount	[formula 1]		1	1	B	Obs	Commit		61
61	51	1:05	B	Prop	Per	Assert	Payment		Pay	Beneficiary	S	1	1	1	B	Inst	Commit		63
62	52	1:05	B	Prop	Per	Assert	Payment		Pay	Ultimate_ Time	[formula 2]		0	1	B	Obs	Commit		61
63	53	1:05	B	Prop	Per	Assert	Payment		Pay	Delivery_ Condition	Agreed_ Delivery	1	1	1	B	Inst	Commit		61
70	20	1:10	B	Prop	Per	Assert	Agreed_ Delivery		Deliver	Product_ Theme	ID	1	1	1	S	Obs	Assert	63	71
71	21	1:10	B	Prop	Per	Assert	Agreed_ Delivery		Deliver	End_ Time	RE_ Date	1	1	1	S	Obs	Assert		72
72	25	1:10	B	Prop	Per	Assert	Agreed_ Delivery		Deliver	Actual_ Time	Date		1	1	S	Obs	Assert		70
80	51	1:15	B	Prop	Per	Assert	Payment		Pay	Beneficiary	S	1	1	1	B	Obs	Assert		81
81	53	1:15	B	Prop	Per	Assert	Payment		Pay	Delivery_ Condition	Agreed_ Delivery	1	1	1	B	Obs	Assert	72	82
82	54	1:15	B	Prop	Per	Assert	Payment		Pay	Actual_ Time	Date		1	1	B	Obs	Assert		80

formula1: Payment. Pay Delivery_ Condition. Deliver Price_ Condition. Amount * Payment. Pay Delivery_ Condition. Deliver Quantity_ Theme. Measurement
formula2: Payment. Pay Delivery_ Condition. Deliver Date + 30d

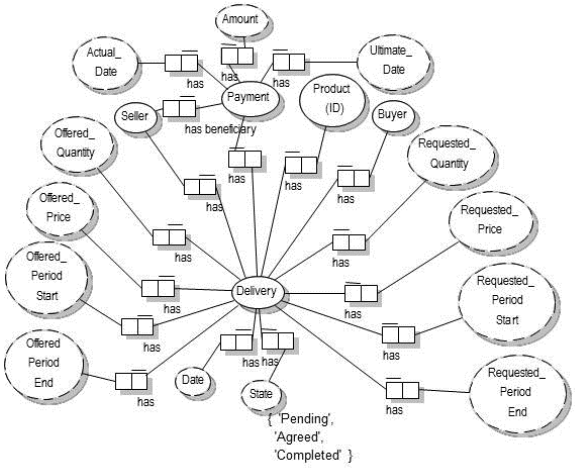
Table 6.28 Perceptions

In Utt. #50 to #54 a Payment event is defined. That definition is the basis for both a commitment to pay (Utt.# 60 to #63) and for the actual payment (Utt.# 80 to #82). In Utt. #60 to #63 a Perception (col. 17) is defined of the commitment (col. 18) of the Buyer (col. 16) to pay. Payment may be committed by the Buyer only if the associated Delivery is in the Agreed state (Utt. #63 col. 12). The amount and the date of the payment are calculated from the Delivery information (col. 12 of Utt. #60 and #62). These calculations serve as the value space of those Payment attributes.

Utterances #70 to #72 define the perception in which the Seller asserts that delivery has taken place (equivalent to a Despatch Advice). This perception has as precondition the commitment of the Buyer to pay (col. 19). Utterances #80 to #82 define an assertion of the Buyer that he has paid (equivalent to a Remittance Advice). This perception has been made conditional to the Delivery having taken place.

Summarizing, in tables 6.24 through 6.28 a process has been defined in which a Buyer may request for a delivery of some product and a Seller may offer that delivery. As soon

as price, quantity and delivery date are agreed upon, the Buyer may commit payment on the condition that the Seller actually delivers. After actual delivery the Buyer may pay.



For each Delivery
State='Agreed' if and only if
(Offered_ Quantity = Requested_ Quantity and
Offered_ Period Start ≥ Requested_ Period Start and
Offered_ Period End ≤ Requested_ Period End and
Offered_ Price ≤ Requested_ Price)
If Date exists then State = 'Agreed'

For each Payment
If Ultimate_ Date exists then
(Delivery. State = 'Agreed' and
Ultimate_ Date = 'Delivery. Date + 30d)
Amount = Delivery. Offered_ Price * Offered_ Quantity

Figure 6.7 ORM model

In figure 6.7 an ORM model of these metadata and conditions is shown. This model is a simplified version of the model shown in [39]. In [39], the use of dynamic constraints in ORM is illustrated with the negotiation process shown here as an example.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	30	6:00	B	Add	Inst	Request	Delivery		Deliver	Product_ Theme	456								
101	31	6:00	B	Add	Inst	Request	Delivery		Deliver	End_ Time	2010-06-30								
102	32	6:00	B	Add	Obs	Request	Delivery		Deliver	Start_ Time	2010-06-15								
103	33	6:00	B	Add	Obs	Request	Delivery		Deliver	Price_ Condition	EUR 10								
104	34	6:00	B	Add	Obs	Request	Delivery		Deliver	Quantity_ Theme	100 PCE								
105	35	6:10	S	Add	Obs	Offer	Delivery		Deliver	Product_ Theme	456								
106	36	6:10	S	Add	Obs	Offer	Delivery		Deliver	End_ Time	2010-06-30								
107	38	6:10	S	Add	Obs	Offer	Delivery		Deliver	Price_ Condition	EUR 9								
108	39	6:10	S	Add	Obs	Offer	Delivery		Deliver	Quantity_ Theme	150 PCE								
110	30	6:20	B	Add	Obs	Request	Delivery		Deliver	Product_ Theme	456								
111	31	6:20	B	Add	Obs	Request	Delivery		Deliver	End_ Time	2010-06-30								
112	34	6:20	B	Add	Obs	Request	Delivery		Deliver	Quantity_ Theme	150 PCE								

Table 6.29 Delivery instantiation

Now the concepts and perceptions have been defined, actual business may start. In Utterances #100 to #104 the Buyer (col. 4) instantiates a delivery request. The utterances are based on (col. 2) the perceptions defined in Utt. #30 to #34. The Seller reacts in

Utterances #105 to #108 with a counterproposal, increasing the quantity (Utt. #108) and lowering the price (Utt. #107). In Utterance #110 to #112 the buyer accepts the proposal by increasing his requested quantity to equal the quantity in the counterproposal.

At this moment the test of the state defined by Utterances #40 to #43 evaluates True. As a result the Buyer is afforded to commit payment, which he does in Utterance #115 to #120. Note that the term 'Delivery' in col. 12 of Utt. #118 refers to the Delivery instance that is defined in the same transaction (identical timestamps in col. 3) in Utt. #119 and #120. In Utterance #125 to #127 the Seller reports he has delivered by asserting an actual delivery time and in Utterance #120 the Buyer states he has paid.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
115	60	6:30	B	Add	Inst	Commit	Payment		Pay	Amount	EUR 1350								
116	61	6:30	B	Add	Inst	Commit	Payment		Pay	Beneficiary	S								
117	62	6:30	B	Add	Obs	Commit	Payment		Pay	Ultimate_ Time	[formula 2]								
118	63	6:30	B	Add	Obs	Commit	Payment		Pay	Delivery_ Condition	Delivery								
119	30	6:30	B	Add	Obs	Request	Delivery		Deliver	Product_ Theme	456								
120	31	6:30	B	Add	Obs	Request	Delivery		Deliver	End_ Time	2010-06-30								
125	70	6:40	S	Add	Obs	Assert	Agreed_ Delivery		Deliver	Product_ Theme	456								
126	71	6:40	S	Add	Obs	Assert	Agreed_ Delivery		Deliver	End_ Time	2010-06-30								
127	72	6:40	S	Add	Obs	Assert	Agreed_ Delivery		Deliver	Actual_ Time	2010-06-18								
130	80	6:50	B	Add	Obs	Assert	Payment		Pay	Beneficiary	S								
131	81	6:50	B	Add	Obs	Assert	Payment		Pay	Delivery_ Condition	Agreed_ Delivery								
132	30	6:50	B	Add	Obs	Request	Agreed_ Delivery		Deliver	Product_ Theme	456								
133	31	6:50	B	Add	Obs	Request	Agreed_ Delivery		Deliver	End_ Time	2010-06-30								
134	82	6:50	B	Add	Obs	Assert	Payment		Pay	Actual_ Time	2010-07-15								

formula 2: Payment. Pay Delivery_ Condition. Deliver Actual_ Time + 30d

Table 6.30 Payment and actual delivery

This example illustrates how a B2B knowledge base is populated in the course of a business conversation. Chapter 12 shows more elaborate examples in which more features of the knowledge base structure are illustrated.

In this chapter the structure of a B2B knowledge base was derived from general characteristics of business communication, taking into account requirements as stated in chapter 4. In chapter 7 the capability of the structure to define process flows is elaborated. In chapter 8 the structure is mapped to a number of widely used modelling- and data exchange languages.

References

[1]. Chomsky: (1965). Aspects of the Theory of Syntax. MIT Press
[2]. Montague (1970b). Universal grammar. Theoria 36:373-398.
[3]. Van Benthem et al (2010). Handbook of Logic and Language (2nd ed.). Elsevier
[4]. Fillmore: Frame Semantics for Text Understanding. Proceedings of NAACL 2001, WordNet and Other Lexical Resources Workshop, Pittsburgh, Pennsylvania June 2001
[5]. Zamenhof: Dr. Esperanto's International Tongue, Ch. Kelter, Warsaw, 1888
[6]. Sowa, Knowledge Representation: Brooks Cole Publishing Co., Pacific Grove, CA, 2000.
[7]. Bennett et al: Toward a Computational Model of Aspect and Verb Semantics, Machine Translation, 1990: 247-280.
[8]. Gruber: (1965), Studies in lexical relations, Massachusetts Institute of Technology
[9]. Verbnet: http://verbs.colorado.edu/~mpalmer/projects/verbnet.html retrieved 2011-09-03
[10]. Levin: English Verb Classes and Alternations: A Preliminary Investigation, The University of Chicago 1993
[11]. Baker et al: The Berkeley framenet project, COLING 1998 Proceedings of the 17th international conference on Computational linguistics
[12]. Martha Palmer, Dan Gildea, Paul Kingsbury: The Proposition Bank: A Corpus Annotated with Semantic Roles, Computational Linguistics Journal, 31:1, 2005
[13]. Matsubayashi et al: A Comparative Study on Generalization of Semantic Roles in FrameNet. Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP.
[14]. Verbnet Thematic roles: http://verbs.colorado.edu/~mpalmer/projects/verbnet.html, retrieved 2011-09-03
[15]. http://www.jfsowa.com/ontology/thematic.htm retrieved 2011-09-03
[16]. Berners-Lee et al:. "The Semantic Web". Scientific American Magazine. http://www.sciam.com/article.cfm?id=the-semantic-web&print=true. retrieved March 26, 2008.
[17]. RDF: http://www.w3.org/RDF/, referenced 2011-09-03
[18]. OWL: http://www.w3.org/TR/owl-features/, retrieved 2011-09-03
[19]. http://www.bpiresearch.com/Resources/RE_OSSOnt/re_ossont.htm , retrieved 2011-09-03
[20]. http://www.jfsowa.com/cg/, retrieved 2011-09-03
[21]. ISO/IEC 24707:2007 Information technology -- Common Logic (CL): a framework for a family of logic-based languages
[22] SBVR: http://www.omg.org/spec/SBVR/1.0/PDF/, retrieved 2011-09-03
[23]. Kimbrough: A Note on Getting Started with FLBC - Towards a User Guide for FLBC, Newsletter ACM SIGGROUP Bulletin, Volume 22 Issue 2, August 2001
[24]. Weigand: An extensible business communication language. International Journal of Cooperative Information Systems, 10(4), 2001.
[25]. Moore: Categorizing automated messages, Journal of Decision Support Systems, Volume 22 Issue 3, March 1998
[26]. Wieringa: Design methods for reactive systems. Morgan Kaufmann, 2003.
[27]. UN/ECE: Core Component Technical Specification, http://www.unece.org/fileadmin/DAM/cefact/codesfortrade/CCTS/CCTS_V2-01_Final.pdf, retrieved 2011-10-20.
[28]. International Standardisation Organisation: ISO11179, Information technology - Metadata registries, 2005.
[29]. Searle: Searle on conversation, in: Pragmatics and beyond, John Benjamins Publishing Company. 1992.
[30]. Object Management Group: Meta Object Facility, http://www.omg.org/spec/MOF/2.4/Beta2, retrieved 2011-10-20.
[31]. OAGis: Business Object Document Message Architecture, http://www.oagi.org/oagis/9.0/Documentation/Architecture.html, retrieved 2011-10-20.
[32]. UN/CEFACT: Core Components Message Assembly Technical Specification, Draft, 2007.
[33]. Renssen: Gellisch: een nieuwe standaardtaal voor mensen en computers, PhD thesis, Delft university, 2005.

[34] Object Management Group: Object Constraint Language, http://www.omg.org/spec/OCL/2.2/PDF/, retrieved 2011-11-12.
[35]. W3C: XML Schema Part 2: Datatypes Second Edition, 2004, http://www.w3.org/TR/xmlschema-2/, retrieved 2011-10-20.
[36]. International Telecommunications Union: X.680-683, Abstract Syntax Notation 1, 2002, http://www.itu.int/ITU-T/studygroups/com17/languages/X.680-0207.pdf, retrieved 2011-10-20.
[37]. Sciore: Using Semantic Values to Facilitate Interoperability Among Heterogeneous Information Systems, ACM Transactions on Database Systems, Vol. 19, No 2, June 1994.
[38]. Lee: Context-dependent Semantic Values for E-negotiation, Proceedings of WECWIS, 2000
[39]. Balsters, van Blommestein: ORM-Based Semantics of B2B Transactions. OTM Workshops 2009

7 B2B Process control

Summary

Traditionally, in B2B systems, the process flow, or the sequence in which information is exchanged, is defined separated from the message structures that contain the information. It is then difficult to define the interdependencies between process flow and information content. In this chapter it is proposed to define the process flow by means of adding preconditions to the information, or the utterances, that is (are) exchanged.

Process flows are defined as sequences of states of concept instances. The allowed transitions between the states are defined as preconditions of the resulting states.

It is shown that with state orientation, the information can be better integrated with the process, and processes and process parts may be defined more modular and reusable. It is also shown that the utterance structure as presented in chapter 6 allowes to define the process flow and to monitor process instances against those definitions.

7.1 Introduction

In chapter 6 a mechanism was presented to fill a B2B knowledge base with knowledge about concepts and entities that are relevant to a B2B relationship. Although the knowledge base is filled in the course of the business process, it contains relatively static information. The sequence in which knowledge is added is not prescribed. The objective of a B2B process however is to coordinate activities of business partners in order for them to reach agreed goals. Sequence of activities is essential in such process. E.g. delivery should only occur after an order has been placed and payment should follow delivery.

In this chapter the mechanism to populate the knowledge base is extended with process definitions and process monitoring. Traditionally, process definitions are agreed on at "design time", before actual trading begins. Systems are customized for certain predefined processes. In a normal trading relationship, however, processes are agreed during trading, and they may change in the course of the trading relation.

The research challenge is to combine process- and information flow definitions in one consistent model. The model should be derived from the requirements that stem from business models and business policy. The model should be maintained in the B2B knowledge base, and be directly used by middleware and application systems that monitor the communication processes and process the data exchanged. Message schemas and application interfaces can then be generated from it.

Most business process languages (including UML activity diagrams and BPMN) are activity oriented: they describe the sequence of activities that may or must take place

(including choices, iterations and parallelism). These languages suffer from a number of flaws that make them less suitable for dynamic B2B relationships:

- most activity oriented languages have a weak binding with the information that is exchanged
- activity oriented languages are hard to manipulate, the negotiation of a business process is often a huge combinatorial challenge
- activity oriented models cannot easily be modularized
- most activity oriented languages are based on the "case" paradigm, while many business relations are more complex than that.

These issues are elaborated on below.

In this chapter it is shown that a state oriented language does not have these disadvantages. Moreover, state orientation aligns neatly with the knowledge base structure as described in chapter 6. The state oriented language presented in this chapter support all relevant process patterns activity oriented languages can describe.

It is argued in this chapter that for the purpose of transactional business systems, the mechanism defined here can be used as a basis for standardisation of business semantics. The mechanism allows the decoupling of business semantics and technology, enabling smooth transitions in upgrading the technology (e.g. from EDI systems to Web services) without affecting the business processes supported.

As stated in chapter 1, traditionally the focus of Electronic Data Interchange systems has been totally on the definition of electronic messages and the information contained therein. Some years ago several initiatives (Open-edi, OO-edi) have advocated the definition of the inter-organizational business processes, rather than of the data, as the basis for interface specification. These initiatives have resulted in a number of standard specifications, most notably ISO/IEC 14662 (open-edi Reference Model) [1] and UMM (UN/CEFACT Modelling Methodology) [2]. With the advent of XML [3] and the ebXML project [4] these specifications have been further elaborated on, what resulted in the ebXML Business Process Specification Schema (BPSS) [5].

The way most (inter-organizational) workflow systems and most EDI or XML peer-to-peer systems work (if they support process management at all) is that in some process definition language the sequence is defined of the message types that may be exchanged. The sequence may include forks, parallel paths and iteration, but the most fine-grained element is the message type. An example shows that in even the most basic B2B scenario this approach falls short.

Consider an order change process in which a customer may change the Purchase Order sent previously to a supplier. The Purchase Order defines a delivery date and a number of order lines, each with a product number, a quantity and a price. The order change procedure allows the customer to change the delivery date (which needs confirmation of the supplier) or to add, to change or to delete order lines. Furthermore the customer is not allowed to change prices of existing order lines.

From this example it becomes clear that a single process step “Send Order Change Message” does not define sufficiently how the process continues. If the delivery date was changed, a confirmation must be expected by the customer. If all orderlines of an order were deleted, the process stops. In all other cases a delivery takes place. The business process monitor therefore needs to inspect the contents of the order change message in order to know what next step is to be set. Furthermore, it should compare the content of the message with the data already present in order to know whether a line is added, changed or deleted. In addition it should refuse a message with a changed price in an already existing line.

The granularity of the process should therefore be determined by the transactions on the data of individual business objects (e.g. order lines) rather than by complete messages. In a business relationship one should be able to indicate which transactions are allowed, given the existing state (preconditions) and which update results are valid (post-conditions). Moreover, it should be possible to indicate how updates may or should be combined to atomic transactions.

An alternative could be to define many different message types, e.g. one for each different combination of data that triggers different future actions. Given the fact that typical trade messages may contain many different information entities, the number of message types then would explode. And even if that would be acceptable, operations such as splitting orders or defining multiple deliveries in an order anyway would make inspection of the data content necessary.

Present inter-organizational process definition methods and languages (such as BPSS [5], BPMN [6] or BPEL [7]) do not meet two other sets of requirements. First, a process model should fit with (or be derived from) business objectives and goals. Business goals are stated in terms of results (the “what”), not in terms of activities (the “how”). Present process languages mainly define the “how”. Even in UMM [2], the derivation of the process flow from the business objectives is performed during the Business Requirements phase and is informal and descriptive.

A second requirement is modularity. In order to be used by many industry sectors and organizations world-wide, process models must be easily adaptable, reusable, refinable and be layered in abstraction levels. Although process modelling languages, describing graphs of activities, often allow sub processes to be defined (thus creating layers), real modularity is not supported. Activities in process models typically have names and informal definitions. One series of activities cannot easily be replaced by another series of activities in a part of the process, as without knowing what the activities accomplished in the real world, it cannot be determined if the replacing series makes sense. Most workflow languages abstract from defining the (ontological) accomplishments of activities.

The standardisation of business meta-information on a semantic level is now well underway [8]. Within the UN/CEFACT organization some 15 groups, each representing a business sector, are creating data models and message schemas. The models are being

harmonized by a central committee before they are published in the UN/CEFACT library as “Core Components” (data model building blocks). No equivalent effort exists for process flow standardization. “Process Components” as a dynamic equivalent of Core Components have not been defined. A "Common Business Process Catalog" exists, but is not used.

The reference model in chapter 5 makes clear that communication synchronizes knowledge about events that have happened. An interface specification prescribes what events are allowed to happen. The interface serves as a filter. It only allows messages or information to pass that fits in the predefined conversation and that reports on permitted events. Parties are themselves responsible for the happening of the events. The interface cannot force them to take action, only to remind them to do so.

An event changes properties of one or more business objects or business entities. The interface definition must therefore define the data that may change per event per business entity. In this chapter we shall investigate what language may be used to describe such an interface.

A business interface can be described on three levels. On the highest level the interface is analysed in an abstract way. It is investigated how trade is conducted, irrespective of the technology used, be it based on paper documents, on verbal communication over the phone, on ‘e-shops’ on the web or on interconnected ERP systems.

On the second level it is attempted to define a specific business interface in a (semi) formal language that can be executed by some automatic system. This level is directed towards the use of information technology, but still largely technology neutral. The interface description should be implementable in an EDI environment, in Web services, in peer-to-peer XML messaging or in interconnected workflow systems. This level aligns with the Business Operational View in ISO/IEC 14662.

The lowest level defines the implementation in one of the technologies. In chapter 11 we describe an implementation of the model in peer-to-peer XML messaging. In chapter 13 we describe an implementation in a stand-alone Internet workstation.

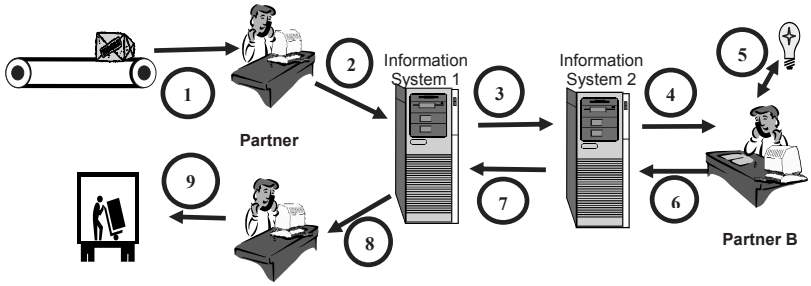
The e-business reference model in chapter 5 departed from an empirical model of how business people interact. In chapter 6 the static part of this model (the ‘Universe of Discourse’) was translated into semi-formal specifications that can be represented in computer readable languages. The dynamics of business interrelation were however not yet elaborated. Modelling the dynamics is the topic of this chapter.

In sections 7.2 through 7.4 B2B processes are analysed. We take one step back to the abstract model and enhance it to define the dynamics of inter-organizational business. Afterwards, in section 7.5 we will derive requirements for modelling languages to use to define business interfaces. A number of candidate languages to support such modelling will be assessed in sections 7.6 to 7.8. A solution will be chosen and designed in section 7.9, which will be assessed in sections 7.10 and 7.11. The chapter ends with an example.

7.2 Processes as conversations

In chapter 5 a high level e-business architecture was defined. We need such architecture, to identify and position the components of the specification of the interface between organizations. The interface specification is not only the basis for the application systems and middleware that support the interrelationship; it also defines the mutual understanding of the legal, commercial and operational commitments, obligations, expectations and acts at any moment during the business process.

When two organizations collaborate, their information systems' view on the state of affairs (on the events that took place) should be synchronised, at least with regard to the information that is relevant to the collaboration. Information on physical events, decisions (speech acts) and contextual conditions together define the commitments, obligations and expectations of the parties in a business collaboration. This process is illustrated in figure 7.1.



Transaction 1	Transaction 2
1. Event happens 2. Partner A records event in local information system 3. Information in both information systems is synchronised 4. Partner B can act on new information	5. Partner B takes decision 6. Partner B records the results of his decision 7. Information systems are synchronised 8. Partner A acts, triggered by new information 9. Physical action is started

Figure 7.1 Process and knowledge synchronisation

The synchronization mechanism must be envisaged on another view level than the event registration. In practice the synchronization can be performed (see section 2.3):

- By using the same information system,
- by keeping databases in sync on (database-)transaction level,
- by exchanging EDI or XML messages,
- by letting one system interrogate the other
- by means of Web services
- by exchanging paper documents and letting administrative clerks handle the input and output.

In all cases the same business process can take place. Note that a scenario that defines the process of document- or EDI-message-exchange is in fact defining the synchronization, not directly the sequence of events that need to take place (see figure 7.2). When specifying the interface, by shifting the focus from the messages that are exchanged to the events that happen, the specification is made technology neutral. We then avoid the pollution of the specification with technology-oriented aspects, allowing discussions and negotiations to deal with the business aspects only.

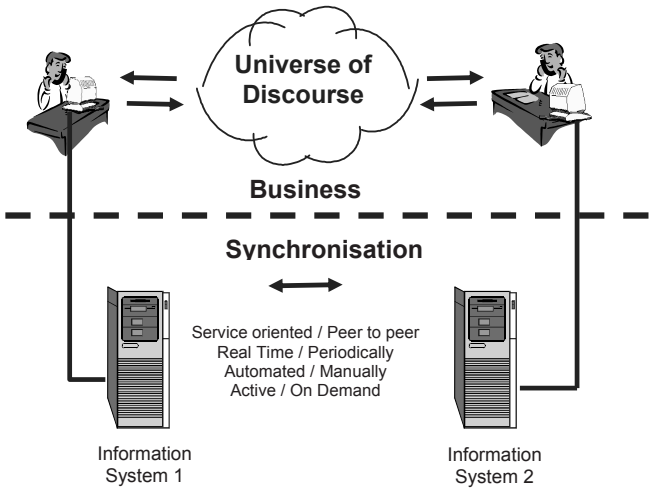


Figure 7.2 Process versus Synchronisation

If we abstract from the synchronization mechanism and concentrate on the process itself, the reference model can be limited to one (virtual) information system in which events are recorded. Each event takes place or is reported under the responsibility of one of the partners. The information system reflects the situation after each event. The other partner's next step (decision or physical activity, causing a new event) is dependent on that situation.

The information system contains a model of the real world, which contains the aspects of the world that are relevant to the business relation. This model, which can be represented as a database schema, we call the “Universe of Discourse”.

As all relevant (physical) events are recorded in the Universe of Discourse, the business process boils down to a process in which the parties are communicating with each other through the system.

Communication between people is analysed in Conversation Analysis. Conversations appear to follow certain patterns. In a conversation also various levels can be discovered.

If we are to support conversations by a system that not just relays the utterances, but also interprets them, these levels need to be taken into account.

Few authors have used conversation analysis to design business communication systems. Medina Mora has designed the "Action Workflow" pattern by analysing conversations within an organization [10]. Van Reijswoud, Steuten and Dietz have developed a transaction pattern as part of the DEMO method [14]. In that pattern grounding and argumentation are explicitly layered. The pattern is targeted at concluding a business transaction. Business conversations are divided in three phases: the actagenic phase (in which the conditions for the transaction are negotiated and the commitments are expressed), the phase of objective action, in which the physical actions take place and the factagenic phase in which partners agree on those facts.

These attempts to describe business processes from a Language Action Perspective contain very valuable insights. Yet the presented patterns only seem to support either a straightforward command structure within a company or simple trade transaction agreements and fulfilments between companies. Business practice unfortunately is far more complex. A system to support business communication must also be able to support e.g. complex logistical control, governmental services and law enforcement, medical services and insurance, etc. Moreover, it must be possible to renegotiate process flows based on e.g. altered trust levels through the system. In chapter 9 we shall further elaborate such more complex patterns in relation to business objects and ontologies. Here we will investigate how human business conversation can be captured in modelling languages.

One aspect is still worth to be mentioned. Many of the LAP approaches rightfully make a distinction between Speech Acts and Instrumental Acts (real life behaviour). In our meta-model we assume that all instrumental acts are being reported to the information system by means of speech acts, so we may abstract from instrumental acts as such.

Bollen [15] has studied enterprise systems and concludes that a way to consistently describe both data and process aspects of such systems is to define preconditions, post-conditions and derivation rules for each process step. His scope is however a (designed) enterprise system, not B2B communication.

Human conversations seldom adhere to a strict pattern. In some environments the conversation is highly structured (e.g. in aircraft pilot protocols or call-centre scripts), but in most environments frequent interruptions, expressions of doubt, requests for clarifications, etc. occur. It seems not possible to model all exceptions to the standard pattern. Like a chess game, it seems more feasible to define rules (behavioural space of the participants given the state of the conversation) than to elaborate all possible sequences of moves. That is one of the reasons why inter-organizational business processes should be modularized. With a limited number of modules ('chess pieces') and a limited set of rules a very rich set of conversations may be defined.

State changes

As stated in chapter 5, the real world can be envisaged as consisting of objects. Objects can be physical (e.g. some product) or non-physical (e.g. some commitment). Objects have properties that may change value in the course of the business process. Properties may be simple pieces of data (“Number”, “Text”, “Date”) or be complex (other objects).

Objects can be in a State. A State is a situation in the life time of an object during which it satisfies some condition. [16]. The conditions that define a state of an object are defined on the values of the objects properties, as recorded in the Universe of Discourse.

An example is a delivery event that may be proposed (delivery event exists), accepted, planned (planned delivery date exists and lies in the future) and completed (actual delivery date exists).

When events are reported by the participants, some property values change. So events may change the state of an object. The other way around it is possible to define the effects of an event by identifying the state changes the event brings about. As all we are interested in (and are able to measure) are the (side) effects of events, events themselves can be defined as the state changes they cause to the objects we are interested in. Therefore it should be possible to define the sequence of events that are allowed to happen as a sequence of state changes of objects, defined in the Universe of Discourse.

The sequence of states one object is going through is called the object lifecycle. For a class of objects it is possible to define the possible sequences that the instances of the class may experience. That sequence (which may include forks and joins) is called the lifecycle of the class.

The value space of an object is defined as the Cartesian product of all possible property values. Conditions may constraint the value space. A state is an area within that space.

Events may cause several objects to change state. Lifecycles are bound to one (class of) object(s). Lifecycles of various objects therefore may be correlated. State changes of different object types that must happen simultaneously are called transactions. Transactions, like state changes, may be defined as a transition from a set of states (the precondition of the transaction) to another set of states (the post-condition).

We can conclude that each activity causes a change of the state of one or more objects. In defining a business interface an activity can be defined as a state change to one or more object classes. Each object class state change can be defined by stating the preconditions (the original state) and the post-condition (the resulting state). The activity can therefore be defined by a collection of pre-conditions and post-conditions on the objects that are affected by the activity.

7.3 Goals and information requirements

In this section we investigate how activities can be derived from business requirements.

A business process ultimately changes an initial set of states into a final set of states of objects. The final set of states may be the mutual business goal. The goal may be the successful exchange of products and money. The conclusion of a contract, in which the conditions are set for future trade, can however be a goal as well. Even the failure to reach agreement, but collecting information on the process that lead to such a failure, can be regarded as a business goal.

For each Process step the prerequisites can be defined as a collection of preconditions. A process step must report about an event, e.g. the decision to accept an order or the (physical) delivery of goods. Preconditions are the information requirements of the responsible partner to enable him to take the step. E.g. in order to accept an order, the requested delivery date must be known. Therefore for order acceptance the presence of a valid requested delivery date is a precondition.

Each event leads to new information that can be expressed in object property values. The post-condition of the event defines its result. The final result of a process should be a business goal or at least a valid end state (UMM makes a distinction between business goals and business failures). A business failure is a valid end state. So the last events that take place have as post-condition the business goal or a business failure. The pre-conditions of these events set the post-conditions of the preceding events, etc.

The precondition of some step in the process is partly derived from the information the partner needs in order to perform the task to set the step. This may be obvious (shipping products needs information on quantities), it may involve legal and trust issues (“I am only prepared to deliver if you can guarantee you are credible”) or it may be subjective (“I can only make the decision to accept the order if I have a clear view on the production capacity needed”). Sometimes the preconditions need to be negotiated. Some initially requested information may not be available at that time. Then the request may be relieved, the information provider may try harder [17] or the process flow may change, e.g. by inserting an intermediate step.

Profile matching

A simpler mechanism than negotiation is profile matching. Each of the companies in advance specifies the information requirements and other preconditions per process step plus the information he has available afterwards (post-conditions). When comparing the two profiles some pre-conditions will appear to be the subset of the post-conditions of other steps. This way the process flow can automatically be generated.

No need to say the conditions in both profiles must be based on the same ontology. Without feedback loops in the process in which the profiles are adapted, the probability

that a feasible process will result from such exercise seems low. But then, apart from automation, this in fact is the mechanism with which business works with for centuries.

Computer applications are more rigid than human employees. That is why in most companies humans are needed to interpret incoming business documents, before they are input to the application. The 'real' profile of a company is far more flexible than the profile that would describe the interface to its application system. For business-to-business interconnectivity therefore probably two profiles are needed: an external profile and an internal profile. The external profile defines what the company really accepts from the market. The internal profile describes the (rigid) interface to the system. The external profile is communicated to trading partners. The internal profile is used to control middleware and human interfaces to translate the incoming and outgoing messages to. In chapter 9 we show how middleware can connect internal and external profiles and how the information system can dynamically adapt to different partner requirements.

Let us first more precisely inspect how an interface specification is derived at.
Companies develop and exchange or publish a profile based on goals, business models, policy, risk assessments and capabilities. In chapter 10 some methods and instruments are described for achieving that. The profile must be derivable from the requirements analysis that gathers these aspects. The profile must be in line with the internal processes. The profile must be understood by the trading partners, so be based on a common understanding of event types that may take place in the collaboration. In chapter 9 the ontology of inter-organizational events is elaborated.
Profiles must be negotiable. It must be possible, in an automatic fashion, to compare two profiles and to find a common inter-organizational process description that fits both.

In the present UMM during the BRV phase the inter-organizational process is being discovered or designed. UMM does not (yet) include profile-matching. The process is defined by modelling business activities. There is a notion of business entities that are affected by the activities, but it is not clear how.

Defining process steps as State Transitions has another advantage. In a State Transition is defined what property values of the affected object(s) must change, what values may change and what values may not change. That information can be used, both to directly generate the database transaction type (e.g. as an SQL Update statement) and as the schema of the (EDI or XML) message that is exchanged during the process step.

7.4 Case orientation

Process and workflow modelling are based on the "case" paradigm. Various employees in series or in parallel process a file with information on a specific case (e.g. a mortgage request). Logically, the data travels with the case. The emphasis of workflow control is on the activities employees perform, not on the data they access or alter. Control system decisions can be made on basis of data, but such data is mainly administrative data from the file cover, not from within the file itself. In the workflow modelling application of Petrinets, Petrinet Tokens represent the cases (see section 7.6).

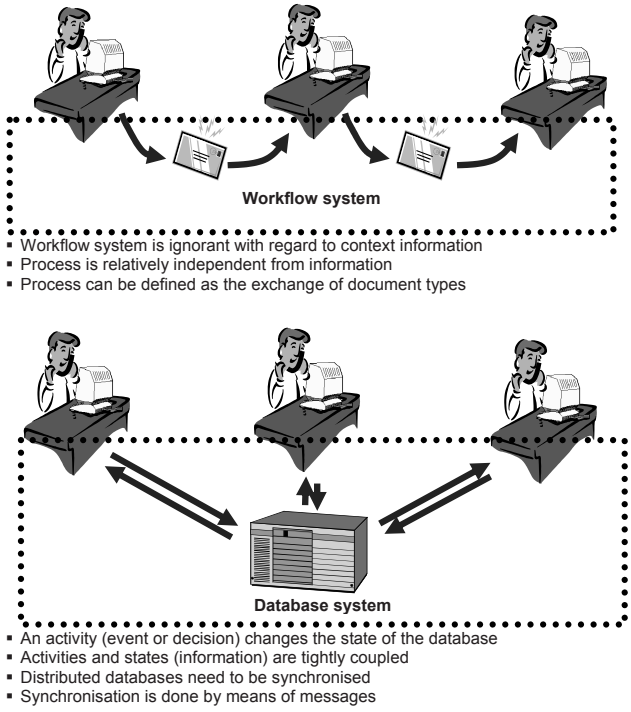


Figure 7.3 Workflow versus Database

Process models usually have a Process or Case as their scope. In case of B2B however an overall contract may exist that sets conditions for the total commercial relationship, followed by a number of blanket orders, one per product group, many call offs per blanket order, shipments that span several call offs and even blanket orders and invoices that may span multiple shipments or apply to parts of a shipment. On top of that we may have master-data alignment, such as the exchange of catalogues and price lists. These processes influence each other. The scope is the total B2B relationship, not an individual order or case.

For electronic business we therefore prefer the Database paradigm rather than the Case paradigm (see figure 7.3). In the Database paradigm activities between people in an organization are co-ordinated by means of an information system. The information system records events, relevant to the co-ordination. Events can be physical (e.g. the arrival of a truck) or be the decision of some employee (e.g. to accept an order). An organizations' information system may be a fully automated Enterprise Resource System, a patchwork of functional, departmental and personal systems or even a paper-based system using documents, forms and file cabinets. We assume that within the organization information and knowledge is shared.

7.5 Requirements

In the sequel of this chapter we shall investigate which modelling languages are suited to support B2B interface definitions, taking into account the requirements stated in chapter 3. The previous section abstracted from language implementations, the description was abstract and the analysis was empirical rather than constructive. In order to choose or design a language, its requirements need to be stated.

It follows from the analysis, that a language for defining the interface should be able to capture states and state transitions. The language should be able to model transactions, involving multiple object types as more than one object can change state during a process step. The language should be able to support matching of profiles and (re)negotiation of the process flow. It should also contain a constraint language to express pre-, post and transition conditions.

Further, the business interface specification language should lead to models that are:

- Complete (including both informational aspects and process dynamics, Req. 4 in section 3.2)
- Consistent (Req. 8)
- Modular (parts being reusable in different contexts, Req. 7)
- Extensible (Req 5)
- Enforceable (being part of contractual or legal arrangements, Req. 8)
- Implementable (Req. 13)
- Understandable (by business people, Req.2)

As shown in chapter 5, the logical view on business-to-business interface definitions (Business Operational View, BOV, in ISO14662), abstracts from the 'flowing' of information or messages. Messages offer a technical mechanism to synchronise mutual knowledge. The synchronization mechanism takes place on another level: in the Functional Service View (FSV). On the interface, information is provided that can be shared by the business partners. The interface definition consists of a (metadata) schema of the (virtual) information base and of a process definition that describes the sequencing of the update transactions to that information base.

In principle three approaches exist to model business processes:
1. Activity based languages
2. State charts
3. Message or transition definitions
We shall inspect each of these in sections 7.6, 7.7 and 7.8 respectively.

7.6 Activity based languages

Activity based languages define a business process as a directed graph in which the nodes are activities and the edges are (allowed) conditional or causal relationships between the

activities. When two activities A and B are interconnected, activity B may take place if and when A has been completed. Most languages to express business processes in this way support forks and choices. Some support the modelling of the time aspect, e.g. defined delays.

Activity based languages include UML Activity diagrams, Petrinets and workflow modelling languages such as BPMN and BPEL.

According to e.g. Söderström [18] the basic grammar of most process modelling languages is derived from Petrinets. Petrinet language [19] is a formal and graphical method, which is used to model computer software, hardware, information flow, control flow, and business processes.

Petrinets and modelling languages derived from Petrinets (such as UML Activity Diagrams, BPMN and BPEL) offer a view on the process that may be recognized and discussed by business people. Experience shows that activity models are easier to understand than data models. For many non-insiders data models, state transition diagrams and (database) transaction models are counter-intuitive and difficult to manipulate.

Petrinet model types are however not complete, especially not in the context of the reference model as described in chapter 5. They miss the notion of information on objects that are affected by the process. Activity based models are also not modular. It is not possible to simply cut a part out of a Petrinet diagram and paste it in the middle of another one. Activities (if not defined as state transitions) carry all previous activities in their semantics.

Petrinets and other activity based models are represented as graphs. Graphs are difficult to manipulate and to match. That matching is possible has been shown by Krukkert [23] and Wombacher [24], who both present mechanisms to match process graph profiles (in their case UML activity diagrams). Matching process definitions is necessary if and when process flows are (automatically) negotiated. The algorithms developed are however complex and it is questionable whether they sufficiently scale.

Georgakopoulos et al [25] assessed the use of activity based models in Virtual Enterprises in the telecommunication industry. He came to the conclusion that the present modelling languages would lead to an explosion of complexity.

We may conclude that the Petrinet language (and languages based on Petrinets) offers a very precise and formal way to specify process flows, but it lacks a few important possibilities and features, notably the binding to (information on) business objects and manipulation (e.g., negotiation).

7.7 State charts

State charts have been introduced by Harel [26]. State charts resemble activity based graphs, but where in most activity graphs the nodes are activities and the edges are causal or temporal relationships between them, in State charts the nodes are states and the edges are transitions (see figure 7.8).

In UML, state charts have been implemented as state diagrams. A state diagram is bound to an object type. Synchronization between transitions of states of different objects cannot be modelled by means of state diagrams.

State charts have a better binding to object data than activity diagrams. Although in a state chart as a diagram the data binding is not explicit, a State may correspond to a query to the database. For example: all orders have state Order. Pending where [Order. Date <> null AND Order. Acceptance date = null]. So it is always possible to verify whether a State is actual or not. Bider [27] argues that one of the advantages of state charts is that a system described with State Charts can always recover from unexpected changes. As soon as one of the valid States is restored, the process may continue. Activity based models need the process history, which may be corrupted after such incident. Frank [28] shows that state charts are also very suitable to integrate the process views from the perspective of different actors.

State Charts are modular. Two state-based models can simply be combined (even with cut-and-paste). States can be subsets of each other, so models may be refined and specialized. As states are defined on property values, even if objects specialize, a state diagram may still be valid. Matching two (profiled) state diagrams is easy: if the states correspond, the transitions remain that are present in both profiles. If the states do not precisely correspond, still a match may be found by inspecting subset relations between states [28].

State Charts have, like Petrinets, a mathematical foundation.

As said, although some authors (Bider [27], Frank [28]) plea for using state charts for requirements gathering, state charts are counter-intuitive to most business users. It is possible though, to present a state transition diagram as a Petrinet, an Activity diagram or a BPMN drawing. Then consistency checking and simulation can be performed and the process can be validated by a business expert.

State charts cannot completely define a business interface. Synchronization between state transitions of different objects is not supported. Transitions are modelled as simple edges. Constraints that go beyond state definitions (such as Transition Conditions: e.g. some attribute value may only be increased) are not supported. Yet, state orientation offers a more promising ground for B2B process specification than activity orientation. The binding with information of business objects is clearer and the modularity allows easier manipulation.

7.8 Transition definitions

Message structures are (static) descriptions of the information exchanges between the partners. Messages update the information base. Message exchange can be documented in, e.g., a UML Sequence diagram. Such a diagram is very simple.

What information should be present in a message is partly dependent on the State definitions. A State defines value spaces of object properties. The messages, exchanged in advance of an object being in a State need to bring the properties in that value space. If, for instance, some transition is to bring an Order object into a 'Delivered' State that is defined by the presence of a Delivery Date the preceding message must contain that date. So the message schema at least contains the information elements that distinguish the States before and after the transition (the 'delta' between the States). In addition a message should contain the information that is needed by the receiving partner to perform the subsequent activity that leads to the next State transition.

Messages are usually defined by means of message schemas. Such schemas may simply define the (hierarchical) structure of data types that are present in the message. In traditional EDI such schema is called a Message Implementation Guideline and it usually may contain all kinds of (textual) constraints and conditions.

XML also has its schema language (XML schema). XML schemas may be enriched by Schematron clauses that contain XPath statements that also may define all kinds of constraints. It is possible to define pre-, post- and transition conditions in Schematron.

It is thus feasible, departing from state oriented process models, to define the dynamics of a business collaboration totally in message specifications. As message specifications are the traditional way to define an inter-organizational interface, this will offer the industries a smooth transition from pure message orientation to process orientation.

At design and standardization time syntax and technology neutral mechanisms may be used to develop and store message and process definitions, e.g. based on Core Components, OCL and/or business rules.

7.9 B2B process definitions

Keeping the high level architecture in mind, we now draft the outlines of B2B process specifications. In order to assess and monitor mutual commitments, obligations and expectations, business partners need to agree which events are allowed to take place under certain conditions, in what order and what their consequences are. The sequence of events that may take place can be stated in some business process modelling language, like UML Activity diagrams, BPMN or Petrinets. Most of these business process modelling languages however do not directly relate the events to the state of affairs, which is defined by the (allowed) set of property values of the objects in the universe of discourse.

Events can be defined by the state transitions they cause. If we are to define the agreed set of allowed events, we can define the states and define which transitions are allowed between the various states. States can be defined as sets of first order logic statements, and can be expressed in some language that supports references to object property values and first order logic. OCL [29] is such language.

States and events can be defined and published relatively independently from each other and from the business process(es) they are part of. States are only related to the definition of the Universe of Discourse (a reference information model). Events cause transitions between states and consequently are related to those states, but not to other events or other states. Sets of States and Events therefore can be developed and published in modules.

An important advantage of this approach is that process definitions and data definitions are isomorph, so can be expressed in exactly the same way and by the same mechanisms. In present activity based solutions, like BPSS and BPEL, data definitions and process definitions are defined by means of entirely different structures. By defining processes in a state-oriented way they can be expressed as constraints to message schemas. This guarantees that process and data are consistent and eases implementation. In chapter 13 we illustrate this by designing a stand-alone, browser based workstation that can handle B2B processes and messages without the need to install software.

The meta-model as defined in chapter 6 allows to define states. A state is the presence of certain utterances in the B2B knowledge base. States can be preconditions to new utterances and can be used to define transactions. In chapter 6 a table representation of B2B information was presented. The tables allow to present both instances and definitions of concepts and entities. The tables also allow to document States and to use States as preconditions of transactions. In this section it is shown that business processes can be presented as tables that were defined in chapter 6. Note that the table presentation is only one way to represent the meta-model that has been presented in chapter 6. The table information can also be represented in modelling and exchange languages such as UML, ORM and XML. This is also true for the process models in this section. Such mapping is described in chapter 8.

States, defined on object types are reusable across process types. States can be measured by inspecting the property values of an object, without knowing the history. States are further constraining property values, so the mechanism to define states can be the same as the mechanism to define object specializations. As States interconnect process steps and as States are defined on the data content, States form the glue between dynamical behaviour (process) and static definition (ontology).

As an illustration, imagine an ordering process as pictured in figure 7.5. The rounded rectangles represent states of the order object, the arrows state transitions. In principle all shown state transitions are allowed.

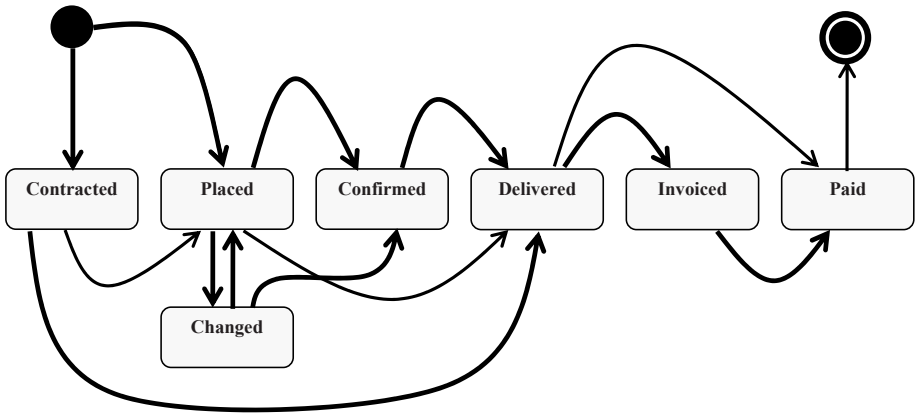


Figure 7.5 Ordering state transitions

Each state is defined as the combined value space of order properties. The property may be a date (e.g. the Order. Date) that is filled or not filled, but it may also be an association with another object (e.g. an Invoice) that is in some specified state. The knowledge base of table 7.2 abstracts from the precise state definitions. The purpose of table 7.2 is to illustrate specification of state transitions and the manipulation of those transitions to define several process flows. In section 7.12 the definition of states is illustrated in a more elaborated example.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10*	00:10	B	Add	Sta	Assert	Contracted_ Order	State definition											
101	10*	00:10	B	Add	Sta	Assert	Placed_ Order	State definition											
102	10*	00:10	B	Add	Sta	Assert	Changed_ Order	State definition										101	
103	10*	00:10	B	Add	Sta	Assert	Confirmed_ Order	State definition										101 OR 102	
104	10*	00:10	B	Add	Sta	Assert	Delivered_ Order	State definition										100 OR 101 OR 103	
105	10*	00:10	B	Add	Sta	Assert	Invoiced_ Order	State definition										104	
106	10*	00:10	B	Add	Sta	Assert	Paid_ Order	State definition										104 OR 105	

* Assuming that the Order Definition is asserted in Utt #10.

Table 7.2 Order states

In a Vendor Managed Inventory situation, simply some of the transitions are ruled out. This can be done by agreeing on additional constraints. The mechanism to negotiate such constraints will be elaborated in chapter 10.

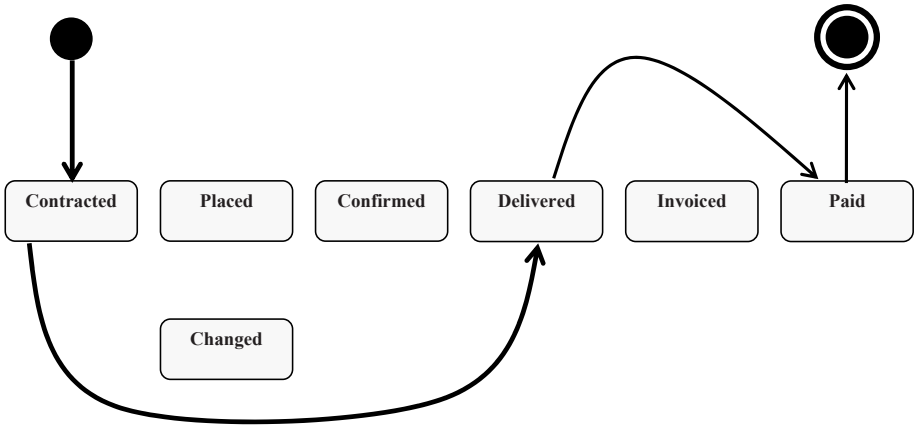


Figure 7.6 Vendor Managed Inventory state transitions

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10*	00:10	B	Add	Sta	Assert	Contracted_ Order	State definition											
104	10*	00:10	B	Add	Sta	Assert	Delivered_ Order	State definition										100	
106	10*	00:10	B	Add	Sta	Assert	Paid_ Order	State definition										104	

* Assuming that the Order Definition is asserted in Utt #10.

Table 7.3 Order states for a Vendor Managed Inventory process

A business process is a sequence of utterances. A business process type is a graph of allowed utterances, where each preceding utterance is a precondition of a following utterance. It is therefore possible to define a business process type by defining preceding utterances as preconditions of following utterances.

7.10 Workflow patterns

All activity oriented process definition languages offer four basic patterns for process flows: Sequence, Choice, Iteration and Parallelism. In state oriented languages Sequence can be modelled by defining the activity that precedes another activity as precondition for that other activity. Choice is defined by mutually defining the non-existence of the two choices as precondition to each other. Iteration is modelled by defining a cyclic flow.

Entrance to the loop and exit from the loop are modelled by means of additional choices (or conditions based on information values). Parallelism is modeled by giving the same precondition to two or more activities.

As an additional pattern we should mention the sub process pattern. A high level process (e.g. ‘contracting’) models a number of steps or documents without detailing them. At a lower level the steps are specified. Although such abstract process does not really define a document to be sent or received, the delta between the information in the knowledge base before and after the process was executed can be specified. Also the preconditions for the process to start can be defined. That way the set of documents that are exchanged during the process can be represented as one information flow or (abstract) document.

This way it is possible to define a business process first as big steps, without detailing the process in terms of smaller steps nor detailing the information that needs to be exchanged. Next the process can be refined. Smaller steps can be included and documents may be defined more detailed.

In a dynamic environment process definitions may change. In workflow systems this is an issue, because it may not be clear how existing workflow instances may be affected. In a B2B environment as described here, process definitions (placeholders) are being exchanged just like and alongside process instances. Each instance may have its own definition. It is therefore always clear which activities are allowed and which are not.

Most activity based specifications (and the systems that support them) suffer from poor integration with the information that is stored and processed. Van der Aalst c.s. [31] have listed 39 features regarding process-data integration that may be required. They investigated 6 popular workflow solutions. On average only 14 features were fully supported and 9 features were partly supported.

All these features are supported in a state or transaction oriented system. Pre-, post and transaction conditions can reach all data of all objects that can be navigated to. Activities are update transactions that have access to the full information base. Activities that share the same precondition may run in parallel, so parallelism is supported as well.

It can be concluded that for B2B process specification, state-oriented modelling is as powerful as activity based modelling, and is superior with regard to data visibility and manipulation.

It must be noted that, though representation of a process flow in a state oriented way has many advantages over activity oriented representation, it also has a disadvantage. When activity oriented flows are represented by means of Petrinets it is relatively simple to identify process flow defects, such as the possibility for dead locks and live locks. In state oriented models identification of such defects is not that straightforward (unless they are transformed to Petrinets, which is always possible).

7.11 Assessment

As mentioned before, languages to define business interface specifications should be:

- Complete (including both informational aspects and process dynamics)
- Consistent
- Modular (parts being reusable in different contexts)
- Extensible
- Enforceable (being part of contractual or legal arrangements)
- Implementable

The most important advantage of expressing business processes in terms of States and Transitions is the consistency between process steps and information (property values). Another advantage is the modularity of the resulting models. States can be nested and it can be determined whether states are disjoint or overlapping.

Modular process models can be developed and standardised in a modular way. Modules can be published. In a specific business context only those modules that are applicable to that context may be selected and implemented. The process choreography may even be negotiated bilaterally based on specific interests business partners may have. Without modularity business processes must be designed and fine-tuned for each different situation (which is the present situation for most EDI and XML based communication). State oriented models can easily be extended by adding States and Transitions.

The State definitions unambiguously define the data that must be present at any point in the process. The definitions can be expressed in a formal language such as OCL. Behaviour of trading partners is measurable and consequently enforceable. The formal definition also makes the specifications implementable in the partners' application systems and/or in middleware that supports the communication between the application systems. States can be represented as database schema, and transitions as database transactions or (XML-)message schema.

7.12 Example

In a simple example we shall illustrate State oriented process development. The process in the example is a simple spot-ordering process. We assume that orders are always being accepted and not negotiated. Orders are neither changed nor split, and products are delivered as ordered. When the delivery is planned, the total order is invoiced and the invoice is paid after the delivery. An order may consist of multiple orderlines. Only one delivery per order may take place and only one invoice is sent for the order. The seller must accept the order (or not) and sends the invoice when he delivers the order. In figure 7.7 we show the UML Class diagram and the Activity Diagram of this process[4].

[4] For simplicity, the Order in this example is not modelled as a Planned Delivery, but as a business object in its own right.

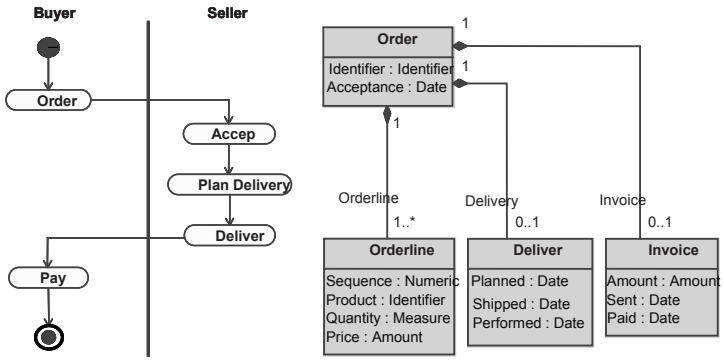


Figure 7.7 Activity (left) and Class (right) Diagrams

Order, Orderline, Delivery and Invoice are defined as Business Objects. The activities shown in the activity diagram are Transitions. The Class diagram can be represented as the following list of Business Information Entities:

Order. Details Order. Identifier Order. Acceptance. Date Order. Orderline Order. Delivery Order. Invoice	Delivery. Details Delivery. Planned. Date Delivery. Shipped. Date Delivery. Performed. Date
Orderline. Details Orderline. Sequence. Numeric Orderline. Product. Identifier Orderline. Quantity. Measure Orderline. Price. Amount	Invoice. Details Invoice. Total. Amount Invoice. Sent. Date Invoice. Paid. Date

Table 7.4 Business Information Entities

Alternatively to the Activity diagram, the process can be represented by a State Transition diagram (figure 7.8). In the diagram we indicate the Pre- and Post-conditions, and the events that trigger the transitions.

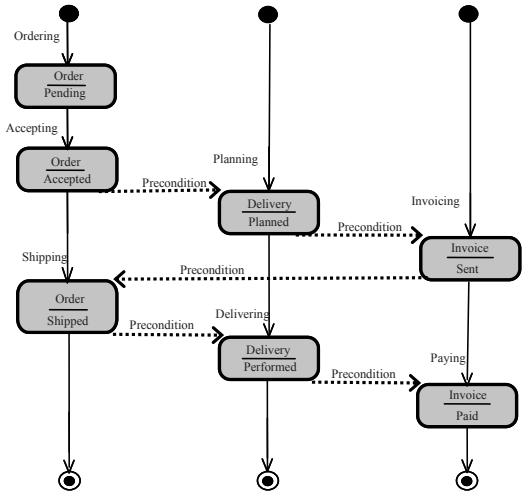


Figure 7.8 State Transition Diagram

The States and Conditions are defined as follows:

States	
Pending_ Order	**Order. Acceptance. Date = null**
Accepted_ Order	**Order. Acceptance. Date <> null** **Order. Delivery. Shipped. Date = null**
Shipped_ Order	**Order. Delivery. Shipped. Date <> null**
Planned_ Delivery	**Delivery. Planned. Date <> null** **Delivery. Performed. Date = null**
Performed_ Delivery	**Delivery. Performed. Date <> null**
Sent_ Invoice	**Invoice. Sent. Date <> null** **Invoice. Paid. Date = null**
Paid_ Invoice	**Invoice. Paid. Date <> null**

Table 7.5 States

Event	Initiating Role	Precondition		Post-condition		Delta
		Object	State	Object	State	
Ordering	Buyer	Order	null	Order	Pending	Order Order. Identifier
Accepting	Seller	Order	Pending	Order	Accepted	Order. Acceptance. Date
Planning	Seller	Order Delivery	Accepted null	Order Delivery	Accepted Planned	Order. Delivery Delivery. Planned. Date
Invoicing	Seller	Invoice	null	Invoice	Sent	Order. Invoice Invoice. Sent. Date
Shipping		Order	Accepted	Order	Shipped	Delivery. Shipped. Date
Delivering	Seller	Delivery	Planned	Delivery	Performed	Delivery. Performed. Date
Paying	Buyer	Order Delivery Invoice	Shipped Performed Sent	Order Delivery Invoice	Shipped Performed Paid	Invoice. Paid. Date

Table 7.6 Events

The B2B knowledge base can be filled on the basis of these models and tables.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotyne	With allowed Intention	Precondition #	Transaction with #
100	0	00: 10	B	Prop	Def	Assert	Order	Owner	Have	Identifier	Identifier	1	1	1	B, S		Assert		
101	0	00: 10	B	Prop	Exp	Assert	Order	Owner	Have	Acceptance	Date		0	1	B, S		Assert		
102	0	00: 10	B	Prop	Exp	Assert	Order	Owner	Have	Line	Orderline		1	n	B, S		Assert		
103	0	00: 10	B	Prop	Exp	Assert	Order	Owner	Have	Delivery	Delivery		0	1	B, S		Assert		
104	0	00: 10	B	Prop	Exp	Assert	Order	Owner	Have	Invoice	Invoice		0	1	B, S		Assert		
105	0	00: 10	B	Prop	Def	Assert	Orderline	Owner	Have	Sequence	Numeric	1	1	1	B, S		Assert		
106	0	00: 10	B	Prop	Exp	Assert	Orderline	Owner	Have	Product	Identifier		1	1	B, S		Assert		
107	0	00: 10	B	Prop	Exp	Assert	Orderline	Owner	Have	Quantity	Measure		1	1	B, S		Assert		
108	0	00: 10	B	Prop	Exp	Assert	Orderline	Owner	Have	Price	Amount		0	1	B, S		Assert		
109	0	00: 10	B	Prop	Def	Assert	Delivery	Owner	Have	Planned	Date		1	1	B, S		Assert		
110	0	00: 10	B	Prop	Exp	Assert	Delivery	Owner	Have	Shipped	Date		0	1	B, S		Assert		
111	0	00: 10	B	Prop	Exp	Assert	Delivery	Owner	Have	Performed	Date		0	1	B, S		Assert		
112	0	00: 10	B	Prop	Def	Assert	Invoice	Owner	Have	Sent	Date		1	1	B, S		Assert		
113	0	00: 10	B	Prop	Exp	Assert	Invoice	Owner	Have	Total	Amount		0	1	B, S		Assert		
114	0	00: 10	B	Prop	Exp	Assert	Invoice	Owner	Have	Paid	Date		0	1	B, S		Assert		

Table 7.7 Knowledge base

First, in table 7.7 the data model is proposed by the Buyer. Order, Orderline, Delivery and Invoice are defined as entities with their properties. For simplicity, the acceptance of the utterances by the Seller is not shown in table 7.7. Acceptance is done by the Seller by copying the utterances and filling column 5 with 'Acc'(ept) instead of 'Prop'(ose).

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
121	101	0:20	B	Add	Sta	Assert	Pending_ Order	Owner	Have	Acceptance	Date		0	0					
122	101	0:20	B	Add	Sta	Assert	Accepted_ Order	Owner	Have	Acceptance	Date		1	1				123	
123	103	0:20	B	Add	Sta	Assert	Accepted_ Order	Owner	Have	Delivery	A_ Delivery		0	1				122	
124	110	0:20	B	Add	Sta	Assert	A_ Delivery	Owner	Have	Shipped	Date		0	0					
125	103	0:20	B	Add	Sta	Assert	Shipped_ Order	Owner	Have	Delivery	S_ Delivery		1	1				129	
126	110	0:20	B	Add	Sta	Assert	S_ Delivery	Owner	Have	Shipped	Date		1	1					
127	111	0:20	B	Add	Sta	Assert	Planned_ Delivery	Owner	Have	Performed	Date		0	0				122	
128	111	0:20	B	Add	Sta	Assert	Performed_ Delivery	Owner	Have	Performed	Date		1	1				125	
129	114	0:20	B	Add	Sta	Assert	Sent_ Invoice	Owner	Have	Paid	Date		0	0				127	
130	114	0:20	B	Add	Sta	Assert	Paid_ Invoice	Owner	Have	Paid	Date		1	1				128	

Table 7.8 State definitions

In table 7.8 the states are defined that play a role in the process definition. In Utt. #121 the Pending order state is defined. According table 7.5 the Pending state is defined as an Order without an Acceptance. Date. This is represented in Utt. #121 by setting the minimum and maximum repetitions of the Acceptance. Date to zero (col. 14 and 15). Utterances #122 to #124 define the Accepted order state. In this state the Acceptance. Date must have been filled, so the repetitions in col. 14 and 15 are set to one. The order may have a Delivery (Utt. #123), but that Delivery must not have a Shipped. Date (Utt. #124, col. 14 and 15). Likewise the other states are defined. In Utt. #125 and #126 the Shipped_ Order, in Utt. #127 to #130 the Planned_ Delivery, the Performed_ Delivery, the Sent_ Invoice and the Paid_ Invoice. In col. 19 the preconditions as indicated in figure 7.9 are stated. Then, in table 7.9, the messages or transactions are defined as Perceptions (col. 6). For each transaction the responsible role is defined in column 16. The Preconditions for each transaction, consisting of utterances that must be exchanged in advance and/or states from table 7.8, are listed in column 19. Table 7.9 both specifies the process specification and contains the data for the message schemas.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotyne	With allowed Intention	Precondition #	Transaction with #
141	100	0:30	B	Prop	Per	Assert	Order	Owner	Have	Identifier	Identifier	1	1	1	B	Ins	Request		142
142	102	0:30	B	Prop	Per	Assert	Order	Owner	Have	Line	Orderline		1	n	B	Obs	Request		143
143	105	0:30	B	Prop	Per	Assert	Orderline	Owner	Have	Sequence	Numeric	1	1	1	B	Ins	Request		114
144	106	0:30	B	Prop	Per	Assert	Orderline	Owner	Have	Product	Identifier		1	1	B	Obs	Request		145
145	107	0:30	B	Prop	Per	Assert	Orderline	Owner	Have	Quantity	Measure		1	1	B	Obs	Request		141
146	100	0:30	B	Prop	Per	Assert	Order	Owner	Have	Identifier	Identifier	1	1	1	S	Obs	Accept	121	147
147	101	0:30	B	Prop	Per	Assert	Order	Owner	Have	Acceptance	Date		1	1	S	Obs	Accept		148
148	102	0:30	B	Prop	Per	Assert	Order	Owner	Have	Line	Orderline		1	n	S	Obs	Accept		149
149	105	0:30	B	Prop	Per	Assert	Orderline	Owner	Have	Sequence	Numeric	1	1	1	S	Obs	Accept		150
150	108	0:30	B	Prop	Per	Assert	Orderline	Owner	Have	Price	Amount		1	1	S	Obs	Accept		146
151	100	0:30	B	Prop	Per	Assert	Order	Owner	Have	Identifier	Identifier	1	1	1	S	Obs	Plan	123	152
152	103	0:30	B	Prop	Per	Assert	Order	Owner	Have	Delivery	Delivery		1	1	S	Obs	Plan		153
153	109	0:30	B	Prop	Per	Assert	Delivery	Owner	Have	Planned	Date		1	1	S	Ins	Plan		154
154	104	0:30	B	Prop	Per	Assert	Order	Owner	Have	Invoice	Invoice		1	1	S	Obs	Claim		155
155	112	0:30	B	Prop	Per	Assert	Invoice	Owner	Have	Sent	Date		1	1	S	Ins	Claim		156
156	113	0:30	B	Prop	Per	Assert	Invoice	Owner	Have	Total	Amount		1	1	S	Obs	Claim		151
157	100	0:30	B	Prop	Per	Assert	Order	Owner	Have	Identifier	Identifier	1	1	1	S	Obs	Assert		158
158	104	0:30	B	Prop	Per	Assert	Order	Owner	Have	Invoice	Invoice		1	1	B	Obs	Assert	129	159
159	103	0:30	B	Prop	Per	Assert	Order	Owner	Have	Delivery	Delivery		1	1	S	Obs	Assert		160
160	110	0:30	B	Prop	Per	Assert	Delivery	Owner	Have	Shipped	Date		1	1	S	Obs	Assert		157
161	100	0:30	B	Prop	Per	Assert	Order	Owner	Have	Identifier	Identifier	1	1	1	S	Obs	Assert	125	162
162	103	0:30	B	Prop	Per	Assert	Order	Owner	Have	Delivery	Delivery		1	1	S	Obs	Assert		163
163	111	0:30	B	Prop	Per	Assert	Delivery	Owner	Have	Performed	Date		1	1	S	Obs	Assert		161
164	100	0:30	B	Prop	Per	Assert	Order	Owner	Have	Identifier	Identifier	1	1	1	B	Obs	Assert		165
165	103	0:30	B	Prop	Per	Assert	Order	Owner	Have	Delivery	Delivery		1	1	S	Obs	Assert	128	166
166	104	0:30	B	Prop	Per	Assert	Order	Owner	Have	Invoice	Invoice		1	1	B	Obs	Assert		167
167	114	0:30	B	Prop	Per	Assert	Invoice	Owner	Have	Paid	Date		1	1	B	Obs	Assert		164

Table 7.9 Perceptions

Finally, in table 7.10, the operational data exchange is shown. This data can be represented as (standard) XML or EDI messages. On receiving each transaction (set of utterances with the same time stamp) the middleware may check whether the preconditions are fulfilled. That way the process may be monitored.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
200	141	2:00	B	Add	Ins	Request	Order	Owner	Have	Identifier	123	1							
201	142	2:00	B	Add	Obs	Request	Order	Owner	Have	Line	1_ Line	1							
202	142	2:00	B	Add	Obs	Request	Order	Owner	Have	Line	2_ Line	1							
203	143	2:00	B	Add	Ins	Request	1_ Line	Owner	Have	Sequence	1								
204	144	2:00	B	Add	Obs	Request	1_ Line	Owner	Have	Product	4711								
205	145	2:00	B	Add	Obs	Request	1_ Line	Owner	Have	Quantity	120 PCE								
206	143	2:00	B	Add	Ins	Request	2_ Line	Owner	Have	Sequence	2								
207	144	2:00	B	Add	Obs	Request	2_ Line	Owner	Have	Product	4712								
208	145	2:00	B	Add	Obs	Request	2_ Line	Owner	Have	Quantity	50 PCE								
209	146	2:10	S	Add	Obs	Accept	Order	Owner	Have	Identifier	123								
210	147	2:10	S	Add	Obs	Accept	Order	Owner	Have	Acceptance	2010-05-24								
211	148	2:10	S	Add	Obs	Accept	Order	Owner	Have	Line	1_ Line								
212	148	2:10	S	Add	Obs	Accept	Order	Owner	Have	Line	2_ Line								
213	149	2:10	S	Add	Obs	Accept	1_ Line	Owner	Have	Sequence	1								
214	150	2:10	S	Add	Obs	Accept	1_ Line	Owner	Have	Price	15 EUR								
215	149	2:10	S	Add	Obs	Accept	2_ Line	Owner	Have	Sequence	2								
216	150	2:10	S	Add	Obs	Accept	2_ Line	Owner	Have	Price	25 EUR								
217	151	2:20	S	Add	Obs	Planned	Order	Owner	Have	Identifier	123								
218	152	2:20	S	Add	Obs	Planned	Order	Owner	Have	Delivery	456								
219	153	2:20	S	Add	Ins	Planned	Delivery	Owner	Have	Planned	2010-06-24								
220	154	2:20	S	Add	Obs	Claim	Order	Owner	Have	Invoice	789								
221	155	2:20	S	Add	Ins	Claim	Invoice	Owner	Have	Sent	2010-06-01								
222	156	2:20	S	Add	Obs	Claim	Invoice	Owner	Have	Total	3050 EUR								
223	157	2:30	S	Add	Obs	Assert	Order	Owner	Have	Identifier	123								
224	158	2:30	S	Add	Obs	Assert	Order	Owner	Have	Delivery	Delivery								
225	159	2:30	S	Add	Obs	Assert	Order	Owner	Have	Delivery	456								
226	160	2:30	S	Add	Obs	Assert	Delivery	Owner	Have	Shipped	2010-06-25								
227	164	2:40	B	Add	Obs	Assert	Order	Owner	Have	Identifier	123								
228	165	2:40	S	Add	Obs	Assert	Order	Owner	Have	Delivery	456								
229	166	2:40	B	Add	Obs	Assert	Order	Owner	Have	Invoice	789								
230	167	2:40	B	Add	Obs	Assert	Invoice	Owner	Have	Paid	2010-07-25								

Table 7.10 Operational information exchange

7.13 Summary

We have shown that State orientation offers a feasible approach for defining consistent, extensible, modular and implementable inter-organizational e-business interfaces. State oriented standard models may be expressed in OCL and stored in extended Core Component libraries. State oriented models define consistent relation between data definitions and process dynamics. Like normal Core Component models, dynamic models too can be restricted to be applied to specific business environments by further constraining them. Further constraining in fact means to add additional OCL statements. The link between process steps and data to be exchanged is the State. States (in Petrinet: Places) and Process Steps are two sides of the same coin.

To take the State as prime orientation instead of the Process step offers several advantages:

- Process and data are formally linked
- States can easier be defined in a modular way
- States can be specialised from more generic models
- States can be linked to business goals
- States are anchoring points when unexpected events occur
- State transitions can automatically be translated into message schema

References

[1]. International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC): ISO14662 Information Technologies - Open-EDI Reference Model. ISO/IEC 14662:1997(E).

[2]. United Nations.Economic Commission of Europe: UMM UN/CEFACT' s Modelling Methodology. To be downloaded from http://www.unece.org/cefact/umm/umm_all.zip. Accessed on 30 Nov 2005.

[3]. XML Extensible Markup Language (XML) 1.0 (Third Edition), W3C, http://www.w3.org/TR/REC-xml/, Last accessed 2 Nov 2010

[4]. van Blommestein, c.s.: ebXML for Managers, ECP.NL, Leidschendam, The Netherlands

[5]. UN/CEFACT Business Process Specification Schema, 2003-10-18, to be downloaded from http://www.untmg.org/index.php?option=com_docman&task=docclick&Itemid=137&bid Accessed on 30 Nov 2005.

[6] Object Management Group: BPMN. http://www.omg.org/spec/BPMN/2.0/ Accessed on 20 Nov 2011

[7] BEA, IBM, Microsoft, SAP and Siebel, “Business Process Execution Language for Web Services Version 1.1”, S. Thatte, et al., May 2003. http://www.oasis-open.org/committees/download.php/2046/BPEL%20V1-1%20May%205%202003%20Final.pdf Accessed on 20 Nov 2011

[8]. UN/CEFACT TBG17 Terms of reference, to be downloaded from http://xml.coverpages.org/TBG17TermsofReference20040309.pdf. Accessed on 11 December 2005.

[9]. Traum: Speech Acts for Dialogue Agents. In Wooldridge, M., Rao, A., eds.: Foundations of Rational Agency. Kluwer Academic Publishers (1999) 169—201.

[10]. Medina Mora c.s.: The action workflow approach to workflow management technology. In: CSCW '92 Proceedings of the 1992 ACM conference on Computer-supported cooperative work.

[11]. Goldkuhl: Generic business frameworks and action modelling, In proceedings of conference Communication modelling - Language/Action Perspective´96, Springer Verlag, 1996

[12]. Flores c.s.: Computer Systems and the Design of Organizational Interaction. ACM Transactions on Office Information Systems, Vol. 6, No. 2, April 1988, Pages 153-172

[13]. Auramaki c.s.: Modelling Offices Through Discourse Analysis: A Comparison and Evaluation of SAMPO with OSSAD and ICN. in The Computer Journal, vol. 35, n° 5, 1992, pp. 492-500

[14. van Reijswoud c.s.: Modelling business communication as a foundation for business process redesign: a case of production logistics. HICSS (4) 1995: 841-850
[15]. Bollen Conceptual Process Configurations in Enterprise Knowledge Management Systems. In: SAC '06 Proceedings of the 2006 ACM symposium on Applied computing
[16]. Frank c.s.: Equivalence Transformations on Statecharts. Proceedings of the 12th International Conference on Software Engineering and Knowledge Engineering (SEKE 2000), July 6-8, Chicago, USA, pp 150-158.
[17]. Balsters c.s.: Semantics of Interoperable and Outsourced Information Systems. In: G. Doumeingts c.s. (ed.) Enterprise Interoperability, Springer Verlag London, 2007, pp 13-22.
[18]. Söderström c.s.: Towards a Framework for Comparing Process Modelling Languages. CAISE 2002, LNCS 2348, pp. 600-611, 2002. Springer-Verlag Berlin Heidelberg 2002
[19]. Murata: Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE, Vol. 77, No 4, April, 1989, pp. 541-580
[20]. Van der Aalst: Time coloured Petrinets and their application to logistics, PhD thesis, TU Eindhoven, 1992.
[21]. Eshuis c.s.: A Comparison of Petri Net and Activity Diagram Variants. In Reisig Weber, Ehrig, editor, Proc. of 2nd Int. Collaboration on Petri Net Technologies for Modelling Communication Based Systems, pages 93--104. DFG Research Group "Petri Net Technology", September 2001.
[22]. Verbeek c.s.: XRL/Flower: Supporting Inter-organizational Workflows Using XML/Petri-Net Technology. WES 2002: 93-108
[23]. Krukkert: Matching of ebXML Business Processes; Report Project number IST 2001-28548 openXchange
[24]. Wombacher: Matchmaking for Business Processes based on Choreographies. 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04)
[25]. Georgakopoulos c.s.: Managing process and service fusion in virtual enterprises. In: lnformation Systems Vol. 24, No. 6. pp. 429-456, 1999
[26]. Harel: Statecharts a visual formalism for complex systems. In: Science of Computer Programming Volume 8 Issue 3, June 1, 1987
[27]. Bider c.s.: Towards a Formal Definition of Goal-Oriented Business Process Patterns. Business Process Management Journal, Vol.11(6), MCB University Press 2005
[28]. Frank: Integration of Statecharts, Proceedings of the 3rd IFCIS International Conference on Cooperative Information Systems, New York City, USA (CoopIS `98
[29]. Object Management Group: Object Constraint Language. http://www.omg.org/spec/OCL/2.2/PDF Accessed 20 Nov 2011.
[30]. Van der Aalst c.s.: Workflow Patterns. Distributed and Parallel Databases, 14(3):5-51, July 2003
[31]. Van der Aalst c.s.:, Workflow Data Patterns. QUT Technical report, FIT-TR-2004-01, Queensland University of Technology, Brisbane, 2004

8 Mapping to modelling languages

Summary

The table structure for utterances, as presented in chapter 6, is neither intended to serve as an new exchange language, nor as a modelling language to design, standardize or exchange B2B protocols. It is an abstraction to be mapped to existing exchange and modelling languages.

In this chapter such mapping is illustrated. A few exchange and modelling languages (ORM, UML, ERD, SQL, CCTS and XML) serve as examples. Not all features of the utterance table are supported by all languages. In some cases a combination of languages can be used to design and agree on a protocol and to exchange information at run time. In other cases aspects that are explicitly featured in the table may be implicitly modelled in the models.

8.1 Modelling Languages

At the implementation level the business partners use their own computer systems to store the knowledge they have gathered and agreed upon. The systems are interconnected and exchange serialized data in order to synchronize the stored information. Business partners operate their respective systems using private user interfaces. Systems are designed using some modelling language, as illustrated in figure 8.1.

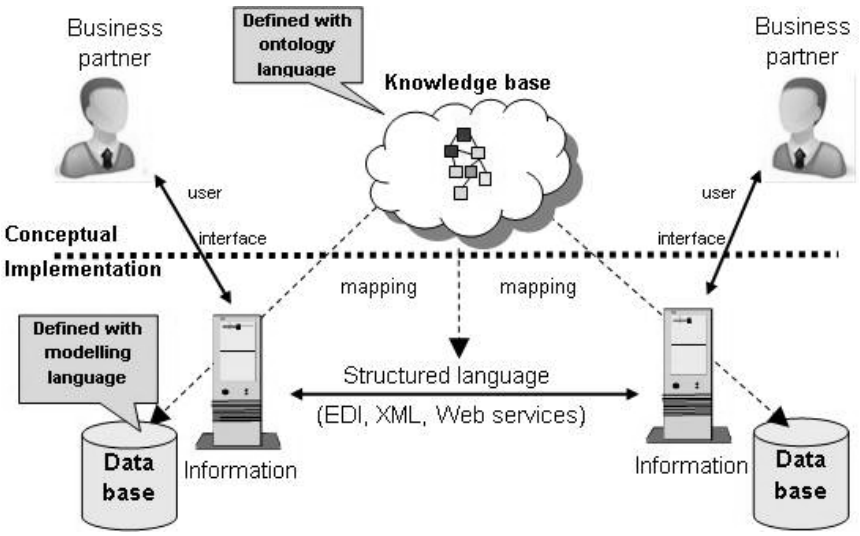


Figure 8.1 Implementation

In section 8.2 through 8.6 it is shown how the knowledge base structure may be mapped to modelling languages to enable implementation. This exercise is needed because the information to be stored is to be mapped to the structure of (existing and new) business information systems. Such systems are usually designed using a modelling language.

In some environments information systems are designed and maintained model driven in a (semi) automatic fashion. Users and domain experts maintain a model that is translated by automatic tooling to database and transaction schemas. In these environments definitions received from a trading partner are presented to the user in the modelling language he is familiar with, or (based on rules previously entered by the user) are directly translated into database, program or service adaptations.

Conversely, as is shown in chapter 10, the model on which the information system is based is transformed into a profile that is proposed to the trading partner. Concept definitions that result from the profile can then be used to base instantiations and observations on.

In sections 8.2 through 8.5 the meta-model is mapped to meta-model elements of ORM, UML, Common Logic Structured English (CLSE) and ERD, to facilitate the implementation of B2B communication in information systems that were designed with those languages.

Most business application systems are modelled in a modelling language such as UML or ERD. The mapping in this chapter allows to translate the meta data of the application to knowledge base utterances. The way these utterances may be conveyed to the business partner is by means of standard (e.g., XML) messages. The translation from utterances to XML messages can be performed by mapping the utterances on UN/CEFACT Core Components [1].

In section 8.6 is illustrated how the M2 model elements are mapped on UN/CEFACT Core Components [1]. The Core Component Technical Specification is the international standard for representing B2B (syntax neutral) data models and messages. It is based on UML Class diagramming. CCTS has been documented as a UML profile [2]. Tools exist to model CCTS structures and to generate XML messages from them. The way CCTS structures are represented in XML has been standardized [3]. By expressing the knowledge base structure and B2B utterances in Core Components, an already accepted and standardized mechanism can be used for implementation. That considerably facilitates adoption.

The objective of sections 8.2 through 8.6 is to show that the structure of the knowledge base and of the utterances that are exchanged between business partners may be expressed in various languages. This is illustrated in figure 8.2. Each language has its own application area, user community and tool support. Adoption of the proposed B2B mechanisms is increased with wider language support.

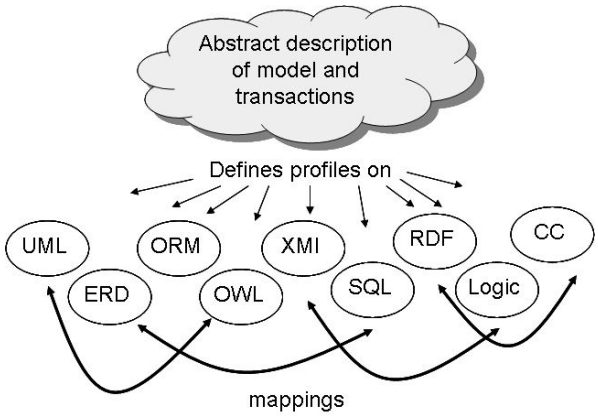


Figure 8.2 Mapping to ontology, modelling and exchange languages

Many languages have been developed to define or describe information systems. Some of these languages were designed for future users of the systems to state their requirements, other are directed towards professional developers and have facilities to automatically produce programming code. Some are informal diagramming methods, other are formalized. In this chapter four languages are assessed in their capability to map their meta-concepts to the meta-concepts needed for B2B systems: ORM, UML and ERD/SQL.

In practice a partner's business information system may be modelled in a modelling language such as ORM or UML. The interface to the business information system (the transactions that are accepted by the system, their information content and their sequence, triggers and preconditions) may also be modelled in a modelling language. The interface model may serve as a basic profile for B2B interconnection. For that purpose the interface model is to be exchanged in the format of an exchange language, such as XML. This thesis offers the way to express profiles in utterances to exchange, by means of the utterance meta-model in chapter 6. That meta-model may be implemented in a middleware system or shortcut, using the functionality of the business information system.

Whether the meta model is implemented or not, the interface to the information system (the profile) need to be translated into XML or EDI messages and the contents of these messages need to be transformed into an adaptation of or mapping to that interface. As here we use the B2B knowledge base structure as an intermediate structure, in this chapter is shown how modelling languages such as ORM and UML and exchange languages such as XML may be mapped to the knowledge base structure.

8.2 ORM

One method for modelling information is Object Role Modelling (ORM [4]). ORM starts with identifying elementary facts. These facts are mainly binary relations between concepts, although also ternary and quarter nary facts may exist. This brings ORM very close to ontological notations, such as conceptual graphs (see section 6.5).

ORM supports interpreting natural language sentences and translating those into structures of object types. ORM has been derived from NIAM, an information analysis methodology to gather requirements to an information system by analysing user assertions on the application area of the system. ORM more or less bridges the ontology world to the database world. The methodology defines how to translate an ORM graph (which is similar to a conceptual graph) to a (relational or Object Oriented) database structure. ORM models can be converted into SQL database schemes.

ORM is a rich data/ontology modelling language. ORM does however not support modelling of conversations. There are no modelling elements to indicate who is uttering a fact or with what intention the fact is uttered. In ORM inheritance is supported, but has a slightly different semantics from the 'Based on' relation in B2B knowledge bases.

The ORM specifications consist of a diagram method with mathematically well founded syntactical and semantic rules. Tools exist that implement this method. No formal exchange language between tools of different make has been defined. Some ORM tools may export ORM models to SQL. Also transformation tools exist between UML and ORM. In order to present definition proposals from a trading partner to a business user, ORM tools should have an import function.

In this thesis informal rules are stated to transform knowledge base clauses to and from ORM diagrams. The set of rules is however not complete and cannot be regarded as a functional or technical design of such transformation.

Core proposition

8	9	10	11	12
Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name

Table 8.1 Core proposition

ORM supports the modelling of facts. Facts are associations between concepts that each plays a role. In ORM models frequently the roles are expressed as verbs. E.g. in figure 8.3 the association is modelled between a car and its wheels.

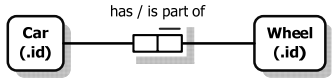


Figure 8.3 Example

In a knowledge base this association would be represented as:

8	9	10	11	12
Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name
Car	Assembly	Have	Part	Wheel

Table 8.2 Example

The knowledge base semantics for verbs and role names are richer than the ORM semantics. The Verb and the role names should be extracted from the role reading when converting an ORM model into a knowledge base structure. Verb and role names must be combined when expressing a knowledge base as an ORM model.

Instantiations and Observations are modelled differently in ORM, see figure 8.4

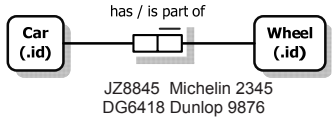


Figure 8.4 Instantiation

The information in Figure 8.4 is represented in the knowledge base as in table 8.3.

8	9	10	11	12
Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name
Car	Assembly	Have	Part	Wheel
JZ8845	Assembly	Have	Part	Michelin 2345
DG6418	Assembly	Have	Part	Dunlop 9876

Table 8.3 Instantiation

So for Instantiations and Observations mapping to ORM is made as illustrated in figure 8.4, while for Definitions and Expansions the mapping is conform figure 8.3.

Perceptions may be represented in ORM as derived fact types. In the ORM diagram a derived fact type has a star (*) after the role names. Derivation rules can be added as text blocks.

States are represented in ORM as textual annotations, not as diagram items.

Identification

1	2	3	4
Utterance #	Based on Utterance #	Timestamp	Uttered by Party

Table 8.4 Identification

Utterance numbers, Timestamps and Uttering parties are no built-in model elements in ORM, so they should be explicitly modelled, as in figure 8.5.

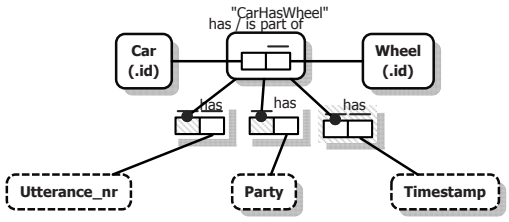


Figure 8.5 Identification

ORM has a subtyping notation, see figure 8.6. The semantics of that mechanism are equivalent to the semantics of the Based on relation. It is even possible to define the discriminating properties of different subtypes of the same super-type. The Based on relation however is a relation between utterances, while ORM subtyping is defined on facts (called events in a knowledge base). Nevertheless a based on relation may be represented in an ORM model as a subtype.

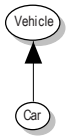


Figure 8.6 Subtyping

Intention

5	6	7
Action	Stereotype	With Intention

Table 8.5 Intention

Actions, stereotypes and intentions are not supported in ORM. Actions (Add/Propose/Accept/Reject) are in fact no modelling elements, but a mechanism to control the addition of new utterances, events or facts. Intentions are modelled by adding the intention as a state to the fact.

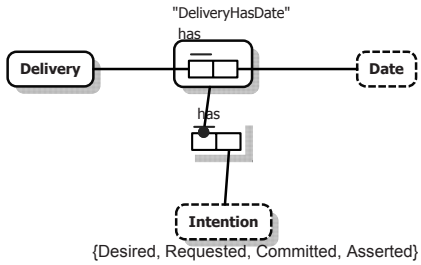


Figure 8.7 Intention

All utterances are isomorph: they fit in the same table structure, although not all attributes are used for all stereotypes. In ORM Instantiations and Observations are differently modelled than Definitions and Expansions and than States and Perceptions. Instance data is shown as a table under the role associations. States are not modelled graphically. ORM models static information models. Dynamic modelling in ORM is under development.

Cardinality

13	14	15
Part of ID #	Min Repetition	Max Repetition

Table 8.6 Cardinality

Cardinality in ORM is modelled by two model elements: uniqueness constraints and mandatory constraints. These can satisfactory be mapped to the repetitions 0, 1 and many (n).

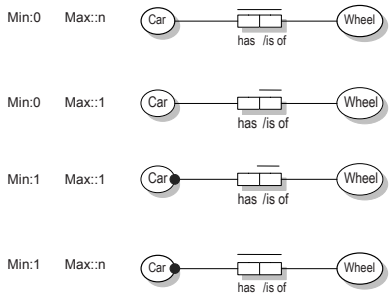


Figure 8.8 Cardinality

Specific repetitions (e.g. 3) can be represented in ORM as well by means of 'frequency constraints'. A frequency constraint is an expression (e.g. '≤2' or '3 – 4') of the frequency some role has with regard to instances of the other role. See figure 8.9.

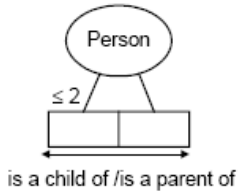


Figure 8.9 Complex cardinality

Definition

16	17	18	19	20
To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #

Table 8.7 Definition

ORM can be extended with dynamic constraints that define or limit transactions on the instance population of a data model.

Example

As an example the ORM model in figure 8.10 is used to populate a knowledge base with proposals for Definitions and Expansions.

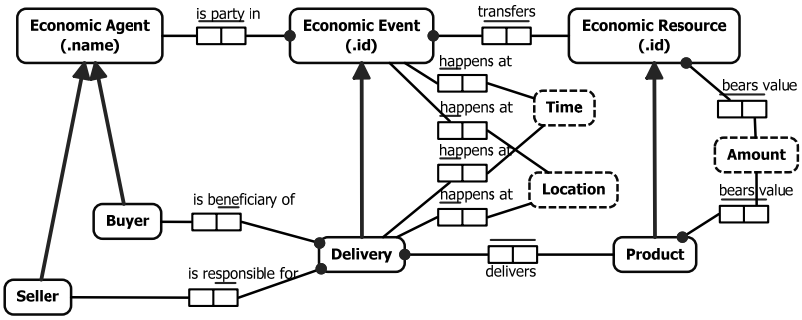


Figure 8.10 Definitions

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	1	01:00	Buyer	Prop	Def		Economic Agent		have		Name	1	1	1	B,S	Def			
101	2	01:00	Buyer	Prop	Exp		Economic Agent	Party	participate		Economic Event		0	n	B,S	Exp			
103	1	01:00	Buyer	Prop	Def		Economic Event		have		Identifier	1	1	1	B,S	Def			
104	2	01:00	Buyer	Prop	Def		Economic Event		transfer		Economic Resource		1	n	B,S	Def			103
105	1	01:00	Buyer	Prop	Def		Economic Event		happens		Time		1	1	B,S	Def			103
106	1	01:00	Buyer	Prop	Def		Economic Event		happens		Location		1	1	B,S	Def			103
107	1	01:00	Buyer	Prop	Def		Economic Resource		have		Identifier	1	1	1	B,S	Def			
108	1	01:00	Buyer	Prop	Def		Economic Resource		bear	Value	Amount		1	n	B,S	Def			107
200	100	01:10	Buyer	Prop	Def		Buyer		have		Name	1	1	1	B,S	Ins			
201	101	01:10	Buyer	Prop	Exp		Buyer	Beneficiary	participate		Delivery		0	n	B,S	Obs			
202	100	01:10	Buyer	Prop	Def		Seller		have		Name	1	1	1	B,S	Ins			
203	101	01:10	Buyer	Prop	Exp		Seller	Responsible	participate		Delivery		0	n	B,S	Obs			
204	103	01:10	Buyer	Prop	Def		Delivery		have		Identifier	1	1	1	B,S	Ins			
205	105	01:10	Buyer	Prop	Def		Delivery		happen		Time		1	1	B,S	Ins			204
206	106	01:10	Buyer	Prop	Def		Delivery		happen		Location		1	1	B,S	Ins			204
207	104	01:10	Buyer	Prop	Def		Delivery		deliver		Product		1	n	B,S	Ins			204
208	107	01:10	Buyer	Prop	Def		Product		have		Identifier	1	1	1	B,S	Ins			
209	108	01:10	Buyer	Prop	Def		Product		bear	Value	Amount		1	n	B,S	Ins			208

Table 8.8 Knowledge base

8.3 UML

The Unified Modeling Language (UML) is today the de-facto standard modelling language for systems development. The language has been developed in the 1990s by the Object Management Group (OMG). UML offers nine diagram types in order to graphically describe an information system. Some diagram types describe the dynamic behaviour of the system (such as the Activity diagram); other diagram types define the structure of the system (Class diagram, Component diagram, Object diagram).

In order to improve the expressivity of UML, two extensions were added: Tagged Values and the Object Constraint Language (OCL). With Tagged Values arbitrary meta-values may be assigned to UML artefacts. OCL allows adding additional constraints to model elements.

UML Class diagrams contain the following artefacts:
- Classes, consisting of a (qualified) name, a set of attributes and a set of methods.
- Associations between Classes, having a name, a type, role names and cardinalities.

For adoption of more flexible methods to define B2B interfaces, such as the method described in this thesis, it is crucial that the information structures can be expressed in UML. In this section the elements of a B2B knowledge base are mapped on UML artefacts.

The knowledge base is populated with utterances. In actual B2B communication, utterances are grouped in messages or business documents. A business document bundles all utterances that are uttered by some party to another party at the same time.

Utterances have two faces: they contain instance data and they are a template or schema for future utterances, according to the Russian Doll pattern (see section 6.9). Here mainly the second face of the utterances is mapped to UML Class diagrams. UML Class diagrams describe the model or schema for future population with instance data.

Core proposition

8	9	10	11	12
Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name
Car		Have	License	Text
Car	Whole	Possesses	Part	Wheel

Table 8.9 Core proposition

UML supports for each association an association name and two role names. Usually the association name is represented as a noun, not as a verb.

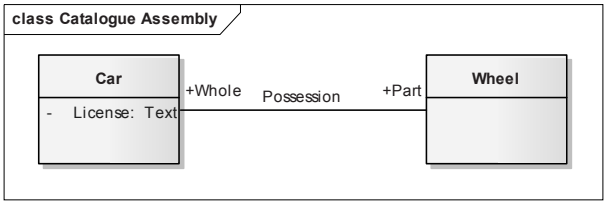


Figure 8.11 Association

In UML a distinction is made between associations between Classes (concepts) and associations with data types. The latter are called attributes and they appear inside the Class boxes. An example is the License attribute in figure 8.11. With attributes only the target role (the role of the attribute) can be defined, not the source role or the association type. Utterances that have a data type as their target role are mapped to UML attributes. When the target role is a concept, the utterance core is mapped to a UML association.

Instances are usually not modelled in UML. It is however possible to include attribute values in UML models in the way as shown in figure 8.12. If needed, a mapping can be made of Data Type values to this UML model element. In order to map individual entities, other than data type values, the entities are modelled as UML Classes.

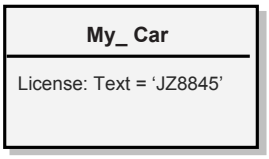


Figure 8.12 Attribute value

Identification

1	2	3	4
Utterance #	Based on Utterance #	Timestamp	Uttered by Party

Table 8.10 Identification

The administration of a UML model is not maintained in the model itself. Attributes such as Utterance #, Timestamp and Uttered by Party can be added in the UML model as (administrative) attributes to all classes. A UML Attribute can however not be attributed.

The Core Component specification (CCTS, [1]) allows UML elements to be registered in a registry. In CCTS ABIE's (Classes), BBIE's (Attributes) and ASBIE's (Associations)

are defined as Registry classes. Registry classes have a set of administrative properties, such as “Submitting Organization” and “Change Date” that can be mapped to Uttered by Party and Date (Timestamp). Utterance # may be mapped to the “Unique Identifier” of a Registry Class member.

The based on relationship is not natively supported. UML knows an inheritance relation, but it works in a different way. Instead of mentioning all properties on the higher level that may be relevant to lower level classes and instances, only those properties are modelled that are relevant to all subclasses. Those properties may however be optional: not all instances need to possess them. The based on relation should be modelled explicitly and used instead of the inheritance relation. In an existing model, the inheritance relation can simply be converted into a based on relation by adding properties to higher levels.

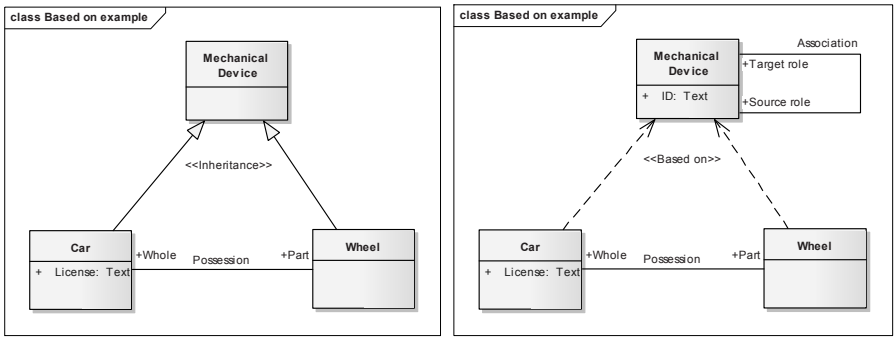


Figure 8.13 Inheritance versus Based On

Intention

5	6	7
Action	Stereotype	With Intention

Table 8.11 Intention

Intentions are not natively supported by UML. Event types with different intentions are modelled as different classes. The stereotype determines whether a transaction is changing the model (Definitions, Expansions) or populates the model (Instantiations, Observations). Actions determine if an utterance is processed in the model or not.

Cardinality

13	14	15
Part of ID #	Min Repetition	Max Repetition

Table 8.12 Cardinality

UML supports minimum and maximum repetitions, both for attributes and for associations. Identification schemes are not supported by UML.

Definition

16	17	18	19	20
To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #

Table 8.13 Definition

If Party and Intention are defined as attributes in the UML model, their value space may be limited. This will affect instantiations, but also classes that are based on the limited class. Allowance of certain stereotypes is not supported. It is possible to define a class as 'abstract class'. An abstract class may not be instantiated, only be based upon. Preconditions and Transactions are not defined in UML classes. They are defined in the UML diagrams that specify the dynamics of a system: State Machine diagrams and Activity Diagrams. In class diagrams it is possible to add OCL statements. OCL constraints may specify preconditions and transactions.

In the following table the right hand UML Class diagram in figure 8.13 is represented as a B2B knowledge base.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	1	0:10	S	Pro pose	Def	Assert	Mechanical Device		have	ID	Text	1	1	n	B,S	Def	Any		
101	2	0:10	S	Pro pose	Exp	Assert	Mechanical Device	Source role	associat es with	Target Role	Mechanical Device		0	n	B,S	Exp	Any		
102	100	0:10	S	Pro pose	Def	Assert	Car		have	License	Text	1	1	1	B,S	Inst	Any		
103	101	0:10	S	Pro pose	Exp	Assert	Car	Whole	possess	Part	Wheel		0	5	B,S	Obs	Any		102

Table 8.14 Definition and expansion

UML models may be expressed in XML using a language called XMI [5]. XMI has been designed to exchange models among tools, but by means of XSLT it should be possible to transform XMI models of messages and utterances to XML schemas. In such a way UML models could be fed into middleware that controls the message flow.

8.4 ERD/SQL

Entity Relationship diagrams are the de-facto standard for designing relational database structures. Relational databases may be interrogated and manipulated by means of the Structured Query Language SQL. SQL is well known and has widely been implemented.

In relational databases classes are defined as a relation between attributes. Each class and each attribute type has a name and attributes may serve as identifiers of instances of other classes (so called foreign keys).

A relational model consists of tables. Tables have columns and rows. A table in the relational model represents a set of phenomena in the real world that have a similar information structure. 'Similar', not 'the same', as in principal null values are allowed to indicate e.g. data elements that are not applicable for a specific instance. When the relational model is 'normalized', at a certain level (4^{th} normal form) null values are being removed so all instances of a relation indeed have the same information structure. Note that the relational model and the underlying mathematics have a 'bottom-up' orientation, reasoning from the relation between data elements and not from the perspective of semantic equivalence of the (real world) instances of objects or concepts.

As a table represents a set of phenomena, a row in the table represents one phenomenon. A phenomenon normally contains part of the data of an object or entity. Each row is identified by one or more key columns. The key columns are used to model relations between phenomena. So associations are modelled through equivalence of attributes. In the relational model one is forced to assign identifying attributes to relations. Inheritance is not supported. So it is not straightforward to refine relations on-the-fly, departing from more generic relations.

Core proposition

8	9	10	11	12
Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name

Table 8.15 Core proposition

In ERD Entities are equivalent to Concepts in a B2B knowledge base. A Relationship between Entities in ERD can be mapped to an Event in a B2B knowledge base. In ERD Entities and Relationships have names. No distinction is made between roles and verbs. Usually the Relationship name is a verb. To map the Role-Verb-Role construct to a Relationship name, v.v., the verb is used as a relationship name, together with a preposition or other indication of the role.

Identification

1	2	3	4
Utterance #	Based on Utterance #	Timestamp	Uttered by Party

Table 8.16 Identification

In ERD models, the administration of transactions is usually not modelled. It is assumed that the database in which the model is implemented has facilities (e.g. a system table) that record the transactions and that administer queries, commitments and roll backs. In order to implement a knowledge base into a data base system, the transaction information should be accessible by the system. If not, the Utterance#, Timestamp and Uttering Party must be part of the model.

In ERD subtyping and inheritance are not supported. In SQL, adding new tables requires different statements (ADD TABLE) than populating tables (UPDATE TABLE). Definitions in a B2B knowledge base lead to addition of tables. Expansions to lead to modification of tables (ALTER TABLE). Instantiations and Observations lead to updates of table contents. The logic of the transactions, obeying the knowledge base rules as specified in chapter 6, is to be coded in SQL. Such coding is beyond the scope of this thesis. However, the ORM meta-model in chapter 6 may be transformed into a database schema, offering the basis of a database that supports B2B knowledge bases.

Intention

5	6	7
Action	Stereotype	With Intention

Table 8.17 Intention

Actions play a role at the level of composing database transactions. Only utterances that have been Accepted (or that are Added, and thus are accepted implicitly) are fed to the database as transactions. Different stereotyped utterances lead to different types of transactions (see *Identification*). Intentions are not supported by ERD or SQL. They must be explicitly modelled, either by creating separate tables for entities with different intentions, or by adding the Intention as an attribute.

Cardinality

13	14	15
Part of ID #	Min Repetition	Max Repetition

Table 8.18 Cardinality

Cardinality and Identification are supported by ERD. Relations between entities may have some cardinality: a minimum and a maximum repetition. Repetition is indicated in diagrams by one of several notations: arrows, dots, crow-feet and/or numbers. Attributes may not repeat. If they need to, they should be modelled as separate entities. Identification is represented in ERD as primary keys. A set of attributes may serve as such key. Multiple sets of primary keys are not supported in ERD, but may be implemented in SQL.

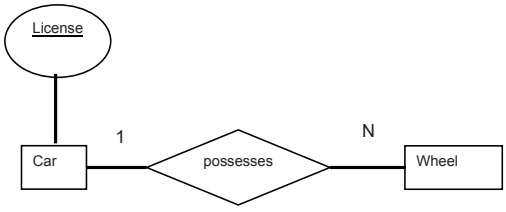


Figure 8.14 ERD

In figure 8.14 is illustrated how entities and relationships are represented in ERD. The Relationship is pictured as a diamond, entities as right angles and attributes as ellipses. Identifying attributes are underlined. Cardinalities are represented as numbers near the lines that connect entities with relationships.

Definition

16	17	18	19	20
To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #

Table 8.19 Definition

The 'inside' of the Russian doll can only be represented in ERD to define possible instantiations. As subtyping is not supported, possible lower level definitions cannot be represented. In SQL constraints to definitions can be coded, by defining transactions with additions and modifications of tables. Transactions (KB column 20) can be coded in SQL. Preconditions can be coded as well, but the SQL language is not very powerful. Many implementations support more powerful, proprietary ways to define preconditions.

8.5 CCTS/XML

UML was not primarily designed to model B2B conversations. It is a general purpose modelling language for information systems. Specifically to support B2B systems, UMM [6] was developed. UMM is a methodology, based on a meta-model expressed in UML. The meta-model includes a set of tagged values that are assigned specifically to support the definition of business conversations.

Standardization of B2B (especially XML-)messaging makes extensive use of UML. UN/CEFACT has issued a number of UML related specifications. The UML Profile for Core Components [2] defines a cross reference between CCTS and UML artefacts. The XML Naming and Design Rules [3] present a method to generate XML schema from a UML/CCTS model. The method has been implemented in several UML modelling tools.

CCTS is used to define standard B2B interfaces in a model driven fashion. Tools exist to automatically derive XML Schema from CCTS models.

The Core Component Technical Specification (CCTS [1]) is a technology and syntax neutral language for modelling B2B data models and messages. It is part of the ebXML set of specifications and used by UN/CEFACT to define standardized business libraries. A number of modelling and documentation tools support CCTS.

UN/CEFACT Core Components support a naming mechanism in which names reflect the 'based on' inheritance tree. An object or property name consists of a string of "qualifiers"

that defines the taxonomy of object classes and properties. When a subset of an object class or property is created, the new object (class) or property inherits the name of its parent, and gets an additional extra qualifier term. So an utterance may add qualifiers to name strings to define subsets of object classes or specializations of properties. With the name an object or qualifier always can be related to its parents or (super)classes. As the constraints on the parent also apply to the child, such naming is practical. The back side is that names tend to be very long. For human consumption, longer name strings may be replaced by a nickname or "business term".

CCTS is a UML profile. CCTS uses from UML a subset of Class diagramming. The formal mapping of CCTS artefacts on UML artefacts is defined in a separate specification: UPCC [2]. CCTS uses the UML artefacts Class, Attribute and Association.

Core proposition

8	9	10	11	12
Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name

Table 8.20 Core proposition

The CCTS equivalent of an object is called an Aggregated Core Component (ACC) or Aggregated Business Information Entity (ABIE). An ACC is an abstract object type that cannot be instantiated, only subtyped. An ABIE is normal object type that may be both subtyped and instantiated. An ACC or ABIE is identified by a Dictionary Entry Name (DEN). The DEN of an ACC or ABIE consists of an Object Class Term, a Property Term and a Representation Term. The terms are separated by a dot and a space. The Object Class Term is equivalent to the Source Concept Name. The Source Role Name, the Verb and the Target Role Name are mapped to the Property Term. Property Terms are in practice mostly adjectives or nouns. To translate a Property Term into a Role-Verb-Role combination vice versa needs some language processing. The exact ruling of such processing is outside the scope of this thesis. Basically the ACC or ABIE is first expressed in a full sentence, after which the verb and the (thematic) roles can be identified.

The Representation Term maps on the Target Concept Name. CCTS distinguishes two types of Representation Terms: Data Types and Associated Object Class Terms. An Associated Object Class Terms is the name of a concept. In the knowledge base structure the Target Concept Name can also be either the name of a data type or of another concept.

As an example, consider the Aggregate Core Component 'Authorization. Receiver. Party'. The sentence would read: 'Party receives authorization'. The Verb is 'Receive'.

The Party has the Beneficiary thematic role. The ‘Authorization’ is the Theme. The Knowledge Base structure is:

8	9	10	11	12
Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name
Authorization	Theme	Receive	Beneficiary	Party

Table 8.21 Example

Identification

1	2	3	4
Utterance #	Based on Utterance #	Timestamp	Uttered by Party

Table 8.22 Identification

CCTS defines the same based-on relation as proposed here for the B2B knowledge base. However, the CCTS based-on relation is at ACC/ABIE level, not on utterance level. Utterances or messages are outside the scope of CCTS. Another draft UN/CEFACT specification, CCMA [7] (abandoned and being replaced by a specification named Core Component Business Document Assembly), is to cover Message Assembly.

In CCTS, ABIE’s must be based on ACC’s. ACC’s have at least the same properties as ABIE’s that are based on them, but those properties may be defined on a higher level of abstraction, e.g. Vehicle (ACC) instead of Car (ABIE). So the based-on relationships in CCTS and in B2 knowledge bases may be mapped on one another.

As Message Assembly is outside the scope of CCTS, Utterance#, Timestamp and Uttered by Party are no CCTS constructs. UN/CEFACT has however published a specification of a generic message envelope: UN/CEFACT SBDH [8]. Message Identification, Timestamp and Message Sender are defined attributes of the SBDH. A Message is not the same as an Utterance. A Message spans multiple utterances; it may be equivalent to a Transaction. Sending Party and Timestamp may be copied from message level to utterance level. Numbering of utterances can be based on the sequence of the Utterances in the message.

Intention

5	6	7
Action	Stereotype	With Intention

Table 8.23 Intention

CCTS does not directly support Actions or Intentions. Data defined with Core Components is neutral versus its intention, unless intention is implicitly modelled in the semantic definition (e.g. 'Requested_ Delivery. Date. Date'). Intentions may specialize ABIEs. For instance 'Requested_ Delivery. Date. Date' may be a specialization of 'Delivery. Date. Date'.

The Intention may also be defined at message level. From the semantics of e.g. an 'Order Confirmation' it can be derived that a Delivery Date is the confirmed Date and not the Actual Date. Moore [9] has analysed the semantics of several EDI message types and has derived the implicit intentional Speech Act verb. Some industry messaging specifications, such as GS1 [10] and OAGis [11] have specified envelopes with an explicit verb. UN/CEFACT has drafted a message assembly specification with placeholders for explicit verbs, the specification however was never finalised. In most XML and EDI messaging specs the Intentional Verb is implicit in the semantics of the message types or of the business information entities. Mapping of the knowledge base intention to and from the messaging spec must be made per message and information type. Multiple intentions in a message (one per set of asserted concepts) is supported by the draft (but abandoned) CCMA.

Actions are also often implicitly defined in the message semantics or even in the process specification.

For the stereotyping a distinction should be made between meta-level (M1) stereotypes, such as Definitions, Expansions, States, Restrictions and Perceptions, and instance level (M0) stereotypes such as Instantiations and Observations. The distinction between Instantiations and Observations is often implicit to the semantics of the information entity or to be interpreted in the context of the message. The M1 level stereotypes are usually not exchanged in an operational message exchange. M1 level structures are being designed in standards committees or in bilateral off-line negotiation. They may be communicated and configured by means of CCTS library extractions, UML models or XML Schemas.

Cardinality

13	14	15
Part of ID #	Min Repetition	Max Repetition

Table 8.24 Cardinality

Identification schemes are not explicitly supported in CCTS. The context of the information must show which properties are identifying for an Aggregated Business Information Entity. Most ABIE structures start with the definition of an Identifier.

Minimum and maximum repetition is supported by CCTS.

Definition

16	17	18	19	20
To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #

Table 8.25 Definition

CCTS supports the same 'based-on' relationship as proposed for B2B knowledge bases. However, precise controlling this mechanism as offered by columns 16 through 20 is not supported by CCTS. These attributes may be regarded as extra constraints on the based-on mechanism. The attributes reside at M1 (meta-)level. They should be included in libraries, dictionaries, registries and schemas that are used to capture and exchange CCTS meta-information. If not directly supported by those libraries etc., they may be specified in annotations. Most library (etc.) structures offer a way for annotating the formal content, either structured (e.g. by means of tagged values) or unstructured (comments).

8.6 Example

In this section the meta-model of a B2B knowledge base and the mapping to modelling and exchange languages are illustrated by means of a simple example. The example illustrates how a data-model may be exchanged between business partners to be populated by business messages. More complete examples are shown in chapter 12.

Suppose a data model needs to be defined as the knowledge base structure of two parties. The data model is represented by the Buyer in ORM and by the Seller as a UML Class diagram.

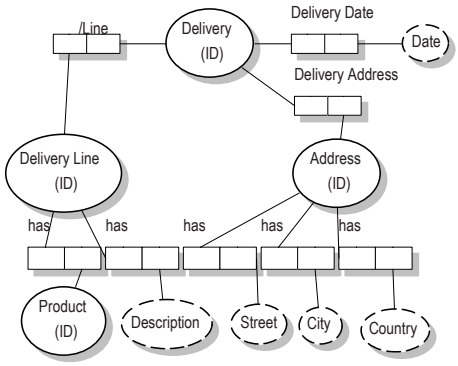


Figure 8.15 Example ORM diagram

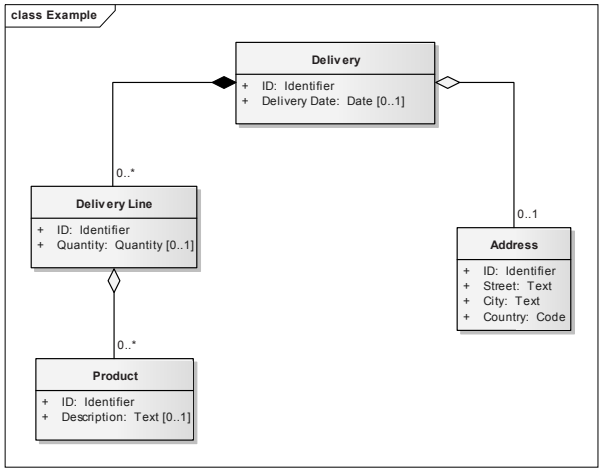


Figure 8.16 Example UML Class diagram

Note that the Buyer modelled a Description attribute to the Delivery Line, and the Seller to the Product, and that the Seller added a Quantity attribute to the Delivery Line.

The Buyer has created a database conform the ORM model by means of the following SQL statements:

```
CREATE TABLE Delivery(
        ID Identifier not null,
        DeliveryDate Date,
        DeliveryLine Identifier REFERENCES DeliveryLine(ID),
        DeliveryAddress Identifier REFERENCES Address(ID),
        PRIMARY KEY(ID));
CREATE TABLE DeliveryLine(
        ID Identifier NOT NULL,
        Product Identifier REFERENCES Product(ID),
        Description Text,
        PRIMARY KEY(ID));
CREATE TABLE Product(
        ID Identifier NOT NULL,
        PRIMARY KEY(ID));
CREATE TABLE Address(
        ID Identifier NOT NULL,
        Street Text NOT NULL,
        City Text NOT NULL,
        Country Code NOT NULL,
        PRIMARY KEY(ID));
```

Figure 8.17 SQL

Suppose the initial knowledge base exists of the generic "Business Concept" with an attribute named Data Type and a property with the Business Concept as its target.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
1	0	0:01	S	Add	Def	Propose	Business Concept		have	Attribute	Data Type		0	n	B,S	Def, Exp	Any		
2	0	0:01	S	Add	Def	Propose	Business Concept		have	Role	Business Concept		0	n	B,S	Def, Exp	Any		
3	0	0:01	S	Add	Def	Propose	Data Type		have	Content	Primitive Type		1	1	B,S	Def, Exp	Any		

Table 8.26 Initial knowledge base

This defines the Core Components:

Business Concept. Attribute. Data Type
Business Concept. Role. Business Concept
Data Type. Content. Primitive Type

This basic core ontology may be represented in an XML schema. This schema is generated from the model (either from a data base representation of the knowledge base or from an ORM or UML representation). The schema is exchanged between the business partners as the core ontology to build upon.

```
<xsd:complexType name="BusinessConcept">
    <xsd:annotation>
        <xsd:documentation xml:lang="en">
            <ccts:Acronym>ABIE</ccts:Acronym>
            <ccts:DEN>BusinessConcept. Details</ccts:DEN>
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:element name="Attribute" type="DataType">
            <xsd:annotation>
                <xsd:documentation xml:lang="en">
                    <ccts:Acronym>BBIE</ccts:Acronym>
                    <ccts:DEN>BusinessConcept. Attribute. DataType</ccts:DEN>
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
        <xsd:element name="Role" type="BusinessConcept">
            <xsd:annotation>
                <xsd:documentation xml:lang="en">
                    <ccts:Acronym>ASBIE</ccts:Acronym>
                    <ccts:DEN>BusinessConcept. Role. BusinessConcept</ccts:DEN>
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="DataType">
</xsd:complexType>
```

Figure 8.18 Initial XML schema

Note that the XML schemas in this example are only illustrations. They miss many XML elements and features, such as namespace declarations.

The Buyer defines the four specific Business Concepts as modelled in the ORM diagram:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
4	1	0:01	B	Add	Def	Propose	Delivery		have	ID	Identifier	1	1	1	B,S		Any		
5	1	0:01	B	Add	Exp	Propose	Delivery		deliver	Date	Date		0	1	B,S		Any		4
6	2	0:01	B	Add	Exp	Propose	Delivery		have	Line	Delivery Line		0	n	B,S		Any		4
7	2	0:01	B	Add	Exp	Propose	Delivery		deliver	Address	Address		0	1	B,S		Any		4
8	1	0:01	B	Add	Def	Propose	Delivery Line		have	ID	Identifier	1	1	1	B,S		Any		
9	1	0:01	B	Add	Exp	Propose	Delivery Line		have		Description		0	1	B,S		Any		8
10	2	0:01	B	Add	Exp	Propose	Delivery Line		have	Product	Product		0	1	B,S		Any		8
11	1	0:01	B	Add	Def	Propose	Address		have	ID	Identifier	1	1	1	B,S		Any		
12	1	0:01	B	Add	Exp	Propose	Address		have	Street	Text		1	1	B,S		Any		11
13	1	0:01	B	Add	Exp	Propose	Address		have	City	Text		1	1	B,S		Any		11
14	1	0:01	B	Add	Exp	Propose	Address		have	Country	Code		1	1	B,S		Any		11
15	1	0:01	B	Add	Def	Propose	Product		have	ID	Identifier	1	1	1	B,S		Any		

Table 8.27 Populated knowledge base

This results in the following Core Components:

Delivery_ Business Concept. ID_ Attribute. Identifier_ Data Type
Delivery_ Business Concept. Delivery Date_ Attribute. Date_ Data Type
Delivery_ Business Concept. Line_ Role. Delivery Line_ Business Concept
Delivery_ Business Concept. Address_ Role. Address_ Business Concept
Delivery_ Business Concept Line. ID_ Attribute. Identifier_ Data Type
Delivery_ Business Concept Line. Quantity_ Attribute. Quantity_ Data Type
Delivery_ Business Concept Line. Product_ Role. Product_ Business Concept
Delivery_ Business Concept Line. Description_ Attribute. Description_ Data Type
Address_ Business Concept. ID_ Attribute. Identifier_ Data Type
Address_ Business Concept. Street_ Attribute. Text_ Data Type
Address_ Business Concept. City_ Attribute. Text_ Data Type
Address_ Business Concept. Country_ Attribute. Code_ Data Type
Product_ Business Concept. ID_ Attribute. Identifier_ Data Type

The definitions and expansions are communicated from the Buyer to the Seller in an XML schema conform the schema in figure 8.19 (some lines have been truncated or removed for brevity).

```
<xsd:complexType name="Delivery">
    <xsd:annotation>
        <xsd:documentation xml:lang="en">
            <ccts:Acronym>ABIE</ccts:Acronym>
            <ccts:DEN>Delivery_ Business Concept. Details</ccts:DEN>
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:element name="ID" type="Identifier">
            <xsd:annotation>
                <xsd:documentation xml:lang="en">
                    <ccts:Acronym>BBIE</ccts:Acronym>
                    <ccts:DEN>Delivery_ Business Concept. ID_ Attribute. Identifier_
                        Data Type</ccts:DEN>
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
        <xsd:element name="DeliveryDate" type="Date"/>
            <xsd:annotation/>
        <xsd:element name="Line" type="DeliveryLine"/>
            <xsd:annotation>
                <xsd:documentation xml:lang="en">
                    <ccts:Acronym>ASBIE</ccts:Acronym>
                    <ccts:DEN>Delivery_ Business Concept. Line_ Role.
                        Delivery Line_ Business Concept</ccts:DEN>
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
        <xsd:element name="Address" type="Address">
            <xsd:annotation/>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<!--- Declarations of the other complexTypes: Delivery Line, Address and Product
        and of the Data Types -->
```

Figure 8.19 Example XML instance

On receiving of the XML schema, the Seller populates his knowledge base, so the knowledge bases of the two business partners are in sync. The Seller's middleware then attempts to map this meta-information to the structure of his information system as described by the UML diagram in figure 8.16.

The Seller notices that the Quantity attribute is missing from the proposed information structure and that Description is on Delivery Line level instead of Product level. He decides to propose to add the Quantity and to accept the (mis)location of the Description. He then must map the Description somehow to the Description attribute of the Product. How to handle mappings is described in chapter 10.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
16	1	0:11	S	Add	Def	Propose	Delivery Line		have	Quantity	Measure		1	1	B,S		Any		

Table 8.28 Knowledge base

The addition of the Quantity attribute to the Delivery Line is casted in an XML schema in a similar way as in figure 8.19 and communicated with the Buyer. The proposals of the Buyer are accepted by the Seller as well.

The Buyer receives the proposed schema and adds the Quantity to his system by means of an SQL statement.

```
MODIFY TABLE DeliveryLine(
        ID Identifier NOT NULL,
        Product Identifier REFERENCES Product(ID),
        Quantity Quantity,
        PRIMARY KEY(ID));
```

Figure 8.19 SQL

Now the knowledge base is ready to be populated with instance data. The Buyer first expresses his desire to receive Delivery 123 with product 55003.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
17	4	0:10	B	Add	Ins	Desire	Delivery			ID	123	1	1	1	S		Commit		
18	5	0:10	B	Add	Obs	Desire	Delivery		Deliver	Date	20090523		0	1	S		Commit		
19	6	0:10	B	Add	Obs	Desire	Delivery			Line	123-1		0	n	S		Commit		
20	7	0:10	B	Add	Obs	Desire	Delivery		Deliver	Address	87001		0	1	S		Commit		
21	8	0:10	B	Add	Ins	Desire	Delivery Line			ID	123-1	1	1	1	S		Commit		
22	16	0:10	B	Add	Obs	Desire	Delivery Line			Quantity	50		0	1	S		Commit		
23	10	0:10	B	Add	Obs	Desire	Delivery Line			Product	55003		0	1	S		Commit		
24	11	0:10	B	Add	Ins	Assert	Address			ID	87001	1	1	1	S		Commit		
25	12	0:10	B	Add	Obs	Assert	Address			Street	Koggekade 208		1	1	S		Commit		
26	13	0:10	B	Add	Obs	Assert	Address			City	Zwolle		1	1	S		Commit		
27	14	0:10	B	Add	Obs	Assert	Address			Country	NL		1	1	S		Commit		
28	15	0:10	B	Add	Ins	Assert	Product			ID	55003	1	1	1	S		Commit		

Table 8.29 Populated knowledge base

This information can be exchanged using an XML message conform the schema in figure 8.20.

```
<Delivery>
    <ID>123</ID>
    <DeliveryDate>20090523</DeliveryDate>
    <DeliveryLine>
        <ID>123-1</ID>
        <Quantity>50</Quantity>
        <Product>
            <ID>55003</ID>
        </Product>
    </DeliveryLine>
    <DeliveryAddress>
        <ID>87001</ID>
        <Street>Koggekade 208</Street>
        <City>Zwolle</City>
        <Country>NL</Country>
    </DeliveryAddress>
</Delivery>
```

Figure 8.20 XML instance

The stereotypes and intentions are assumed to be defined implicitly. These attributes should be added to the XML Schema, to take full advantage of the knowledge base features.

The resulting SQL is:

```
INSERT INTO Delivery(Id, DeliveryDate, DeliveryLine, DeliveryAddress)
VALUES (123, 20090523, 123-1, 87001)
INSERT INTO DeliveryLine(Id, Quantity, Product)
VALUES (123-1, 50, 55003)
INSERT INTO Address(Id, Street, City, Country)
VALUES (87001, Koggekade 208, Zwolle, NL)
INSERT INTO Product(Id, Description)
VALUES (55003, Rivet)
```

Figure 8.21 SQL

This example shows that the mechanisms to populate a B2B knowledge base, as described in this chapter allow to dynamically define metadata in a business relationship, using normal XML messaging and SQL interfaces. Metadata are mapped to models of information systems that have been defined in existing modelling languages such as UML and ORM.

8.7 Summary

In this chapter is shown that the structure of a B2B knowledge base can be mapped to existing modelling languages, such as UML, ORM or ERD, to database languages such as SQL and to exchange languages such as XML. The mappings made are illustrations; formal mapping between the meta-models of the various languages and the meta-model of the knowledge base is outside the scope of this thesis. In general it can be concluded that the B2B knowledge base structure is richer than each of the languages, but features, not supported by a language can often be modelled by means of additional (formal) annotations.

The mappings show that a B2B knowledge base does not need to be implemented as such. Instead, the usual tools for modelling and administrating systems and middleware can be used, and the XML (or EDI) messages, as standardized in many industries, need not to be changed.

If however all knowledge base features as described in this thesis need to be used to support business connections, some languages and tools need to be enhanced. Features such as stereotypes and intentions (but also the *based on* relationship) are not yet present in all languages used and need to be defined.

References

[1]. United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT Core Components Technical Specification Version 3.0, 2010
[2]. United Nations Centre for Trade Facilitation and Electronic Business: UML Profile for Core Components (UPCC), Version 1.0, 2008
[3]. United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT XML Naming and Design Rules Technical Specification, version 3.0, 2010
[4]. Halpin, ORM/NIAM Object-Role Modeling, Handbook on Information Systems Architectures, 2nd edition, eds P. Bernus, K. Mertins & G. Schmidt, Springer, Heidelberg, 2006.
[5]. Object Management Group: XML Metadata Interchange Specification (XMI), http://www.omg.org/spec/XMI/ISO/19503/PDF/
[6]. United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT's Modelling Methodology (UMM) UMM Meta Model – Foundation Module, 2006
[7]. United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT Core Components Message Assembly (CCMA) Technical Specification draft, rev. 1.10, 2007
[8]. United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT Standard Business Document Header (SBDH)
[9]. Moore: Categorizing automated messages, Journal of Decision Support Systems, Volume 22 Issue 3, March 1998
[10]. GS1 XML library: http://www.gs1.com/ecom/about/xml
[11]. Open Applications Group: OAGIS 8.0 Design Document, 2002

9 Initial B2B ontology

Summary

As elaborated in chapter 5, in the course of a B2B relationship a knowledge base is populated. The structure, or better: meta-structure, of that knowledge base is detailed in chapter 6. The structure enables business partners to add and define new concepts that are of relevance to their relationship. New concepts are based on concepts already present. The set and structure of concepts defined form the ontology, the world model or the universe of discourse of the trading partners.

To create an ontology from scratch, each time a new business partnership is established, is neither feasible nor desirable. As is shown in chapter 10, the information structure in many cases must be mapped to existing information systems and (internal) business processes. Mapping will be over complicated if with each trading partner a completely new structure of the knowledge base is negotiated. It is also unnecessary, because trading in economic spaces as we know them often follows the same patterns. The present practice however, where ontologies and inter-organizational processes are standardized in detail for entire industry sectors appears also not to be feasible, as was shown in chapter 1. A small initial generic business ontology and (not too detailed) extensions per industry sector seem the way to go.

An attractive starting point for an initial ontology is the REA ontology [1]. REA departs from the fundamental notion that the purpose of business transactions is to exchange resources that bear some value. This fundament makes REA attractive for an ontology in which all types of business transactions and business communication can be specialized.

The core of the REA ontology consists of three concepts: Resources, Events and Agents. REA is about business transactions between business partners or Agents. Agents exchange Resources (products, services and money). The actual exchange of a Resource is an economic Event. In this chapter an initial B2B ontology, based on REA, is further constructed and casted in the knowledge base structure of chapter 6.

9.1 Introduction

Many projects have developed ontologies. The resulting ontologies can be differentiated into top ontologies and domain ontologies. Top ontologies include CYC [2] and SUMA [8]. They attempt to define a set of basic concepts that are relevant to all domains. Domain ontologies include ontologies for geographic information [3], medicine [4] and industrial products [5] (among others).

A number of initiatives have developed business and enterprise ontologies [6]. Enterprise ontologies define the internal structure and processes of an enterprise. They do not focus on the communication between enterprises.

REA focuses on the exchange of resources between businesses. REA allows taking various perspectives: the perspective of one of the participating businesses, or a neutral perspective. The neutral perspective was chosen by ISO/IEC JTC1 SC30 that has chosen the REA ontology as the basis for Open edi [7].

In the sequel of this chapter the REA ontology is analysed, adapted and extended to serve as an initial ontology for any B2B relationship. It is represented in the table format that was introduced in chapter 6. As shown in chapter 8 that format can be converted into other modelling languages, such as ORM or UML. In chapter 10 an information system architecture is presented that allows application systems and middleware to take up the initial ontology and to refine and negotiate the ontology to support specific B2B processes.

9.2 Upper ontology

REA does not pretend to cover all aspects of human knowledge. REA is a domain ontology. REA specifies the concepts of the business and trading domain. The REA concepts must fit in a more generic ontology. One of the initiatives to define a generic ontology that may host specific domain ontologies such as REA is SUMO (Suggested Upper Merged Ontology) [8]. SUMO incorporated Wordnet [9] , which is a huge lexical database. Wordnet includes definitions, synonyms, hyponyms and hypernyms. SUMO additionally lists axioms and integrity rules.

The SUMO upper ontology makes a fundamental distinction between Physical and Abstract entities. Abstract entities are mental and mathematical constructs such as numbers, conditions and propositions. Physical entities include Objects (endurants) and Processes (perdurants).

The top levels of SUMO are represented in figure 9.1.

entity
- physical
  - object
    - self connected object
      - substance
      - corpuscular object
        - organic object
        - artifact
      - content bearing object
      - food
    - region
    - collection
    - agent
  - process
- abstract
  - quantity
  - attribute
  - set or class
  - relation
  - proposition
  - graph
  - graph element

Figure 9.1 SUMO Top level

A few observations can be made. SUMO seems somewhat biased towards a biological view on the world. The concept “Food”, for example, is defined as a direct child of “Self connected object”, while Food merely is to be regarded as a role of some other (organic or inorganic) object, or even as an economical product. The concept “Non-organic natural object” (such as a Stone) even seems to be missing altogether from the SUMO ontology (it is present in Wordnet though).

When fitting REA into an upper ontology we do not follow the SUMO hierarchy completely. It should be noted that an ontology such as SUMO is not a taxonomy: it is e.g. possible to have multiple inheritance. So it is allowed to place some concepts (also) in other branches of the inheritance tree.

All in all, SUMO offers the most complete and well thought through upper ontology that is available today. Somewhat adapted because of reasons just mentioned, the upper ontology to be used to host the REA ontology looks as presented in figure 9.2.

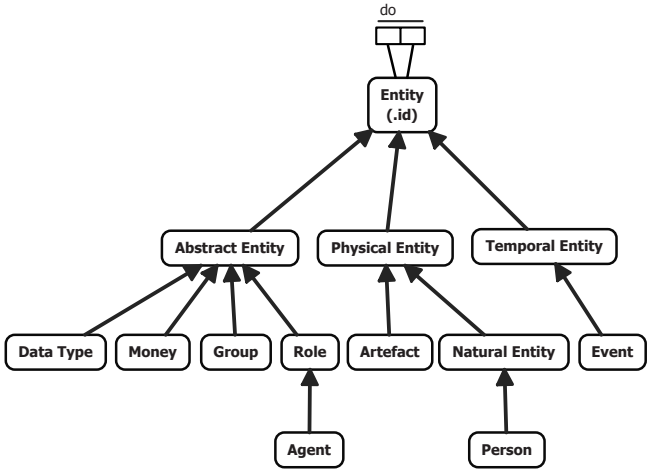


Figure 9.2 Top level ontology

The top-most division makes a distinction between Abstract entities, Physical entities and Temporal entities. Data types, Groups (sets, bags, classes), but also Roles and Money are Abstract entities. Money could have been defined as a "Right on part of the domestic product", but that seems too abstract to serve a purpose. Money is therefore here positioned as a high level (and important) concept in the B2B ontology. An Agent is defined as a role. A Role may be active or passive. Any (physical) entity may play passive roles: a location may e.g. play the role of the destination of a consignment. An Agent plays an active role and may be fulfilled by a Person or a Company. A Company will be defined as an organised group of persons.

Instead of the SUMO concept "Organic object", "Natural entity" is introduced here, as opposed to "Artefact". Natural entities include organic and inorganic objects. A Person is a Natural Entity.

9.3 Resources, Events and Agents

REA [1] is an ontology for business transactions. It was developed by McCarthy and Geerts, not primarily for electronic business to business communication, but for internal business accounting systems. Later REA has been used as the ontology for B2B communication systems as well [10].

REA departs from the fundamental notion that the purpose of business transactions is to exchange resources that bear some value. This fundament makes REA attractive for an ontology in which all types of business transactions and business communication can be specialized.

The core of the REA ontology consists of three concepts: Resources, Events and Agents. REA is about business transactions between business partners or Agents. Agents

exchange Resources (products, services and money). The actual exchange of a Resource is an economic Event.

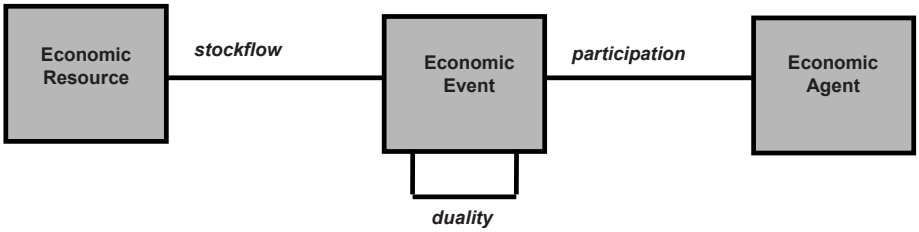


Figure 9.3 Basic REA ontology

The duality association between Economic Events means that in most cases two economic events with participation of the same two agents are related to each other: the exchange of (ownership and custody of) products and the exchange of money. There are however exceptions, such as gifts and swap deals (exchange of goods against goods).

The more complete REA picture looks as depicted in figure 9.4 [11].
It must be noted that REA is not a standard that is maintained by a standards body. REA is being discussed and developed in papers with various authority. Some of the limitations that are discussed in this thesis were covered by proposals to extent REA.

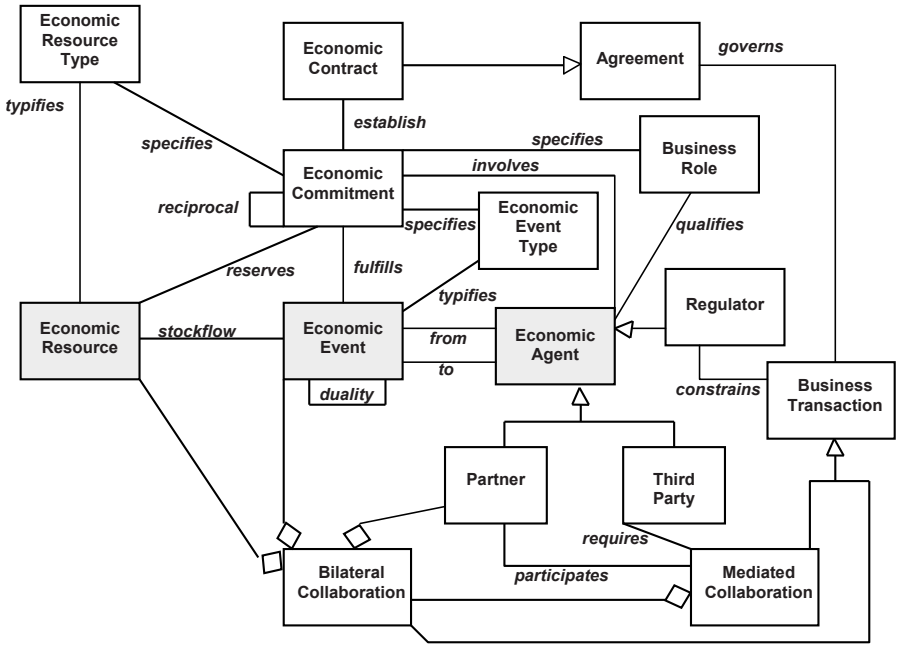


Figure 9.4 REA ontology (taken from [11])

The most important REA concepts in addition to Economic Resources, Economic Events and Economic Agents are Commitments and Contracts. Before an economic event happens it is negotiated between the economic agents. The negotiation results in a Contract that includes a number of Economic Events. The events (resource exchanges) within the contract should be in equilibrium. After all these events have happened, no residual obligations between the agents exist.

REA further makes a distinction between 5 phases in a business transaction [11]:

- Planning: In the Planning Phase, both the buyer and seller are engaged in activities to decide what action to take for acquiring or selling a good, service, and/or right.
- Identification: The Identification Phase pertains to all those actions or events whereby data is interchanged among potential buyers and sellers in order to establish a one-to-one linkage.
- Negotiation: The Negotiation Phase pertains to all those actions and events involving the exchange of information following the Identification Phase where a potential buyer and seller have (1) identified the nature of good(s) and/or service(s) to be provided; and, (2) identified each other at a level of certainty. The process of negotiation is directed at achieving an explicit, mutually understood, and agreed upon goal of a business collaboration and associated terms and

conditions. This may include such things as the detailed specification of the good, service, and/or right, quantity, pricing, after sales servicing, delivery requirements, financing, use of agents and/or third parties, etc.

- Actualization: The Actualization Phase pertains to all activities or events necessary for the execution of the results of the negotiation for an actual business transaction. Normally the seller produces or assembles the goods, starts providing the services, prepares and completes the delivery of good, service, and/or right, etc., to the buyer as agreed according to the terms and conditions agreed upon at the termination of the Negotiation Phase. Likewise, the buyer begins the transfer of acceptable equivalent value, usually in money, to the seller providing the good, service, and/or right.
- Post-Actualization: The Post-Actualization Phase includes all of the activities or events and associated exchanges of information that occur between the buyer and the seller after the agreed upon good, service, and/or right is deemed to have been delivered. These can be activities pertaining to warranty coverage, service after sales, post-sales financing such as monthly payments or other financial arrangements, consumer complaint handling and redress or some general post-actualization relationships between buyer and seller.

REA also makes a firm distinction between types (of Resources, Events and Agents) and the actual instances that are involved in the actual economic exchanges.

9.4 REA inspected and discussed

At a closer look resources are not simply goods and services. Resources are the economic values goods and services bear. In fact (in case of goods) not the goods themselves are a resource but the (ownership or custody) *rights* on the goods. In the initial ontology therefore distinction should made between the goods and the rights on the goods. Goods themselves need not to change when their ownership changes.

One can question what a right fundamentally is. A right is behaviour (towards or involving the goods) that is allowed (by the law or by the society) with some exclusivity. Such behaviour includes the usage or consumption of the goods. A service is also behaviour, but a service is behaviour of the service providing party that is of some benefit to the service receiving party. Handing over ownership, transportation and handing over custody are in fact services as well, so is a payment. They all involve some kind of behaviour.

Therefore an economic event can be typified as a form of behaviour. Behaviour may involve physical goods, rights on those physical goods, money, rights on intellectual property, etc. This notion potentially widens the REA scope to government services and governmental obligations. For many government services no payment directly related to the service is required. Conversely governments require business to perform activities for which they are not paying.

In the scope of this thesis, behaviour is always reflected in information that is stored in the knowledge base. We abstract from behaviour that is not relevant to the B2B relationship and is not visible in the knowledge base. Some behaviour consists only of a statement (utterance) made by a party in the knowledge base, e.g. a declaration of ownership transfer. Other behaviour (e.g. transportation) first happens in the physical reality and is then reported in the knowledge base, either by the party that performed the behaviour or by another party.

The present REA ontology lacks some concepts and structures that are essential to the (automation of) complex commercial and operational business relations today. In particular:

- REA does not include concepts or mechanisms to define the (legal, regulatory and technical) *context* of a business relation, which defines the boundaries (or the 'playfield' of the bilateral contracts to be negotiated and closed.
- A contract in REA is (by definition) a bundle of commitments to make economic events happen, while in practice many contracts just set the conditions for 'lower level' agreements (like call-offs) that really trigger the events.
- Duality and reciprocity are modelled in REA as one-to-one direct associations between Commitments, resp. Events, while in practice these associations are often many-to-many and are defined and controlled by the contract.
- REA supports the definition of *what* resources are exchanged, but not *when* they are and in what order (the business process). Although process definition is probably out of the REA scope, for e-business systems it is essential and the link with the REA concepts must be clear.
- Commitments in REA are defined in an absolute way, while in practice (and certainly during negotiations) commitments are often conditional, to other commitments to be accepted by the other party, or to events to happen.

In order to adapt the REA ontology for the purposes as stated in chapter 6 of this thesis, a more fundamental approach is followed, departing from human communication in general.

Stamper [12] has analysed human communication from the perspective of affordances and norms. His approach is called Semiotics. Affordance is the behavioural space humans allow each other. Someone's behaviour may enable or limit the behaviour of others. It limits or widens the affordance for others. Norms are mutually agreed rules that behaviour must adhere to, in order to have a disciplined society, culture or co-operation.

Applying semiotics to business transactions and mapping semiotic concepts to REA concepts, leads to redefinition of some of the REA concepts.

In REA one or more Resources are being exchanged in a business transaction. A Resource may consist of goods, may be a service, a right or an amount of money. At closer inspection however goods are not resources with value of themselves, but the affordances that the goods allow (the behaviour they make possible) may bear value.

More precisely, the right (ownership or custody) on goods affords the bearer of the right specific behaviour involving the goods. It is assumed that the right on that behaviour is of value for him.

The core of REA is the Economic Event. An Economic Events is the transfer of ownership and/or custody rights on a Resource from one economic agent to another. Resources can be defined as Roles of physical or abstract Entities (money may be defined as an Abstract Entity). A Right is allowed behaviour with relation to the Resource.

In the past money used to be the ownership right on a quantity of gold held by the bank that issued the money. Nowadays it is an abstract 'right' on the resources that an economy produces. For B2B purposes it suffices to define money as some abstract entity that (apparently) has value and can be exchanged against resources.

9.5 Ontology

Before the REA based B2B ontology is further detailed we should go back to the purpose of defining an initial ontology. The ontology is to initially populate the B2B knowledge base so it can be extended and populated by the communicating trading partners. As is shown in chapter 5, an utterance of a trading partner always represents an event. The event may be the utterance itself or the utterance may represent an event that happened (or is to happen) in reality. Events are communicated with an intentional annotation ('plan', 'commit', etc.).

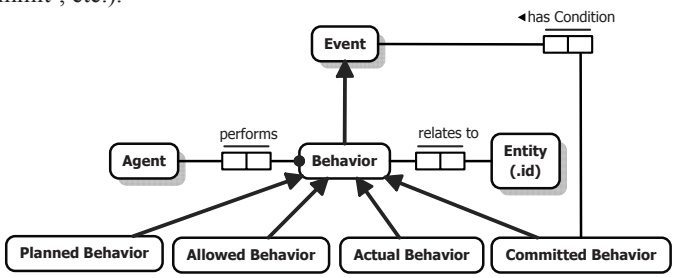


Figure 9.5 Types of behaviour

Behaviour of agents is a kind of event. Behaviour changes the state of the world (the universe of discourse). Behaviour may be Planned, Allowed, Committed or Actual. Those predicates are defined by means of the intention of the agent who utters it (or who performs the behaviour). Behaviour is represented in the knowledge base as utterances of agents. The 'Russian doll' pattern as described in chapter 6 is therefore also true for behaviour. Allowed behaviour creates the "affordance" for actual behaviour, so do planned and committed behaviour.

REA only distinguishes (firm) commitments and actuals; however in many cases business partners commit *conditionally* to perform certain behaviour. For example, a buyer may commit to pay under the condition that the goods are actually delivered. A condition is

usually behaviour of the other party, but it may also be some external event (for instance in case of an insurance policy).

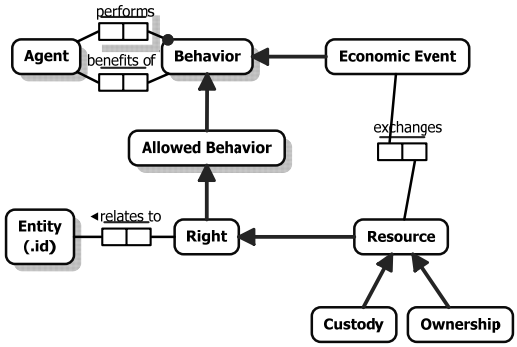


Figure 9.6 Adapted REA ontology

A Resource is the (custody or ownership) right on a good or physical entity, a service or money that can be bought or sold (or otherwise exchanged). A right can also be the right on a right (e.g. related rights on intellectual property). A right is allowed behaviour with respect to the good, service, money or right. Buying or selling a resource is the economic event of establishing or revoking the right. Ownership includes the right to transfer the ownership right to another agent. Buying or selling can be actualized by a declarative speech act of the owner. Custody is the right to use the good, for example to transport it. A service is behaviour of an agent that is beneficial to another agent.

The top level of the ontology is completed by defining an agent as a role, which is an abstract entity. The role can be fulfilled by a person or by an organization (legal entity). A role represents a collection of behaviour of that person or organization.

An intention and money are abstract entities as well. Abstract entities also comprise mathematical constructs such as numbers, sets, texts and alphabets. Physical entities include natural entities and artefacts. Temporal entities include events, but also moments (points in time) and periods.

This top level B2B ontology can be defined using the table structure of chapter 6. In this table the definitions of the data types used are not included. Data typing is elaborated in section 6.14.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
0	0	0:00	B	Prop	Def	Assert	Entity	Source Thematic Role	do	Target Thematic Role	Entity		1	n	B,S	All	All		
1	0	0:01	B	Prop	Def	Assert	Abstract Entity		have	Is Tangible Property	false		1	1	B,S	All	All		
2	0	0:01	B	Prop	Exp	Assert	Abstract Entity		do		Entity		1	n	B,S	All	All		
3	0	0:01	B	Prop	Def	Assert	Physical Entity		have	Is Tangible Property	true		1	1	B,S	All	All		
4	0	0:01	B	Prop	Exp	Assert	Physical Entity		do		Entity		1	n	B,S	All	All		
5	0	0:01	B	Prop	Def	Assert	Temporal Entity		have	Start_ Time	Date Time		1	1	B,S	All	All		
6	0	0:01	B	Prop	Def	Assert	Temporal Entity		have	End_ Time	Date Time		0	1	B,S	All	All		
7	0	0:01	B	Prop	Exp	Assert	Temporal Entity		do		Entity		1	n	B,S	All	All		
8	0	0:02	B	Prop	Res	Assert	Entity		do		Entity		1	n		none			

Table 9.1 Top level ontology

Row #0 is the basic statement that each entity has relations to one or more other entities. It is a 'boot strap' row: the only row in the knowledge base that is based on itself. Every other row is based on this row or on some other row. The verb "Do" was chosen for this basic utterance, because (in English) "Do" is the most generic verb, not to indicate the entity instance actually "does" something.

Row #1 through #7 divide the world into Abstract, Physical and Temporal entities. All other entities that later are defined must be based on one of these three types. The Entity is then restricted to limit further specialization in Utt #8: column 17 is set to 'none'. That means that it is not allowed to define a fourth type.

For each of the three entity types the statement that other entities may have a relation with it is repeated (rows #2, #4 and #7). This is needed because not all properties of the entity some entity is based on are automatically inherited. Properties that are defined must however be based on properties of the entity the entity is based on.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	2	0:10	B	Prop	Exp	Assert	Data Type		consists	Part	Data Type		0	n	B,S		All		
101	2	0:10	B	Prop	Def	Assert	Data Type	Theme	represent	Agent	Sign		1	1	B,S		All		
102	2	0:10	B	Prop	Def	Assert	Money		have	Quantity	Amount		1	1	B,S		All		
103	2	0:10	B	Prop	Def	Assert	Group	Agent	unite	Theme	Entity		1	n	B,S		All		
104	103	0:10	B	Prop	Def	Assert	Company		organise	Theme	Person		1	n	B,S		All		
105	2	0:10	B	Prop	Def	Assert	Role	Theme	play	Agent	Entity		1	n	B,S		All		
106	2	0:10	B	Prop	Exp	Assert	Role		do		Entity		0	n	B,S		All		
107	106	0:10	B	Prop	Def	Assert	Legal Entity	Theme	authorise	Agent	Legal Entity		1	1	B,S		All		
108	105	0:10	B	Prop	Def	Assert	Legal Entity	Role	act	Performer	Person OR Company		1	1	B,S		All		
109	105	0:10	B	Prop	Def	Assert	Agent	Role	play	Agent	Legal Entity		1	n	B,S		All		
110	106	0:10	B	Prop	Exp	Assert	Agent		do		Entity		0	n	B,S		All		
111	4	0:10	B	Prop	Def	Assert	Artefact	Patient	have	Manufacturing _ Agent	Agent		1	n	B,S		All		
112	4	0:10	B	Prop	Exp	Assert	Artefact		do		Entity		0	n	B,S		All		
113	4	0:10	B	Prop	Def	Assert	Natural Entity	Patient	have	Manufacturing _ Agent	Agent		0	0	B,S		All		
114	4	0:10	B	Prop	Exp	Assert	Natural Entity		do		Entity		0	n	B,S		All		
115	114	0:10	B	Prop	Def	Assert	Person		have	Surname_ Attribute	Text_ Data Type	1	1	1	B,S		All		
116	5	0:10	B	Prop	Def	Assert	Event		happen	Start_ Time	Date Time		1	1	B,S		All		
117	6	0:10	B	Prop	Def	Assert	Event		happen	End_ Time	Date Time		0	1	B,S		All		
118	7	0:10	B	Prop	Exp	Assert	Event		relate		Entity		0	n	B,S		All		
119	118	0:10	B	Prop	Def	Assert	Behaviour	Topic	perform	Agent	Agent		1	n	B,S		All		
120	118	0:10	B	Prop	Exp	Assert	Behaviour		relate		Entity		0	n	B,S		All		
121	119	0:10	B	Prop	Def	Assert	Service	Topic	perform	Agent	Agent		1	1	B,S		All		
122	120	0:10	B	Prop	Def	Assert	Service	Topic	have	Beneficiary	Agent		1	n	B,S		All		
123	116	0:10	B	Prop	Def	Assert	Service		happen	Start_ Time	Date Time		1	1	B,S		All		
124	120	0:10	B	Prop	Exp	Assert	Service		relate		Entity		0	n	B,S		All		

Table 9.2 REA

In rows #100 though #118 some basic concepts are defined, such as artefacts (as opposed to natural entities) and events. Behaviour is defined as an Event that is performed by an Agent. A Service is Behaviour that is of benefit to some (other) Agent.

The definition of a "Legal Entity": 'A role, authorized by a Legal Entity' seems and actually is a circular or recursive definition. No ultimate authority exists in modern economies. Any authority is (in a democracy) on its turn ultimately authorized by the members of the society, which are authorized (by the same authority) to vote. A 'Legal

Entity' Role may be played by natural Persons (which are natural entities) or by Companies, which are abstract entities (organized groups of persons).

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
201	121	0:20	B	Prop	Def	Assert	Economic Event	Topic	initiate	Agent	Agent		1	1	B,S		All		
202	122	0:20	B	Prop	Def	Assert	Economic Event		have	Beneficiary	Agent		1	1	B,S		All		
203	123	0:20	B	Prop	Def	Assert	Economic Event		happen	Time	Date Time		1	1	B,S		All		
204	124	0:20	B	Prop	Def	Assert	Economic Event		exchange	Theme	Resource		1	n	B,S		All		
205	124	0:20	B	Prop	Exp	Assert	Economic Event		relate		Entity		0	n	B,S		All		
206	119	0:20	B	Prop	Def	Assert	Right		have	Agent	Agent		1	1	B,S		All		
207	120	0:20	B	Prop	Def	Assert	Right		relate	Theme	Entity		0	n	B,S		All		
208	120	0:20	B	Prop	Exp	Assert	Right		have	Value	Amount		0	n	B,S		All		
209	206	0:20	B	Prop	Def	Assert	Resource		possess	Agent	Agent		1	1	B,S		All		
210	207	0:20	B	Prop	Exp	Assert	Resource		relate	Theme	Entity		1	1	B,S		All		
211	208	0:20	B	Prop	Exp	Assert	Resource		have	Value	Amount		0	n	B,S		All		
212	209	0:20	B	Prop	Def	Assert	Custody		possess	Agent	Agent		1	1	B,S		All		
213	210	0:20	B	Prop	Exp	Assert	Custody		relate	Theme	Entity		1	1	B,S		All		
214	211	0:20	B	Prop	Exp		Custody		have	Value	Amount		0	n	B,S		All		
215	209	0:20	B	Prop	Def	Assert	Ownership		Possess	Agent	Agent		1	1	B,S		All		
216	210	0:20	B	Prop	Exp	Assert	Ownership		relate	Theme	Entity		1	1	B,S		All		
217	211	0:20	B	Prop	Exp		Ownership		have	Value	Amount		0	1	B,S		All		

(Not all discriminating properties have been included in the definitions.

Table 9.3 REA

In rows #201 through #209 of table 9.3 the REA concepts Economic Event and Resource are defined. An Economic Event is the Behaviour of an Agent involving some Resource with another Agent as Beneficiary. A Resource is the Right of an Agent to perform Behaviour with respect to some Entity, which has some value. Two types of Resources are defined: Custody (#210 - #212) and Ownership (#213 - #215).

9.6 Trade transaction

In this section the extended (and partly reduced) REA ontology as proposed above is illustrated by means of the description of a trade transaction.

A trade transaction, whereby ownership and/or custody is transferred, is a process with a number of phases. Phases identified in REA are Planning, Identification, Negotiation, Actualization and Post-actualization. The Planning phase is local; it does not involve the business partner. In the planning phase the desire or requirement to acquire some

resource or to sell a resource is identified. In the Identification phase the desire, requirement or intention is communicated to the market.

The Identification phase usually starts with the publication of the (yet uncommitted) desire to exchange a resource against another resource, e.g. goods against money. The resources are identified or described somehow. Also the conditions under which the exchange will be take place are identified. Conditions may include the time and place of transfer (delivery conditions) and the period that may elapse between the two (or more) transfers (payment conditions).

Conditions may be imposed or defaulted by legislation. In fact a hierarchy exists, where (inter)national legislation defines the playfield where industry sectors fill in general process patterns and conditions. Individual organizations bilaterally specialize those patterns and conditions in contracts. See figure 9.8.

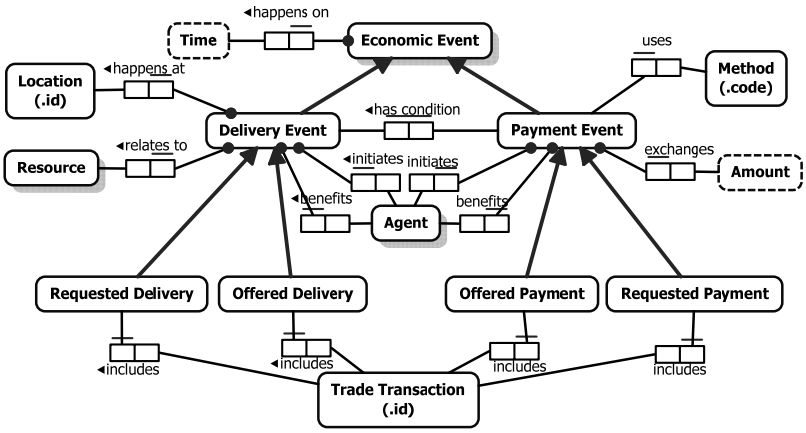


Figure 9.7 Hierarchy of communication patterns and conditions.

The identification phase starts with publication of uncommitted desires and intentions. Sellers state they are the owner of the resource identified and they wish to transfer that ownership under certain conditions (e.g. against an amount of money), buyers utter their desire to acquire ownership of the resource.

Figure 9.8 Trade transaction

A normal trade transaction consists of at least two economic events: usually a delivery event and a payment event. In the identification phase the buyer utters his requirements regarding. the events and the seller utters his offers. The event conditions include specifications of the product and the time, place and quantity of the delivery. They also include the amount of money the buyer is to pay and the time and method of payment. Product specifications as stated by the buyer are (ideally) functional specifications. The seller is to offer technical specifications that match the required functions.

At the end of the identification phase (at least) four entities have been defined: an offered delivery event together with a requested payment event, both specified by the seller, and a requested delivery event with an offered payment event specified by the buyer. In the negotiation phase the offers and requests are updated until they are reconciled (or not, in which case the transaction ends unsuccessfully).

A delivery event has a number of aspects:

- Specification of the resource to be transferred (incl. the quantity)
- Period in which the transfer is to take place
- Location where the transfer takes place

A payment event has:

- The amount to be paid
- The period in which payment is to be made (often stated relative to the delivery period)
- The payment method (e.g. cash or bank transfer).

At the end of the negotiation phase all aspects must have been reconciled:

- The technical specifications must meet the functional requirements
- The offered quantity must be equal or more than the requested quantity
- The offered delivery period must fall into the requested period
- The offered delivery location must be part of the requested location
- The amount requested must be lower than or equal to the amount offered by the buyer
- The payment period offered must be within the period requested

- The payment methods must match.

As soon as the required and the offered aspects match, the transaction shifts from the negotiation phase to the actualization phase. Both business partners are now committed to perform activities to make the actual transfer of products and money happen. The seller is committed to transfer the ownership and custody of the specified product to the buyer, at the agreed time and place; the buyer is committed to pay on delivery the agreed amount of money.

The transaction can be visualized with the following state transition diagram, with the state transitions of the four basic concepts: requested and offered delivery and payment events.

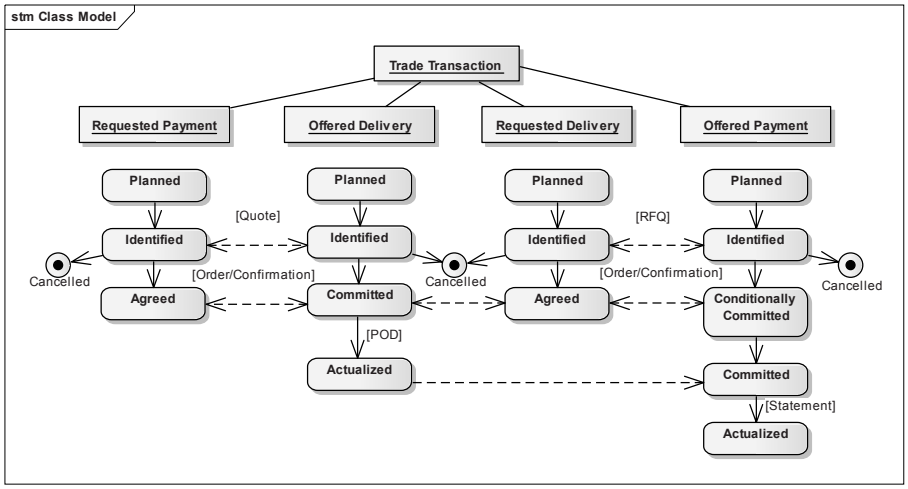


Figure 9.9 Trade transaction states

The states first shift from Planned to Identified. In the Identified state the event has been communicated and is subject to negotiation. In some contexts the seller then has committed to deliver under the conditions offered and under the condition that the buyer accepts the offered conditions. This means that the acceptance of the conditions by the buyer is binding the seller to deliver. In other contexts the offer is not yet binding, and the seller needs to confirm the buyer's acceptance.

In the negotiation phase both buyer and seller relief their requirements and offers until the offers meet the requirements, or the events are cancelled. When the offers and requirements comply, the transaction is agreed upon. The contract is then concluded. The seller has committed to deliver the products and the buyer has committed to pay upon delivery. The latter commitment is conditional. It is changed into an unconditional commitment when the delivery has taken place.

By publishing the intention to sell the resource, a seller must afford the buyer to set the next step, e.g. to place an order.

So a trade transaction is a process in which a number of bundled economic events develop from planned to actualized. This development is to be presented in a B2B knowledge base. In a normal trade transaction two types of events are negotiated and effectuated: a delivery event, in which ownership is transferred (and custody, but here we shall not make the distinction between ownership and custody) and a payment event. Ownership change is an event with the verb "Own" and a number of roles. The most important roles are the Beneficiary (the new owner) and the Theme (the resource that is owned). Other roles are the previous owner, time and place of ownership transfer and the conditions under which ownership transfer may take place.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
300	201	0:30	B	Prop	Def	Assert	Delivery		initiate	Agent	Agent		1	1	B,S		All		
301	202	0:30	B	Prop	Def	Assert	Delivery		have	Beneficiary	Agent		1	1	B,S		All		
302	205	0:30	B	Prop	Def	Assert	Delivery		happen	Date	Date		1	1	B,S		All		
303	203	0:30	B	Prop	Def	Assert	Delivery		exchange	Theme	Resource		1	1	B,S		All		
304	204	0:30	B	Prop	Def	Assert	Delivery		happen	Location	Address_Text		1	1	B,S		All		
305	201	0:30	B	Prop	Def	Assert	Payment		initiate	Agent	Agent		1	1	B,S		All		
306	202	0:30	B	Prop	Def	Assert	Payment		have	Beneficiary	Agent		1	1	B,S		All		
307	205	0:30	B	Prop	Def	Assert	Payment		exchange	Theme	Amount		1	1	B,S		All		
308	203	0:30	B	Prop	Def	Assert	Payment		happen	Date	Date		1	1	B,S		All		
309	204	0:30	B	Prop	Def	Assert	Payment		use	Payment Method	Payment Method_Code		0	1	B,S		All		
310	300	0:30	B	Prop	Res	Assert	Delivery		initiate	Agent	Agent		1	1	B		Request		
311	301	0:30	B	Prop	Res	Assert	Delivery		have	Beneficiary	Agent		1	1	B		Request		310
312	302	0:30	B	Prop	Res	Assert	Delivery		happen	Date	Date		1	1	B		Request		310
313	303	0:30	B	Prop	Res	Assert	Delivery		exchange	Theme	Resource		1	1	B		Request		310
314	304	0:30	B	Prop	Res	Assert	Delivery		happen	Location	Address_Text		1	1	B		Request		310
315	305	0:30	B	Prop	Res	Assert	Payment		initiate	Agent	Agent		1	1	S		Request		
316	306	0:30	B	Prop	Res	Assert	Payment		have	Beneficiary	Agent		1	1	S		Request		315
317	307	0:30	B	Prop	Res	Assert	Payment		exchange	Theme	Amount		1	1	S		Request		315
318	308	0:30	B	Prop	Res	Assert	Payment		happen	Date	Date		1	1	S		Request		315
319	309	0:30	B	Prop	Res	Assert	Payment		use	Payment Method	Payment Method_Code		0	1	S		Request		315

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
320	300	0:30	B	Prop	Res	Assert	Delivery		initiate	Agent	Agent		1	1	S		Commit		315
321	301	0:30	B	Prop	Res	Assert	Delivery		have	Beneficiary	Agent		1	1	S		Commit		320
322	302	0:30	B	Prop	Res	Assert	Delivery		happen	Date	Date		1	1	S		Commit		320
323	303	0:30	B	Prop	Res	Assert	Delivery		exchange	Theme	Resource		1	1	S		Commit		320
324	304	0:30	B	Prop	Res	Assert	Delivery		happen	Location	Address_Text		1	1	S		Commit		320
325	305	0:30	B	Prop	Res	Assert	Payment		initiate	Agent	Agent		1	1	B		Commit		
326	306	0:30	B	Prop	Res	Assert	Payment		have	Beneficiary	Agent		1	1	B		Commit		325
327	307	0:30	B	Prop	Res	Assert	Payment		happen	Date	Date		1	1	B		Commit		325
328	308	0:30	B	Prop	Res	Assert	Payment		exchange	Theme	Amount		1	1	B		Commit		325
329	309	0:30	B	Prop	Res	Assert	Payment		use	Payment Method	Payment Method_Code		0	1	B		Commit		325
330	300	0:30	B	Prop	Res	Assert	Delivery		initiate	Agent	Agent		1	1	S		Assert	325	
331	301	0:30	B	Prop	Res	Assert	Delivery		have	Beneficiary	Agent		1	1	S		Assert		330
332	302	0:30	B	Prop	Res	Assert	Delivery		happen	Date	Date		1	1	S		Assert		330
333	303	0:30	B	Prop	Res	Assert	Delivery		exchange	Theme	Resource		1	1	S		Assert		330
334	304	0:30	B	Prop	Res	Assert	Delivery		happen	Location	Address_Text		1	1	S		Assert		330
335	305	0:30	B	Prop	Res	Assert	Payment		initiate	Agent	Agent		1	1	B		Assert	330	
336	306	0:30	B	Prop	Res	Assert	Payment		have	Beneficiary	Agent		1	1	B		Assert		335
337	307	0:30	B	Prop	Res	Assert	Payment		happen	Date	Date		1	1	B		Assert		335
338	308	0:30	B	Prop	Res	Assert	Payment		exchange	Theme	Amount		1	1	B		Assert		335
339	309	0:30	B	Prop	Res	Assert	Payment		use	Payment Method	Payment Method_Code		1	1	B		Assert		335

Table 9.5 Trade transaction

In table 9.5 the meta-information of the trade transaction has been specified. First, a generic (column 6) Delivery and a Payment are specified in rows #300 - #309. Then the definitions are restricted to allow only certain combinations of agents and intentions to instantiate the definitions. Column 18 (With allowed intention) determines the distinction between requested, offered and actual deliveries and payments. The requested, offered and actual delivery and payment are each presented as a transaction (column 20). Actual payment has been made conditional to actual delivery and actual delivery to a commitment to pay.

Initial B2B ontology

Based on the meta-data in table 9.5 an actual trade transaction may be carried out. This is illustrated in table 9.6.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
400	213	20:10	S	Add	Ins	Assert	123_ Ownership		possess	Agent	S		1	1					
401	214	20:10	S	Add	Obs	Assert	123_ Ownership		relate	Theme	123_ Product		1	1					
402	310	20:20	B	Add	Inst	Request	456_ Delivery		initiate	Agent	S								
403	311	20:20	B	Add	Inst	Request	456_ Delivery		have	Beneficiary	B								
404	312	20:20	B	Add	Inst	Request	456_ Delivery		happen	Date	2010-09-01								
405	313	20:20	B	Add	Inst	Request	456_ Delivery		exchange	Theme	123_ Ownership								
406	314	20:20	B	Add	Inst	Request	456_ Delivery		happen	Location	8017KJ Zwolle								
407	315	20:40	S	Add	Inst	Request	456_ Payment		initiate	Agent	B								
408	316	20:40	S	Add	Inst	Request	456_ Payment		have	Beneficiary	S								
409	317	20:40	S	Add	Inst	Request	456_ Payment		exchange	Theme	EUR 500								
410	318	20:40	S	Add	Inst	Request	456_ Payment		happen	Date	2010-10-11								
411	319	20:40	S	Add	Inst	Request	456_ Payment		use	Payment Method	BT								
412	320	20:40	S	Add	Inst	Commit	456_ Delivery		initiate	Agent	S								
413	321	20:40	S	Add	Inst	Commit	456_ Delivery		have	Beneficiary	B								
414	322	20:40	S	Add	Inst	Commit	456_ Delivery		happen	Date	2010-09-11								
415	323	20:40	S	Add	Inst	Commit	456_ Delivery		exchange	Theme	123_ Product								
416	324	20:40	S	Add	Inst	Commit	456_ Delivery		happen	Location	8017KJ Zwolle								
417	325	20:50	B	Add	Inst	Commit	456_ Payment		initiate	Agent	B								
418	326	20:50	B	Add	Inst	Commit	456_ Payment		have	Beneficiary	S								
419	327	20:50	B	Add	Inst	Commit	456_ Payment		happen	Date	2010-10-11								
420	328	20:50	B	Add	Inst	Commit	456_ Payment		exchange	Theme	EUR 500								
421	329	20:50	B	Add	Inst	Commit	456_ Payment		use	Payment Method	BT								
422	330	2010-10-10	S	Add	Inst	Assert	456_ Delivery		initiate	Agent	S								
423	331	2010-10-10	S	Add	Inst	Assert	456_ Delivery		have	Beneficiary	B								
424	332	2010-10-10	S	Add	Inst	Assert	456_ Delivery		happen	Date	2010-10-10								
425	333	2010-10-10	S	Add	Inst	Assert	456_ Delivery		exchange	Theme	123_ Product								

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
426	334	2010-10-10	S	Add	Inst	Assert	456_ Delivery		happen	Location	8017KJ Zwolle								
427	335	2010-10-09	B	Add	Inst	Assert	456_ Payment		initiate	Agent	B								
428	336	2010-10-09	B	Add	Inst	Assert	456_ Payment		have	Beneficiary	S								
429	337	2010-10-09	B	Add	Inst	Assert	456_ Payment		happen	Date	2010-10-09								
430	338	2010-10-09	B	Add	Inst	Assert	456_ Payment		exchange	Theme	EUR 500								
431	339	2010-10-09	B	Add	Inst	Assert	456_ Payment		use	Payment Method	BT								

Table 9.6 Example

In utterances 400-401 the Seller asserts he is the owner of the resource. Then the Buyer requests for delivery. The seller states his price and offers to deliver under the stated conditions. In utterances #417-421 the Buyer commits to pay, thereby ordering the goods by fulfilling the precondition for actual delivery. Actual delivery and payment are reported in Utt # 422-431.

9.7 Conclusion

In this chapter a core ontology is defined for inter-organizational trade. The ontology is based on the REA ontology, with some adaptations. The ontology can serve as a starting point for (one-off) trade between trading partners from different industry sectors or for the basis of a more specific ontology for trading in a specific sector or between partners that have an enduring relationship.

References

[1]. Geerts, McCarthy (2000) The Ontological Foundation of REA Enterprise Information Systems. Working paper, Michigan State University

[2]. http://www.cyc.com/platform/opencyc

[3]. Fonseca, Egenhofer (1999): Ontology-Driven Geographic Information Systems, 7th ACM Symposium on Advances in Geographic Information Systems,Kansas City, MO

[4]. Pisanelli (ed.). Ontologies in Medicine. Volume 102 Studies in Health Technology and Informatics. IOS Press, 2004

[5]. ISO 10303 Standard for the Exchange of Product model data (STEP), International Organisation for Standardization

[6]. Bertolazzi, Krusich, Missikoff (2001): An Approach to the Definition of a Core Enterprise Ontology: CEO. Paper presented at the OESSEO 2001, International Workshop on Open Enterprise Solutions: Systems, Experiences, and Organizations

[7]. ISO FDIS 15944-1 – Operational Aspects of Open-edi for implementation, International Organisation for Standardization.

[8]. http://www.ontologyportal.org/

[9]. http://wordnet.princeton.edu/
[10]. Huemer, Motal, Schuster, Werthner:"From Economic Drivers to B2B Process Models: a Mapping from REA to UMM", in: "Business Information Systems 13th International Conference, BIS 2010, Berlin, Germany, May 3-5, 2010. Proceedings", Springer Berlin Heidelberg, (2010), ISBN: 978-3-642-12813-4; pp. 119 - 132.
[11]. Geerts, L. G., McCarthy, E. W. (2002, Vol.3) An Ontological Analysis of the Primitives of the Extended-REA Enterprise Information Architecture. The International Journal of Accounting Information Systems
[12]. Stamper et al. (2004): Semiotic Methods for Enterprise Design and IT Applications. Proceeding of the 7th International Workshop on Organisational Semiotics

10 Business goals, profiles and negotiation of conditions

Summary

The B2B architecture that is defined in this thesis allows business partners to present their capabilities in a common knowledge base and to state their goals and the requirements they have with regard to the business relationship in the same knowledge base. Goals, requirements and capabilities are then matched and the matching result into a process flow definition and into information definitions.

The matching mechanism of goals, requirements and capabilities is descibed in this chapter.

10.1 Introduction

In chapter 7 a mechanism is defined to specify B2B processes and to store the definitions in a B2B knowledge base. In the examples it was shown that the process definition evaluates in the course of the process itself. It was not shown explicitly what the process or process patterns are that partners follow to define processes. This 'meta process' is the topic of this chapter.

The examples in chapters 6 and 7 show one-off establishment of business relations to support one business transaction. Though most business relations are not eternal, they usually span multiple transactions. In that case it is not efficient or desirable to re-negotiate the business process per transaction. To optimise the coordination of operational (e.g. logistic) activities, it is better to agree on a business process that supports transactions over a longer period of time. The meta-process patterns that describe how that negotiation may be carried out are described in section 10.4.

The mechanisms described in chapters 6 and 7 are recursive. They allow to reason over concepts *and* instances, over processes *and* meta-processes. So the same mechanisms are used for process negotiation as for the negotiation about a specific transaction.

10.2 B2B Processes and business policy

In previous chapters the structure and functioning of B2B systems is detailed. The elements were identified of languages that are used to specify a B2B interface and a mechanism was presented to enhance such an interface in the course of a business conversation. In chapter 8 mappings were presented to a number of languages that currently are used to specify business information systems and to specify B2B protocols.

In this chapter it is shown how organizations may use those mechanisms to negotiate with their (potential) trading partners how the B2B interface is shaped. In other words, how the requirements of the organization with regard to the B2B process is derived from the organizations' private requirements and how the combination of the organizations'

objectives and those of the partners may (or may not) lead to successful (electronic) business.

Requirements engineering for information systems is usually targeted on the private systems of an organization. Many methods and presentation structures such as diagrams, tables and worksheets have been developed to guide an organization in the process of gathering and structuring requirements and to translate those requirements into an effective information system.

Architectural documents describe the information management functions of an organization and assign the various aspects to responsibilities and internal processes. Separation of concern about the aspects makes the entire functional area manageable. In many architectural approaches the area is layered into objectives, functions and technology.

In the various cells of an information management architecture different notations are used to structure and communicate the architectural decisions and the relevant aspects of the information system. The most frequently used notation methods are:

- Free, semi structured or structured language. Many requirement documents contain statements of responsible business people or business analysts. These statements may be structured or semi structured. SBVR for example describes rules for the structuring of a natural language as English to produce unambiguous and clear requirement in the form of definitions, vocabularies and business rules.
- Structure diagrams that picture aspects of the organization that are static, such as the organizational structure, the information structure and the technical infrastructure. UML Class diagrams, ORM and ERD diagrams are examples of diagrams that may present the information structure.
- Behavioural diagrams picture the dynamics of an organization, notably its work flow and business processes. UML Activity diagrams, Petri nets and state charts are examples.

To derive the B2B interface from the organizations' architecture, two major issues must be inspected:

- The relation between the company's policy and architecture to its external behaviour as demonstrated at a B2B interface (the 'what')
- The structuring of the B2B conversation as derived from the requirement specifications, business rules, information structures and internal process descriptions (the 'how').

Usually the strategy and policy of an organization is derived from the objectives of its stakeholders and from the capabilities the organization has. Capabilities include organizational knowledge, money, technology, trade relations, etc. Porter has analysed how the added value of enterprises in supply chains may be assessed. Gordijn has attempted to develop a method (e^3) [1] to structure the value added in order to specify business processes and the information architecture.

In general though, entrepreneurs take many aspects into account when developing the policy for their company. These aspects include risk, trust, gut feeling, market developments and capabilities. They usually assess these aspects intuitively. Without doubt this assessment may very well be supported by techniques offered by business management science. That is however outside the scope of this thesis.

Here we assume that some policy exists that determines the companies' products and processes. Whenever some new process violates the policy, a responsible manager can be consulted to decide whether the policy is adapted to include the new process, or not. So business policy is not regarded as law casted in stone.

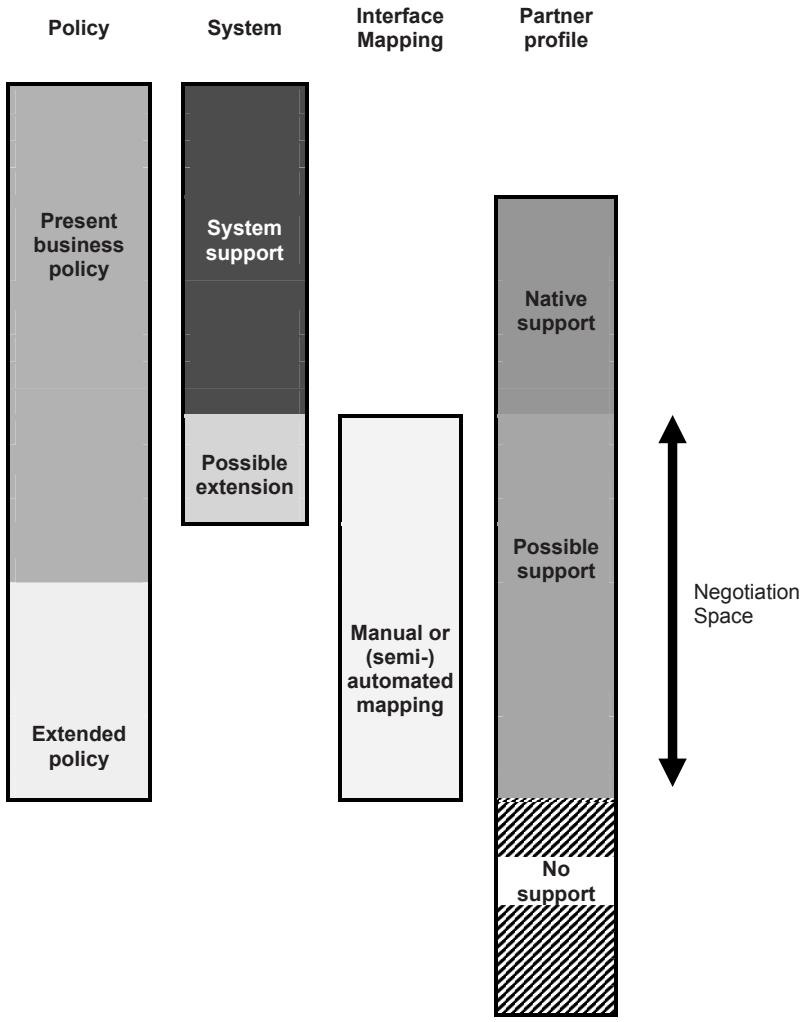


Figure 10.1 Policy, systems and B2B processes

In figure 10.1 is illustrated how a company reacts on requests from (potential) trading partners to participate in a B2B process. The picture may be interpreted as representing a list of statements or business rules. The company has some policy with regard to the products it offers, the market it services, the way it operates etc. This policy is represented as the blue bar, as subset of the business rule list. Part of this policy is implemented in and supported by one or more automated information systems (red). Another part is handled manually (yellow). When some trading partner requests the

company to participate is some B2B process, part of the requested functionality may be natively supported by both policy and the information system (green). If part of the request is not supported by the present policy, the management may decide to extend the policy (the partnership may be a business opportunity). The part of the request that is not directly supported by the information system may lead to adaptation of the information system and when that is not feasible it will be handled manually or by adapting the interface mappings. Interface mapping may be handled by middleware and may be semi-automated.

Here a distinction is made between the information system (database system with input/output functions, workflow system) and middleware that performs the mapping between data sent/received to/from trading partners and the information system.

If requested processes are natively supported by the information system, they can probably be handled more cost effectively than processes that need manual or semi-automated pre- or post-processing and mapping. So in the negotiations the company will try to keep the external processes within the red area. By extending the policy and adapting manual procedures and mapping functions, the company is however prepared to ultimately accept to participate in processes that include the yellow block as well. This negotiation space is indicated in the figure.

For each enterprise the relative sizes of the coloured bars are different. The sizes are determined by market power, company strategy (cost leader or niche player), systems architecture, etc. Some organizations (e.g. governmental organizations) will not show any flexibility in adapting to requirements of trading partners, for other organizations flexibility is the core strategy.

Profiles do not always need to be derived from business policy rules or be specifically constructed. Large parts of profiles may be published by standards organizations or industry sector bodies. For instance a core ontology may be (and is) published by IEEE, which is a standards organization. UN/CEFACT is another standards body that is candidate for generic trading profiles. Specific industries, such as the metal industry or the chemical industry, may publish reference profiles that are specific for their environment.

10.3 Process negotiation

A few major negotiation patterns can be identified. One of the major patterns is profile matching. Some organizations are (for very good reasons) relatively rigid in the way they can handle inter-organizational processes. They demand that the process to negotiate fits in one of their rigid patterns.

Profiles for business processes are defined in the business policy. Business policy is expressed in (formal or informal) business rules. A business rule consists of a statement, assigning properties to concepts or entities, with quantification and some deontic value. The deontic value (“It is necessary that ...”, “It is obligatory that ....”, “It is permitted that

...") resembles closely the 'intention' in B2B utterances, as expressed by speech act verbs. Business rules and B2B utterances, as they may be added to a B2B knowledge base, can therefore easily be converted in one another.

The business rules that an organization has derived from its policy and that are related to its external processes are included in a profile. The profile is a draft knowledge base in which business rules are represented as B2B utterances. Business rules and utterances limit the behaviour of the partners and the properties of the concepts they may talk about. They serve like filters on the affordances of the partners. The intersection of two sets of profiles defines the concepts and the processes that are valid in the B2B relationship.

Profile matching may lead to a viable set of B2B processes or to nothing. In the latter case the B2B relationship ends. If, for example, company A (the seller) has included in his profile that he wants to be paid in advance and company B (the buyer) has added the business rule that he pays 30 days after delivery, the profiles are incompatible and no trade relationship can start.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	0:10	S	Prop	Def	Assert	Payment		happen	Date	Date		1	1	B	Obs	Assert		
101	20	0:10	S	Prop	Def	Assert	Delivery		happen	Date	Date		1	1	S	Obs	Assert	100	
100	10	0:20	B	Accept	Def	Assert	Payment		happen	Date	Date		1	1	B	Obs	Assert		
101	20	0:20	B	Rej	Def	Assert	Delivery		happen	Date	Date		1	1	S	Obs	Assert	100	
102	10	0:20	B	Prop	Def	Assert	Delivery		happen	Date	D_ Date		1	1	S	Obs	Assert		
103	100	0:20	B	Prop	Res	Assert	Payment		happen	Date	D_ Date + 30		1	1	B	Obs	Assert	102	
102	50	0:30	S	Rej	Def	Assert	Delivery		happen	Date	D_ Date		1	1	S	Obs	Assert		
103	100	0:30	S	Rej	Res	Assert	Payment		happen	Date	D_ Date + 30		1	1	B	Obs	Assert	102	

Table 10.1 Failing profile matching

Matching of pre-defined profiles may not always lead to the market position an entrepreneur intends to conquer. He may wish to make his business rules less strict for some potential business partners. In that case the negotiation pattern is more applicable than the profile matching pattern. In the negotiation pattern proposals are being matched rather than profiles. When a match does not lead to a viable B2B process, the partners have the opportunity to relief their business rules and attempt to reach a match after all.

Why negotiate a process and not simply standardize it? One reason of course is that different business sectors and even different business relations have different

requirements to the process choreography. The requirements depend on stability of the relationship, trust levels, power balance, legislation, value of the resources, information quantity, logistic speed, uncertainty, to name a few criteria.

10.4 Negotiation patterns

In this section it is shown how profile matching and negotiation may be represented in the table of a B2B knowledge base. In actual business conversations each utterance is added to the knowledge base as a fact with the proper intentional value. Future utterances then may be based on it. In a process negotiation it must be possible to add utterances tentatively. Future utterances may only be based on such proposed utterance after it was accepted by the other party. For this function the "Action" column is used. In an actual conversation the Action always has the value "Add". Utterances (in fact opinions of the trading partners) are added to the knowledge base. In the process negotiation phase the values of the Action cell may be "Propose", "Accept" or "Reject". A Proposed utterance can only be used to base future utterances on it after it is Accepted by the other party. When it is Rejected it is not valid any longer.

A simple negotiation may serve as an example.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	50	0:01	B	Prop	Def	Assert	Rivet		Have	Length	Length_ Measure ment	2	1	1	B,S		All		
101	50	0:10	S	Rej	Def	Assert	Rivet		Have	Length	Length_ Measure ment		1	1	B,S		All		
102	50	0:10	S	Prop	Def	Assert	Rivet		Have	Nominal_ Length	Length_ Measure ment	2	1	1	B,S		All		
103	50	0:10	S	Prop	Def	Assert	Rivet		Have	Effective_ Length	Length_ Measure ment		1	1	B,S		All		
104	50	0:20	B	Acc	Def	Assert	Rivet		Have	Nominal_ Length	Length_ Measure ment	2	1	1	B,S		All		
105	50	0:20	B	Acc	Def	Assert	Rivet		Have	Effective_ Length	Length_ Measure ment		1	1	B,S		All		

Table 10.2 Negotiation

In utterance #100 the Buyer proposes to define a Rivet, based on the definition of a more generic artefact, which is stated in utterance #50 (utterance #50 is not shown in this example). The Buyer proposes to assign the 'Have Length' property to the Rivet. The Seller however does not accept this proposal (#101) and counter proposes two other measurement properties: Have Nominal_ Length and Have Effective_ Length. These proposals are accepted by the Buyer. In the sequel of the conversation the trading partners

now may base other definitions, expansions, states and observations on this agreed definition.

To negotiate a definition, one of the partners utters a proposal, while the other can either accept the proposal, or reject it and replace it by a counter proposal. A definition is only valid (and may be used in actual processes) after it has been accepted.

In column 16 (To utter by Party) is listed which party may utter the utterances based on the definition. A proposed definition therefore can be the basis for an actual utterance to be made by the proposer, or by the counterpart. In the former case the proposal can be regarded as the *capability* of the proposing party to supply the information contained in the utterance at that part of the process. In the latter case it is to be regarded as an information *requirement* of the proposing party.

10.5 Process profile matching

The Knowledge Base representation of B2B processes allows process matching or negotiation to be performed as a relative simple filtering process. Process steps are represented as specifications of information exchange under certain (pre)conditions. Preconditions are represented as information that is to be present in the Knowledge base. If the information, needed to fulfil a precondition, cannot be present, because no previous process step is defined to exchange it, the subsequent step cannot be performed.

Filtering is a more simple mechanism than process matching as proposed by Krukkert [2] or Wombacher [3], who both use combinatorial calculations to match two process proposals. Filtering however does not reveal the presence of possible dead locks or live locks in the resulting process, while combinatorial mechanisms usually do.

Middleware may support profile filtering by representing the profiles as proposals, acceptances and rejections in a negotiation pattern. The middleware of the initiating party sends the party's profile as a set of proposals to the other party. The middleware of the receiving party then compares the utterances received to the own profile. If the utterances match, acceptances are returned, if not, rejections. The result is a viable business process specification or a set of conditions that can never be met.

More advanced middleware can, in case the resulting process is not viable, issue counter proposals. These can e.g. be derived by relieving the rigid conditions set by the information system and allowing manual pre- or post-processing. The counter proposals may also be the result of relieving trade conditions, based on business rules. In general, the negotiation space as pictured in figure 10.1 may be used to attempt to arrive to a viable B2B process.

10.6 Goals and requirements

A B2B process can only be viable if the business goals of the partners are compatible. Business goals may be expressed in REA terms as the successful exchange of economic

resources. In order a resource exchange to be successful, information must be exchanged. The resource must be specified or identified, the transport services must be defined, payment must be detailed, etc. These are all information requirements, as are the commitments the partners need to express to each other.

A B2B process can therefore be composed from end to beginning. At the end the goals are formulated as end states. In order the end states to be reached, process steps are needed to provide the parties with the information they need. E.g. the Seller can only send goods after he knows the address of the Buyer. So the Buyers' address is a requirement of information to be exchanged in advance of transportation.

The commitment of the Buyer that he will pay for the goods after delivery is another information requirement of the Seller.

As partners gain experience in co-operating and trust levels increase, they may decide to change the process flow. One of the partners may take the initiative to re-open process flow negotiation. The other partner may reject or accept. This way logistics may be improved, payment conditions changed, terms of delivery adapted.

References

[1]. Gordijn and Akkermans: E3-value: Design and Evaluation of e-Business Models. In IEEE Intelligent Systems, Vol. 16(4):11-17, 2001

[2]. Krukkert, D.: Matching of ebXML Business Processes; Report Project number IST 2001-28548 OpenXchange

[3]. Wombacher: Matchmaking for Business Processes based on Choreographies. 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04)

11 Realization

Summary

In this chapter a number of implementation aspects are discussed. The architecture that is proposed in this thesis can be implemented natively in business information systems, in middleware that is connected to legacy systems or in stand-alone workstations for small businesses.

Different semantics of existing systems must be reconciled. Some concept, defined in one system, may be wider than a concept, defined in another system, they may overlap or they may be disjoint. Apart from ontological differences, differences in representation and syntax must be reconciled as well. It is illustrated in this chapter how middlware may handle such discrepancies and how it may communicate with the business information systems.

In section 11.9 the example that is represented in chapter 1 is presented as an implementation of the B2B architecture.

11.1 Introduction

In previous chapters a mechanism is described by which businesses may in a structured way shape the information exchange supporting cross-organizational business processes. The mechanism is described on a fairly abstract level, though in chapter 7 it was mapped to modelling languages such as UML and ORM. In this chapter a number of aspects are inspected regarding the implementation of the mechanism in legacy software, middleware and in newly built systems.

We identify the following stereotypical situations:

- Small business without IT support
- Business with a rigid application landscape
- Business with flexible, dynamic software

The first situation is described in chapter 13. In chapter 13 a browser based solution is designed. The solution is based on the XSLT and XForms standards. It allows small businesses to send and receive structured (XML-) business messages. It can be equipped with functions to negotiate the content of the messages and on the sequence the messages are being exchanged (the process flow). The environment described in chapter 13 has been implemented and is fully functional.

The majority of businesses have an application landscape according to the second situation. Applications (standard software packages or dedicated software systems) usually have a rigid data structure and are rigid in the types of transactions they accept or can produce. Changing the data structure of transactions or changing the conditions (sequence) under which transactions may be initiated requires reprogramming of the

system. This is usually not possible or feasible. Therefore in section 11.2 middleware is described that may form the interface between a set of rigid applications and the more dynamic business environment of the organization.

Internal applications are connected with the outside world by means of so called middleware. Middleware fulfils a number of functions:

- it guards security aspects, such as authorization and integrity and it takes care of technical protocol conversion
- it translates internal syntaxes (e.g. SQL) to external syntaxes (e.g. XML) v.v.
- and last but not least: it maps the external on the internal semantics v.v.

The first function is shortly addressed in section 11.3. The second function is described in section 11.4. The main focus however is on the third function. In sections 11.5 and 11.6 is described how semantics are bridged using the methods defined in earlier chapters. In section 11.7 is described how use can be made of a central repository.

The third situation is treated as a “green field” where newly built software forms the link between users, an SQL database system and the outside business world. The functions of such software are described in section 11.8.

In section 11.9 the architecture as described in this thesis is validated and illustrated with the Lehmann example that was cited in chapter 1. It is described how such a scenario can be supported by the architecture.

11.2 B2B Middleware

The task of B2B middleware is to translate the internal communication channels to the external ones. Three levels of translation will be described:

- technical protocol conversions (including security and authorization)
- syntactical and representational translations
- semantic mapping

The OpenXchange project has presented a model of the four stages of B2B relationships. At Modelling time the data and process models are developed that are valid within a business sector. At Profiling time individual companies decide and define what processes and data they support. At Agreement time profiles of potential trading partners are matched. The matching results in an agreement, or in the cancellation of the (electronic) trade relationship. If the matching is successful, then at Runtime the actual trading happens.

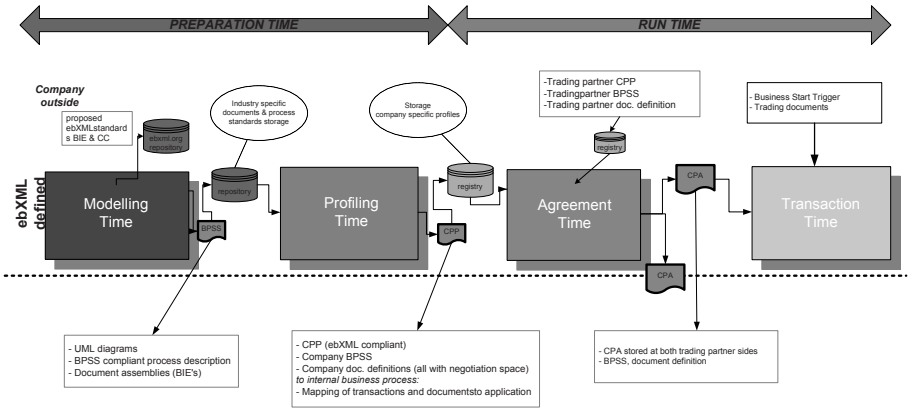


Figure 11.2 OpenXchange

This OpenXchange architecture is taken as a reference. In the architecture, described in this thesis, however, the strict separation between Agreement Time and Transaction Time is relieved. During a trade relationship it is possible to adapt semantics and work flows. In the OpenXchange architecture these are fixed at the start of the relationship. The OpenXchange architecture was dedicated to the ebXML framework. Although that framework served as an important inspiration of this thesis, our architecture has a wider scope. In ebXML profile matching is not defined at the semantical level.

In addition the OpenXchange project has developed a detailed reference architecture for the middleware that is to execute the functions at the various times. This architecture is described in section 2.1. That middleware architecture is taken here as a starting point.

The ECIMF project also has developed an architecture and a methodology for matching different data and process definitions of B2B communication. The presumption of the ECIMF architecture is that at modelling time the rules are developed how heterogeneous profiles (possibly profiles defined on different business sector models) can be matched. At Agreement time the matching rules are made specific for the trade relationship. The rules are used by a runtime engine to translate or map the messages or utterances of the partners.

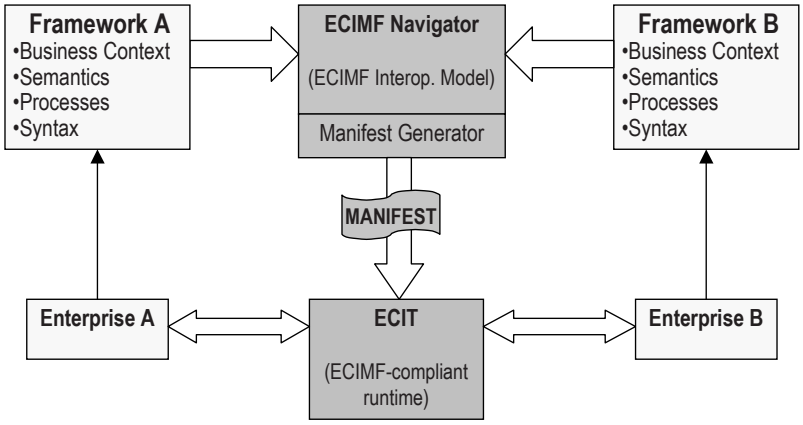


Figure 11.3 ECIMF

Matching rules in ECIMF are placed in four layers: Business context, Semantics, Processes and Syntax.

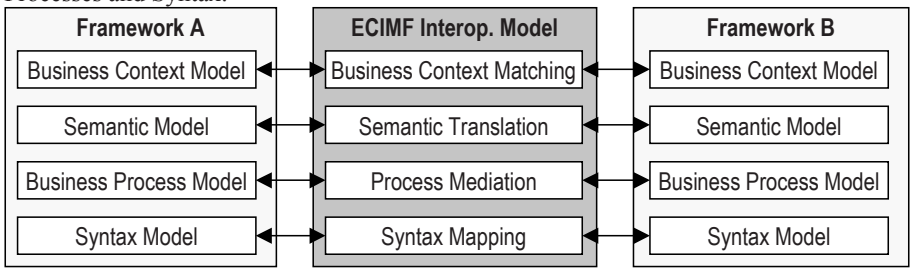


Figure 11.4 ECIMF layering

Layers are implemented as independently as possible.

- At the *Business Context* level the business goals of the partners are matched and the trade conditions are determined.
- At the *Semantic* level the ontologies of the partners are aligned and mapped on each other.
- At the *Business Process* level the process flow or the sequence in which messages or utterances are exchanged is being agreed on.
- At the *Syntax* level the representation of information is being mapped between the systems of the partners.

In this thesis all four levels are covered. The Business Context level is covered in chapter 10. Context and goal matching are expressed in semantic concepts and utterances that are defined in the next two levels. The Process flow is defined by means of preconditions of the semantic concepts. So these three levels are in practice collapsed. The matching is

described in section 11.4. In this thesis the phases are not completely distinct. Matchmaking at Agreement time is described as an ordinary runtime process, which may take place in conjunction with or as part of other business processes, such as contract negotiation.

A level that was out of scope for the ECIMF architecture is the technical protocol level. In the next section some remarks on protocol matching are made.

ECIMF presumes that mappings between internal and external structures are being made at design time. Runtime dynamic mapping was not foreseen in the project. In OpenXchange such a mapping is partly made at negotiation time. Both projects offered valuable insight in how B2B middleware should be layered. Syntax and semantics are separated in different layers. This offers the possibility to enhance the semantic layer with the function mentioned in section 11.1, without the burden of technical or syntactical issues.

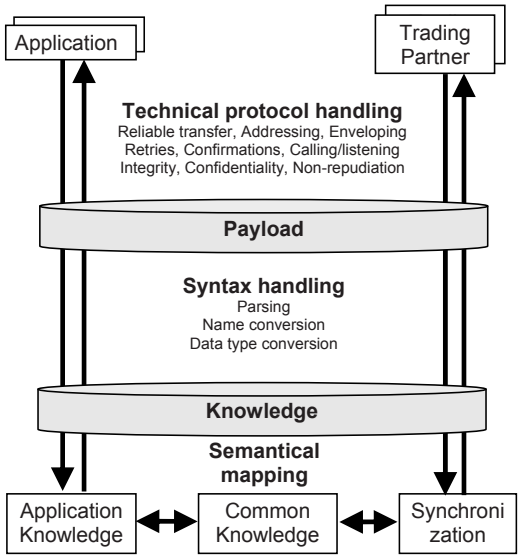


Figure 11.5 Middleware functions

In figure 11.5 the functions of the middleware are presented in a layered way. In the top part of the figure the technical protocol handling functions form the interface between the network that connects the middleware to applications and trading partners, and the 'payload': the information to be exchanged, rendered in a suitable syntax. Technical protocol handling includes addressing and routing, security, safeguarding reliability, etc. Technical protocol handling also includes listening (whether data is to be received), polling and calling external services to deliver information packages. Technical protocol handling is addressed in section 11.3.

Syntax handling converts internal data structures to the structures applications and trading partners may handle. It also includes parsing the data received from applications and trading partners and interpreting the information, converting it into some internal format. Data type conversion is also done at this level, and that function has semantic aspects as well: mainly the precision that may be lost (by e.g. converting weeks into months, or by abbreviating text). Syntax handling is further described in section 11.4.

Semantic mapping is the main topic of this thesis. The syntax handler stores the information in a format similar to the knowledge base structure as defined in chapter 6. Both information that is received from trading partners and from applications is stored in the knowledge base that is common to the partners in a trading relationship. The information is then forwarded to synchronize the states in all information systems. Semantic mapping is elaborated in sections 11.5 and 11.6.

An important aspect of middleware that supports dynamic trade relationships is that information structures and workflow definitions may evolve at run time. The middleware is not only handling instances and information value updates, but also information and process definitions. In the knowledge base such meta-information is structured isomorph to instance data. The utterances are however differently stereotyped. Differently stereotyped utterances may (and usually will) have different syntactical representations. In XML for instance, meta-level utterances (Definitions, Expansions, Restrictions) are represented as XML schema, while instant level utterances (Instantiations, Observations) are represented in ordinary XML.

11.3 Technical protocol handling

B2B conversations, as conceptually described in chapter 5, can be implemented in multiple ways. Some basic mechanisms are:

- File or database sharing. Both businesses share a database that contains the knowledge base information. No synchronisation is required, as information is stored in one place.
- File exchange and messaging. This is the electronic variant of paper document exchange. Utterances are assembled into messages or files that update the local information stores of the trading partners.
- Service orientation. One partner is passive (server), the other is active (client). The client takes the initiative to update the knowledge base or to retrieve updates that the passive partner has effectuated.
- Active synchronization. Both partners run a database. Each (relevant) database update at one partner is replicated to the other partner. This may be implemented by means of message exchange, but it may also be executed by a low level protocol.
- Passive synchronization. Partners only start a synchronisation process, fetching the updates from the other partners' database, when they need to take a decision

or to perform some activity. This may be implemented by means of a (web-) service.

In all mechanisms some protocol should be used whereby information, provided by one trading partner, may be accepted or rejected by the other one. Rejection may have many reasons, some are technical (e.g. wrong authorisation credentials), some syntactical (e.g. not conform the schema) and some semantic (e.g. the utterance is not allowed in the conversation). Acceptance at this level means that the utterance is accepted, within the scope of the speech act connotation. This means that the acceptance of an order means that it has been accepted that the partner has placed an order, not that the order is accepted to be delivered. To accept an utterance is in database terms called “To commit”. In conversation theory is spoken of “Grounding”: agreeing on the state of the conversation.

After an utterance is accepted, it cannot be “rolled back”, even if the information was not yet processed by the partners’ application system. The only way to reverse the business effect of an accepted utterance is by means of another utterance. E.g. an accepted invoice cannot be undone. A Credit note needs to be issued if the invoice appears to be invalid.

Often one or more intermediates are deployed for the interconnection. An intermediate may offer low level services, such as web hosting or e-mail store-and-forwarding, but it may also perform protocol and even syntax conversion. Some intermediates go even further and provide services at the business logic level. For the B2B relation it should not matter whether use is made of intermediates or not. Condition is that the services provided to each party are well identified. If so, processing by the intermediate can be treated equal to internal processing.

One of the requirements stated in chapter 4 is that a business conversation may use several technical channels in parallel. This can be achieved by decoupling the technical services of the middleware from the syntactical and semantic services, see figure 11.5. The semantics are dealt with irrespective of the channel. The channel may even be a printer and a data entry station.

The middleware may also contain mechanisms to negotiate a channel for certain communication with trading partners. Elaboration of such mechanisms is outside the scope of this thesis. One mechanism is profile matching as is defined in the CPP/CPA protocol for matching the parameters of ebMS [6].

11.4 Syntactical handling

There exist various ways to serialize data. During serialization data drawn from a database or from computer memory is composed into a (long) string of bits. The string is decomposed at the other end, the individual data elements and the structure they have in between them are recognised and the data is stored in another database or in memory for further processing.

Popular serialization methods are:

- Fixed position methods. Specific data fields are identified by their absolute position in the string. In the past punch cards used this method, but in fact the raw output of an SQL processor does the same
- Tagging. Each data element is tagged with a name or identifier. XML is a well-known example of a tagged syntax. SQL input data is also tagged.
- Delimiting. Special characters are used to separate the data elements. In a separate schema is defined in which sequence the elements appear. Comma Separated files are an example.
- Length indication. For each element the length is indicated in the file itself. The sequence of the elements is defined separately, or the elements are tagged. ASN.1 is an example.

Often a combination of the methods is used. E.g. in UN/EDIFACT the segments are tagged and the elements and sub elements within the segments are delimited.

For each method specific parser software must be used. The parser reads the serialized stream and schema information, if applicable (schemas are separate files with the definition of the positions or tags of the elements). The parser may store the information in a database, re-serialize it in an internal format or read it in memory.

In this thesis we are not primarily interested in the syntactical structures, nor in parser technology. We assume that data is recognized and stored somewhere. We also assume, and this is essential, that the semantic definitions of the information elements are casted in the same ontology language and are mappable on the B2B knowledge base structure as presented in this thesis.

After the elements were recognized, based on schema information or on annotation, the elements must be mapped on the elements as they are defined in the knowledge base.

Several types of mismatched may occur after syntactical recognition of concepts and entities. Visser [1] has categorized these mismatches. He distinguishes structural mismatches and representational mismatches.

(1) naming conflicts
    (different names are used to represent the same concept),
(2) domain conflicts
    (the same concept is represented by different values),
(3) meta-data conflicts
    (the same concepts is represented at the schema level by one partner and at the instance level by the other), and
(4) structural conflicts
    (different data organization is used to represent the same concept).

These conflicts partly result in syntactical mismatches and partly in ontological mismatches. We define a mismatch as syntactical if the concept definition matches, but is

represented in different ways or in different structures. It is not always clear whether a conflict is syntactical or ontological. The concept, for instance, may be represented in different ways with different precision. E.g. a colour may be represented by an RGB value or by text. The textual representation may be vague and subjective, as a contrast to the precise encoding in RGB. A concept may also be represented explicitly by one partner and implicitly as a set of attributes to another concept by the other.

Conflict resolution is to be performed in a layered way. First conflicts at the data type level are to be solved. These may be mismatches in lexical representation (ASCII or bit string, code or text); mismatches in value space used, in different (measure) units, etc. Then structural mismatches are tackled, such as meta-level conflicts (e.g., car versus vehicle with type 'car') and model structure conflicts (e.g., car-colour versus car-body-colour). Then it can be determined if the resulting concepts are ontologically equal. This determination is described in section 11.5.

The table representation as presented in chapter 6 makes it possible to determine structural conflicts.

Note that the aim here is not to give hints to developers how to solve the mismatches at design time. Dynamic B2B middleware should be capable of solving the mismatches at runtime. That means the middleware must be able to determine the type of mismatch and have mechanisms to resolve it. Many mismatches at data type level, such as unit conversion, can be resolved automatically. Often however, the stored representation is richer than the representation in the communication, or vice versa. In that case it may be needed to consult a human employee, to decide if the loss of information as a result of the conversion is acceptable or not.

After consulting a human the middleware should show self learning behaviour: a next time a similar case is met, the decision can be made automatically, or the human may be hinted with his earlier decisions.

As an illustration some frequently occurring mismatches will be inspected.

Unit mismatches
Examples: inches versus centimetres, weeks versus months, RGB versus RAL code, Latitude-Longitude versus Locode.
Most units may be converted by means of a formula or table. In many cases however precision may be a problem: the target unit system may be coarser than the source system. Human assistance is needed to decide whether a loss of precision is acceptable given the business context.

Text/code
Example: Terms of Delivery represented in free text versus a coded representation as issued by the International Chamber of Commerce. In general it is easier to convert codes to text (using the code table) than the other way around. Sometimes a text recognition system may decipher the text and map it to the (nearest) code. Human intervention is

often needed to decide if the risk that mismatches occur and possible information loss is acceptable.

Combined attributes
A notorious example of this case is the postal address. Addresses may be coded, structured or be represented as free text, or be represented as any combination of those. Here, too, it is easier to convert the more structured or coded variants into the less structured ones than vice versa. For standard concepts such as addresses it is conceivable that standard web services may assist in the conversion from less structured to structured representations. Many other, less standard, examples exist however. One other example is the product description.

Date precision
Date and time information may differ in precision. Translation from e.g. Month to Week or vice versa depends on the business context. In some cases the mapping should be done to the beginning of the target period, sometimes to the end of the period and sometimes to the median.

Character set
In textual elements sometimes diacritical characters can be present. Depending on the business context the loss of diacriticals is acceptable or not. A more tricky conversion is the conversion from the Cyrillic, Greece or Chinese alphabet to the Latin one, vice versa. Another character set related case is the conversion of the base number in numeric representations (e.g. from decimal to hexadecimal).

Facets
The source and the target may have different facets (field lengths, min/max value, patterns) defined on the data. In some cases this can be resolved without loss of information, e.g. by converting one long line of text into 5 shorter ones. Sometimes text must be abbreviated or truncated. Values may be rounded, maximized or minimized. Again, the decision whether this is acceptable must be taken in the business context.

Language
Language translations (English to Spanish, etc.) may be needed. The risk of wrong translation or wrong interpretation must be assessed with business knowledge.

Flattening/normalization
Some systems may have defined intermediate concepts whereas other have a flat data structure. An example is the colour of a car, in one system this may have been defined as the colour of the body of the car (body being a separate concept) and in another system as the colour of the car itself. It should be assessed by humans whether in both cases the same concept is referred to and how to resolve possible cardinality conflicts.

Different coordination systems
Locations may be referred to in many different ways. Some use a system with geographical coordinates, other an indication of country-state-city-street. The main issue is (again) precision.

Master data/operational data
Sometimes information that was already known and has not changed is communicated again. This can have good reasons. One reason may be an official/legal one. The sender may be forced by law to include the information. Other reasons are human readability (in case the system cannot be used) and security. It should be decided if and how redundant data is checked and how to handle in case of deviations.

Number/formula
A last example is the representation of a number or amount as a formula or as a number. Here, too, the precision issue is the main problem to solve.

11.5 Semantic mapping

The syntax interpretation of the middleware leads to a representation of the information as a B2B knowledge base, as described in chapter 6. Information from several sources (trading partners, applications) result in several knowledge bases that are based on a common basic ontology. That means that in the full knowledge base tables the top part is identical. At the bottom deviation may exist.

Middleware is to:
- check the validity of the received information before adding it to a knowledge base
- accept the information into the knowledge base or respond with a rejection
- include the information into the knowledge base of one or more information systems
- query the information systems on states that trigger outgoing utterances
- assemble outgoing utterances
- send utterances by submitting them to a syntax handler.

Note that in the scope of this thesis middleware function can only be sketched roughly. It is not our ambition to present a full functional specification of such middleware here.

Validity checking

Validity of received information is checked against existing information in the knowledge base. The received utterances should each be based on previously entered utterances. The structure of the utterances in relation to the utterances they are based on, needs to be validated.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
10	0	0:00	S	Prop	Def	Assert	Concept	Role	Do	Role	Concept		0	n	B,S	Def, Inst, Obs	Any		
10	0	0:00	B	Acc	Def	Assert	Concept	Role	Do	Role	Concept		0	n	B,S	Def, Inst, Obs	Any		
11	0	0:00	S	Prop	Def	Assert	Event		Do	Role	Concept		1	1	B,S	Inst, Obs	Any		
12																			

Table 11.1 Knowledge base

1. *Utterance number*. The utterance number must be unique and consecutive. Duplicate numbers are an indication of synchronization problems. Exception is the Propose-Accept pattern as defined by the Action code (column 5). Accepted (and Rejected) utterances have the same number as the Proposed one and are identical to it with exception of the Action code.
2. *Based on Utterance number*. Each utterance must be based on a previous utterance and obey the rules stated in that previous utterance. Rules include Cardinality (columns 14 and 15), Party (columns 4 and 16), Stereotype (columns 6 and 17), Intention (columns 7 and 18), Preconditions and Transaction. Party (column 4), Stereotype (column 6) and Intention (column 7) of the new utterance must be part of the allowed Parties (column 16), Stereotypes (column 17) and Intentions (column 18) as included in the utterance it is based on. Preconditions must be checked (see later) and the utterance must be accompanied by utterances that are based on the utterances listed in the Transaction column (column 20) of the utterance it is based on.
3. *Timestamp*. Each utterance has a time stamp. The time stamp indicates the time the utterance was made available to the trading partner; it is not the time stamp of the communication channel. Utterances in the same transaction have the same time stamp.
4. The party that entered the utterance is indicated. For the middleware applications also play the role of a party. So for internal knowledge bases the application ID is entered in column 4.
5. Definitions and expansions are not entered directly, but only after approval of the trading partner. They may only be the base of later utterances if and when the receiving partner has accepted the utterance. This is done by repeating the utterance with an Accept Action instead of the original Propose Action. Other types (e.g. Observations) may be added right away, with an Action of Add.
6. Each utterance is of one of the seven stereotypes (**Def**inition, **Exp**ansion, **Res**triction, **Sta**te, **Ins**tantiation, **Per**ception, **Obs**ervation). The stereotype must be part of the set specified in column 17 of the utterance it is based on.

7. The intention of the sending partner is indicated in column 7. The list or tree of intentions is fixed. It however may be extended and standardized by some standardization body. Definitions of new intentions are not bilaterally negotiated. The intention of an utterance must be part of the set specified in column 18 of the utterance it is based on.
8. Name of the concept. The name uniquely indicates the concept or entity, within the namespace formed by the name of the entity it is based on.
9. Name of the source role. When omitted it is the same as the source role of the utterance it is based on. Otherwise it is defined within the namespace formed by the source role name of the utterance it is based on.
10. Name of the Verb. When omitted it is the same as the Verb of the utterance it is based on. Otherwise it is defined within the namespace formed by the Verb name of the utterance it is based on.
11. Name of the Target role. When omitted it is the same as the Target role of the utterance it is based on. Otherwise it is defined within the namespace formed by the Target role name of the utterance it is based on.
12. Name of the Target concept. Defined within the namespace formed by the target concept of the utterance it is based on.
13. ID scheme number. If the attribute is part of an ID scheme, the number of the scheme is entered here.
14. Minimum repetition. The minimum number of occurrences of this property in any instance of this concept in the knowledge base. Integer: 0 or higher. If equal to 0, the property is optional or conditional, if 1 or more it is mandatory. Instantiations, based on this utterance must have a minimum of occurrences defined in the same transaction as indicated in this column.
15. Maximum repetition. The maximum number of occurrences. Integer: 1 or higher, or ‘n’. If indefinite an ‘n’ is placed in this column. Instantiations, based on this utterance may not have more occurrences than as indicated in this column.
16. With allowed stereotype. Set of stereotypes. Utterances based on this one must have a stereotype listed in this set.
17. With allowed intention. Set of intentions. Utterances based on this one must have an intention listed in this set.
18. Precondition. A Boolean expression of utterance numbers of states. Utterances, based on this one are only valid if the precondition evaluates to true.
19. Transaction with. Utterance number. An utterance, based on this one is only valid if in the same transaction an utterance exists that is based on the utterance indicated in this column.

Acceptance

Instantiations and observations that obey the integrity rules are accepted. Definitions, and other utterances with an action code of Propose, are evaluated against the policy (see chapter 10). If acceptable, an accept utterance is sent, otherwise a rejection.

Inclusion

Definitions result into transactions to be accepted from the trading partner. Those transactions are usually forwarded to one or more information systems. Each information system is therefore represented in the middleware as a profile that lists the allowed transactions with their preconditions. The profile may be formatted as a knowledge base table. The transactions are mapped to update queries of the information system. The update queries have placeholders for the variable data. In section 8.4 is described how transactions are represented as SQL queries.

Many business information systems do not accept raw SQL as update transaction language, but have a proprietary format. In these cases a mapping of the knowledge base table format to that proprietary format is made available to the syntax handler. The semantics of the allowed transactions are formatted as a knowledge base table.

The information in received transactions should be placed in the update query placeholders. This needs not be a straightforward process: information in received utterances may be differently structured than information as it is to be stored in the information systems. But because both structures are represented in knowledge base format, the mapping may be calculated.

When a one-to-one mapping can be made between the partner KB and the application KB the relation can be defined in the following table:

Line #	Knowledge Base	Utterance #	Stereotype	Knowledge Base	Utterance #
1	Seller	10	Instantiation	Application	20
2	Seller	11	Observation	Application	21
3	Seller	12	Observation	Application	22
4	Seller	13	Observation	Application	23
5	Seller	14	Observation	Application	24

Table 11.2 Knowledge base mapping

The table maps utterances from one knowledge base to utterances in another. The utterance numbers in the table are the numbers of the utterances the mapped utterances are based on. So table line 1 reads: “Each utterance entered in the Sellers’ knowledge base that is based on utterance number 10 is mapped to an utterance in the applications’ knowledge base that is based on utterance number 20 with a stereotype of “Instantiation”. ‘Mapped’ means that most utterance attributes are simply copied. An exception is the utterance number, which is generated as the numbering in the application knowledge base must stay consistent (numbers to be unique and consecutive).

The table entries in table 11.2 are an example, as illustrated in the following application knowledge base:

Realization

Application

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
20	1		B	Add	Def	Assert	Auto		have	License	Text	1	1	1	S	Inst	Assert		
21	2		B	Add	Def	Assert	Auto		have	Colour	{red, white, blue}		0	1	S	Obs	Assert		20
22	3		B	Add	Def	Assert	Auto		have	Owner	Person		0	1	S	Obs	Assert		20
23	4		B	Add	Def	Assert	Person		have	Name	Text	1	1	1	S	Obs	Assert		20
24	5		B	Add	Def	Assert	Person		have	Birth	Date		0	1	S	Obs	Assert		20

Table 11.3 Application knowledge base

The partner knowledge base has the following entries:

Seller

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
10	1		B	Add	Def	Assert	Car		have	License	Text	1	1	1	S	Inst	Assert		
11	2		B	Add	Def	Assert	Car		have	Colour	{red, white, blue}		0	1	S	Obs	Assert		20
12	3		B	Add	Def	Assert	Car		have	Owner	Citizen		1	1	S	Obs	Assert		20
13	4		B	Add	Def	Assert	Citizen		have	Name	Text	1	1	1	S	Obs	Assert		20
14	5		B	Add	Def	Assert	Citizen		have	Birth	Date		1	1	S	Obs	Assert		20

Table 11.4 Seller's knowledge base

Note that in the example the concepts that are discussed with the Seller are named ‘Car’ and ‘Citizen”, while in the application concepts exist that are named ‘Auto’ and ‘Person’. If these are simply other names for the same concepts, the renaming may be performed at syntax level. In this example however it is assumed that the concepts differ somehow, and a semantic mapping is to be done. A business person may be interrogated by the system to make such decisions. The differences are not made explicit in the example.

A transaction may be received from the trading partner with the following information:

Seller

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
15	10	12:30	S	Add	Inst	Assert	123		has	License	JZ8845								
16	11	12:30	S	Add	Obs	Assert	123		has	Colour	Blue								
17	12	12:30	S	Add	Obs	Assert	123		has	Owner	456								
18	13	12:30	S	Add	Obs	Assert	456		has	Name	Fred								
19	14	12:30	S	Add	Obs	Assert	456		has	Birth	19501228								

Table 11.5 Seller's knowledge base

According to table 11.2 these utterances are added as utterances to the application table as:

Application

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
55	20	12:30	S	Add	Inst	Assert	123_ Auto		has	License	JZ8845								
56	21	12:30	S	Add	Obs	Assert	123_ Auto		has	Colour	Blue								
57	22	12:30	S	Add	Obs	Assert	123_ Auto		has	Owner	456								
58	23	12:30	S	Add	Obs	Assert	456_ Person		has	Name	Fred								
59	24	12:30	S	Add	Obs	Assert	456_ Person		has	Birth	19501228								

Table 11.6 Application knowledge base

In the application a new ‘Auto’ is instantiated with the License JZ8845. In the knowledge bases the ‘Auto’ is identified by ‘123’. This is an internal identification. The real identification is the ‘License’, as indicated in column 13 of tables 11.3 and 11.4. The ‘Owner’ is ‘Observed’ rather than ‘Instantiated’ as indicated in column 17 of tables 11.3 and 11.4. That means an instance of the ‘Owner’, with identifying ‘Name’ of ‘Fred’ must already exist in the knowledge base. The ‘Observation’ of the identifying ‘Name’ is in fact a reference to that existing entity. The ‘Observation’ of the non-identifying ‘Birth. Date’ means that that date is to be updated.

These utterances may be translated into SQL according to the method in section 7.4:

```
INSERT INTO Auto (License) VALUES (JZ8845)
UPDATE Auto SET Colour=’Blue’ WHERE License=’JZ8845’
UPDATE Auto SET Person=’Fred’ WHERE License=’JZ8845’
UPDATE Person SET Birth=’19501228’ WHERE Name=’Fred’
```

Utterances that cannot be mapped to an application are either ignored or directed towards an employee. In the latter case a knowledge base is populated for communication with that employee and the utterances are not mapped to application transactions (such as SQL), but to a user interface. The definition of the user interface may be represented as an XML schema and the utterances may be sent as XML messages. In chapter 13 a method is described how to present an XML message as a screen form in the employees' browser.

The mapping table allows utterances to be mapped or distributed to multiple applications. It even allows to forward utterances to other trading partners, provided a one-to-one mapping can be made to the utterances in the other partners' knowledge base. When no one-to-one mapping can be made, a semantic mismatch exists. In section 11.6 semantic mismatches are analysed and the way to deal with them is described.

One of the objectives of the architecture presented in this thesis is that no (or minimal) manual effort is needed when establishing a connection with a trading partner, while recognizing semantic heterogeneity. The mapping as presented in table 11.2 therefore should be automatically created during Matching time.

During Modelling time the structure of the application knowledge base(s) are created. The KB structure can be derived from the model of the application, expressed in some modelling language, as described in chapter 7. The application model may have a different structure than the ontology that forms the basis for semantic negotiation with trading partners. At Profiling time the application KB structure is mapped to that ontology. In most cases this is a manual process that however needs to be performed only once when the application is installed (and after is has been changed). The ontology, in the format of a knowledge base, serves as the profile for defining the inter-organizational processes.

The mapping table as illustrated in table 11.2 should be automatically created during Matching time. At Matching time the trading partners propose definitions of concepts to be added to the common knowledge base. The other partner will only accept a proposed definition if he can generate a mapping to his application(s) or if he decides the concept information may be ignored. If he can find a perfect match between a proposed concept and a concept that has been defined in his application knowledge base he accepts the proposal and creates the mapping entry in the mapping table. A perfect match means that the rigid and the mandatory properties of the concept are identical and that the other properties are mappable.

In the example shown, the proposed concept 'Car' has the same properties as the application concept 'Auto'. One of its properties, 'Owner', however has a different target concept proposed: 'Citizen' instead of 'Person'. The middleware then inspects the properties of 'Citizen' and 'Person' and concludes that these are identical (both property terms and data types), so 'Citizen' may be mapped on 'Person'.

Just like the mapping of application transactions to the knowledge base structure is being made at Modelling time, the mapping of data exchange languages is. Each data exchange language (be it UN/EDIFACT, ASN.1, comma separated or XML based) should provide a mapping to the knowledge base structure, that is, to specify what concepts are being manipulated with which intention. Moore [2] has performed such an exercise (mapping to FLBC) with some ANSI X12 messages. FLBC has its own XML based representation language. Here it is assumed that (legacy) languages such as UN/EDIFACT and XML dialects will remain to be used. When the knowledge base structure is standardized it can function as the canonical semantic description of each structured business language.

During Matching or negotiation the specific knowledge base structure agreed with some partner may deviate from the base ontology. Some deviations have no or minor consequences to the mapping (e.g. when the resulting knowledge base contains a subset of the set of transactions as defined in the application). Other may have influence and result in more sophisticated mapping than the one-to-one mapping as described in this section. In section 11.6 a number of deviations are assessed, with their mapping solutions.

Querying

The process to query the information systems on information that is to trigger outgoing utterances is not directly dependent on incoming information. Activities within the organization that are entered in the information system by means of a user interface may also trigger the need to inform the trading partner about the next step in the process. Depending on the context, the information system is polled by the middleware less or more frequently, or dependent on specific events.

11.6 Ontological mismatches

In a B2B knowledge base concepts are defined by the properties they have. Two concepts with the same set of properties are assumed to be equal. That is, they have the same intension (set of properties) and the same (potential) extension (set of instances). A property is the combination of the target concept of the property, for which the equality test is recursively applied as well, and the role the target concept plays. A role is the combination of an event (verb) and a semantic role name.

Concept definitions may include conditions. Different sets of conditions may lead to different concepts. For example the concept “Child of unmarried parents” has as condition that the parents are not married. Dropping the condition leads to a different concept (“Child”), which is a superset of the concept “Child of unmarried parents”. Therefore a “Child of unmarried parents” may be based on a “Child”, with a precondition that points to the married State of its parents (see table 11.7).

Realization

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	12:30	S	Add	Exp	Assert	Person		has	Parent	Person		0	2					
101	10	12:30	S	Add	Sta	Assert	Person		has	Spouse	Person		1	1					
102	100	12:30	S	Add	Def	Assert	Child		has	Parent	Person		0	2					
103	102	12:30	S	Add	Def	Assert	Child of unmarried parents		has	Parent	Person		2	2				¬101	

Table 11.7 Example

Except the mismatches on the level of the ontological definition, there may be a mismatch on the expanded content model of the concept. E.g. it may be clear that both sending and receiving systems store information on a "car" (the definitions match), but the sending system includes properties (e.g. colour) that the receiving system does not know or the receiving system needs information on the car that the sending system cannot provide. In the former case there is no problem, if in the sequel of the business process the information is not needed. In the latter case an information requirement is not fulfilled. The receiving party then must assess whether it can perform its obligations without the information or it should request the sending party to supply the information anyhow, e.g. by manually adding it to the data stream.

We distinguish the following basic semantic (mis-)matches:

Total match

When a total match exists and the definitions of the concept in both Knowledge Base and receiving system are identical, the information can be stored without problems. A match however does not need directly to exist between two utterances. An utterance in the partner knowledge base may map to a number of utterances in the application knowledge base and vice versa.

An utterance may be split in two ways to map on other utterances. First a concept in one knowledge base may be represented as more than one concept in another knowledge base. For example a car may have a colour in one knowledge base, while in another knowledge base a car may have a body which has a colour in another knowledge base. Second a set of concepts may be split based on the value of properties. E.g. one knowledge base may contain the concept of a Transport Means, while another may have two concepts: Road_ Transport Means and Rail_ Transport Means, based on the value of the Transport Mode property.

As utterances are as fine grained as to the level of individual properties, the first case does not pose any problems, as an example shows.

Seller

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	10:00	B	Add	Def	Assert	Car		Has	Colour	{red, white, blue}		1	1	B	Inst	Assert		
200	100	12:30	B	Add	Inst	Assert	123_ Car		Has	Colour	white								

Table 11.7 Total match

Line #	Knowledge Base	Utterance #	Stereotype	Knowledge Base	Utterance #
1	Seller	100	Instantiation	Application	100
2	Seller	100	Instantiation	Application	101

Table 11.8 Mapping

Application

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	10:00	B	Add	Def	Assert	Car		Has	Part	Roof		1	1	B	Inst	Assert		
101	11	10:00	B	Add	Def	Assert	Roof		Has	Colour	{red, white, blue}		1	1	B	Inst	Assert		
201	10	10:00	S	Add	Inst	Assert	123_ Car		Has	Part	123_ Roof								
202	20	12:35	B	Add	Inst	Assert	123_ Roof		Has	Colour	white								

Table 11.9 Total match

In the example the car colour is mapped on the roof colour. As a consequence the car should have a roof. So the roof is instantiated and allocated as part of the car in Utterance # 201 of the application knowledge base.

The colour of a car roof does semantically not to be the same as the colour of a car. During matching time the middleware is probably to consult an employee to verify the mapping. That also is the case when the potential mapping is ambiguous: when e.g. not only car roofs may have a colour, but also car doors.

In the second case the value spaces of the discriminating properties differ. In the example they are even disjoint: a Transport Means is either a Road Transport Means or a Rail Transport Means. Mapping can be made to both target concepts, with the rule that a mapping is ignored by the middleware in case value spaces don't match.

Realization

Seller

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	10:00	B	Add	Def	Assert	Transport Means		Uses	Transport Mode	{road, rail, water, air}		1	1	B	Inst	Assert		
101	11	10:00	B	Add	Def	Assert	Transport Means		May load	Weight	Weight_ Measure ment		0	1	B	Obs	Assert		100
200	100	12:30	B	Add	Inst	Assert	123_ Transport Means		Uses	Transport Mode	road								
201	100	12:35	B	Add	Obs	Assert	123_ Transport Means		May load	Weight	3000 kg								

Table 11.10 Common knowledge Base

Line #	Knowledge Base	Utterance #	Stereotype	Knowledge Base	Utterance #
1	Seller	100	Instantiation	Application	200
2	Seller	101	Observation	Application	201
3	Seller	100	Instantiation	Application	203
4	Seller	101	Observation	Application	204

Table 11.11 Sellers' mapping

Application

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	10:00	B	Add	Def	Assert	Transport Means		Uses	Transport Mode	{road, rail, water, air}	1	1	1	B	Def	Assert		
101	11	10:00	B	Add	Def	Assert	Road_ Transport Means		May load	Weight	Weight_ Measure ment		0	1	S	Obs	Assert		
200	100	11:00	B	Add	Def	Assert	Road_ Transport Means		Uses	Transport Mode	{road}	1	1	1	S	Inst	Assert		
201	101	11:00	B	Add	Def	Assert	Road_ Transport Means		May load	Weight	Weight_ Measure ment		0	1	S	Obs	Assert		
203	100	11:00	B	Add	Def	Assert	Rail_ Transport Means		Uses	Transport Mode	{rail}	1	1	1	S	Inst	Assert		
204	101	11:00	B	Add	Def	Assert	Rail_ Transport Means		May load	Weight	Weight_ Measure ment		0	1	S	Obs	Assert		

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
300	200	12:00	S	Add	Inst	Assert	123_ Road_ Transport Means												
301	200	12:00	S	Add	Obs	Assert	123_ Road_ Transport Means		May load	Weight	3000 kg								

Table 11.12 Application Knowledge Base

In the example the May Load Weight of a Transport Means is mapped on both a Road_ Transport Means and a Rail_ Transport Means, which are also to be instantiated. Because the Transport Mode of a Rail_ Transport Means is ‘rail’, and the Transport Mode of the received Transport Means is ‘road’, the mapping on the Rail_ Transport Means is ignored.

At matching time the middleware notices that the requested Transport Means cannot be instantiated directly, but its subtypes can. It then can automatically map the properties on the subtypes.

Combinations of the two cases are of course possible as well.

Subset
The concept defined in the receiving system may be narrower than the concept on which information was received. For example, a receiving system may have defined the concept truck or Road_ Transport Means, while information is received on “Transport Means”, including ships and planes.

In order to be able to process the information, the criteria that narrow the concept must be part of the instantiation or observation (the set of properties that are transmitted). If the discriminating properties (that determine whether a vehicle is a car and not a ship) are part of the observation, the data can be stored in the receiving system.

Realization

Seller

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	10:00	B	Add	Def	Assert	Transport Means		Uses	Transport Mode	{road, rail, water, air}		1	1	B	Inst	Assert		
200	100	12:30	B	Add	Inst	Assert	123_ Transport Means		Uses	Transport Mode	road								
201	100	12:35	B	Add	Inst	Assert	124_ Transport Means		Uses	Transport Mode	water								

Table 11.10 Common knowledge Base

Line #	Knowledge Base	Utterance #	Stereotype	Knowledge Base	Utterance #
1	Seller	100	Instantiation	Application	20
2	Seller	100	Instantiation	eb-Forms	20

Table 11.11 Sellers' mapping

Application

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
20	10	10:00	S	Add	Def	Assert	Road_ Transport Means		Uses	Transport Mode	{road}		1	1	B	Inst	Assert		
80	20	12:35	B	Add	Inst	Assert	123_ Road_ Transport Means		Uses	Transport Mode	{road}								

Table 11.12 Application Knowledge Base

eb-Forms

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
20	10	10:00	S	Add	Def	Assert	Non-road_ Transport Means		Uses	Transport Mode	{rail, water, air}		1	1	B	Inst	Assert		
90	20	12:40	B	Add	Inst	Assert	124_ Non-Road_ Transport Means		Uses	Transport Mode	water								

Table 11.13 User Interface Knowledge Base

In the example the Buyer has added a definition of a Transport Means to the common knowledge base (Table 11.5, Utt# 100). The Seller has an application in which a Road_ Transport Means has been defined (Table 11.7, Utt# 20). He has mapped instances of Transport Means, received from the buyer to that utterance (Table 11.6, line# 1), so in his application a Road_ Transport Means is instantiated. Such instantiation will fail if the Transport Means uses a different Transport Mode than Road. Therefore he has made a second mapping (Table 11.6, line# 2) to the knowledge base of a user interface, for non-road means of transport. On receiving an instantiation of a Non-road Transport Means the instantiation is redirected to an employee (Table 11.8, Utt# 90) who can take appropriate action.

Another subset relation exists when the concept spans the same (potential) extension, but contains less information (properties).

Seller

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	10:00	B	Add	Def	Assert	Transport Means		Has		ID	1	1	1	B	Inst	Assert		
101	11	10:00	B	Add	Exp	Assert	Transport Means		Has	Capacity	Volume_ Measure ment		0	1	B	Obs	Assert		
200	100	12:30	B	Add	Inst	Assert	123_ Transport Means		Has		561234								
201	100	12:35	B	Add	Obs	Assert	124_ Transport Means		Has	Capacity	30m3								

Table 11.14 Common knowledge Base

Realization

Line #	Knowledge Base	Utterance #	Stereotype	Knowledge Base	Utterance #
1	Seller	100	Instantiation	Application	20

Table 11.15 Sellers' mapping

Application

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
20	10	10:00	S	Add	Def	Assert	Transport Means		Has		ID		1	1	B	Inst	Assert		
80	20	12:35	B	Add	Inst	Assert	123_ Transport Means		Has		561234								

Table 11.16 Application Knowledge Base

The example shows that in the common knowledge base a Transport Means has been defined that has an ID and a capacity. In the Sellers application the Capacity is not included and is irrelevant for the Sellers operation. Therefore only the ID has been mapped, the received Capacity is ignored.

Superset

The concept in the receiving system may also be wider than the concept that is received. The received data can then be stored without problems, complemented with the properties that define the narrower concept. These properties are usually not transmitted, as they are fixed and can be derived from the definition.

Seller

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	10:00	B	Add	Def	Assert	Transport Means		Has		ID	1	1	1	B	Inst	Assert		
101	11	10:00	B	Add	Exp	Assert	Transport Means		Uses	Transport Mode	{road, rail, water, air}		1	1	B	Inst	Assert		
102	100	10:01	B	Add	Def	Assert	Road_ Transport Means		Has		ID	1	1	1	B	Inst	Assert		
103	101	10:01	B	Add	Def	Assert	Road_ Transport Means		Use	Transport Mode	road		0	1	B	Inst	Assert		
200	100	12:30	B	Add	Inst	Assert	123_ Road_ Transport Means		Has		451234								

Table 11.17 Common knowledge Base

Line #	Knowledge Base	Utterance #	Stereotype	Knowledge Base	Utterance #
1	Seller	102	Instantiation	Application	10
2	Seller	103	Instantiation	Application	11

Table 11.18 Sellers' mapping

Application

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
20	10	10:00	S	Add	Def	Assert	Transport Means		Has		ID	1	1	1	B	Inst	Assert		
21	11	10:00	S	Add	Exp	Assert	Transport Means		Uses	Transport Mode	{road, rail, water, air}	1	1	1					
80	10	12:35	B	Add	Inst	Assert	123_ Transport Means		Has		451234								
81	11	12:35	B	Add	Inst	Assert	123_ Transport Means		Uses	Transport Mode	{road}								

Table 11.19 Application Knowledge Base

In the example in the common knowledge base a concept Road_ Transport Means has been defined, based on Transport Means. The application knows Transport Means as a concept with a code for the Transport Mode used. When a Road_ Transport Means is initiated in the common knowledge base, the ID is mapped to the ID of the applications' Transport Means. Implicitly, however, also the Transport Mode has been defined, as all Road_ Transport Means have as the Transport Mode they use 'Road'. Mapping rule 2 states that (also implicit) instantiation utterances that are based on common Knowledge Base Utt# 103 are mapped as instances of Utt# 11 in the application Knowledge Base.

Overlap

There may be an overlap in the concept received and the concept to be stored. For example the receiving system may have defined a "freight means of transport" (including trucks and barges), while the received concept is a road vehicle, including buses. In this case, like the superset case, the extra discriminating properties need to be transmitted (in this case the fact that the transport means carries goods and no persons). On receiving the property set is extended with the defining properties of the received concept (in this case that it is a road vehicle).

Realization

Seller

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	10:00	B	Add	Def	Assert	Transport Means		Has		ID	1	1	1	B	Inst	Assert		
101	11	10:00	B	Add	Exp	Assert	Transport Means		Uses	Transport Mode	{road, rail, water, air}		0	1	B	Inst	Assert		
102	12	10:00	B	Add	Exp	Assert	Transport Means		Carries	Load type	{persons, cargo}		0	1	B	Inst	Assert		
103	100	10:01	B	Add	Def	Assert	Road_ Transport Means		Has		ID	1	1	1	B	Inst	Assert		
104	101	10:01	B	Add	Def	Assert	Road_ Transport Means		Use	Transport Mode	road		0	1	B	Inst	Assert		
105	102	10:01	B	Add	Def	Assert	Road_ Transport Means		Carries	Load Type	{persons, cargo}		0	1	B	Inst	Assert		
200	103	12:30	B	Add	Inst	Assert	123_ Road_ Transport Means		Has		451234								
201	105	12:30	B	Add	Inst	Assert	123_ Road_ Transport Means		Carries		cargo								

Table 11.20 Common knowledge Base

Line #	Knowledge Base	Utterance #	Stereotype	Knowledge Base	Utterance #
1	Seller	103	Instantiation	Application	103
2	Seller	104	Instantiation	Application	104

Table 11.21 Sellers' mapping

Application

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	10:00	B	Add	Def	Assert	Transport Means		Has		ID	1	1	1	B	Inst	Assert		
101	11	10:00	B	Add	Exp	Assert	Transport Means		Uses	Transport Mode	{road, rail, water, air}		0	1	B	Inst	Assert		
102	12	10:00	B	Add	Exp	Assert	Transport Means		Carries	Load type	{persons, cargo, both}		0	1	B	Inst	Assert		
103	100	10:01	B	Add	Def	Assert	Freight_ Transport Means		Has		ID	1	1	1	B	Inst	Assert		
104	101	10:01	B	Add	Def	Assert	Freight_ Transport Means		Use	Transport Mode	{road, rail, water, air}		0	1	B	Inst	Assert		
105	102	10:01	B	Add	Def	Assert	Freight_ Transport Means		Carries	Load Type	cargo		0	1	B	Inst	Assert		
200	103	12:30	B	Add	Inst	Assert	123_ Freight_ Transport Means		Has		451234								
201	105	12:30	B	Add	Inst	Assert	123_ Freight_ Transport Means		Use		road								

Table 11.22 Application Knowledge Base

In the example in the common knowledge base a concept Road_ Transport Means is defined. It uses a transport mode of road, but may carry cargo or persons. In the application knowledge base a concept Freight_ Means of Transport is defined, which may use various modes but only may carry persons. The ID of a Road_ Transport Means is mapped on the ID of a Freight_ Transport Means. The implicit Transport Mode of the Road_ Transport Means (value: road) is mapped on the Transport Mode of the Freight_ Transport Means. As the Load Type of the Freight_ Transport Means always has the value ‘cargo’, Road_ Transport Means with other Load Types will not be mapped.

Disjunction
When the intentional and extensional sets of received and to-be-stored concept are disjoint, the data obviously cannot be mapped. Another stored concept must be found or defined.

A special case arises when in two applications the property directions are opposite to each other. For instance in one application a Car is treated as the property (ownership) of a Person, while in the other application the Person is a property (owner) of the car. In fact this is not leading to a semantic mismatch. Even, in a knowledge base that adheres to the

natural language rules on Verbs and Roles (see section 6.2), the semantic relation between a Car and a Person is the same, irrespective of the property direction. The same Verb and the same thematic roles apply.

11.7 Central repository

It seems not feasible if each business relationship starts from the root utterance ('Thing has Property'). It is even undesirable if it would start from an REA based ontology, reinventing industry specific specialisations multiple times. It would be much more efficient (and effective) to store negotiated metadata in an open, central repository and to pick sub-ontologies from that repository when needed.

Such repository seems the right 'middle-way' between rigid standardization and re-inventing the wheel over and over again. The repository may even keep track of the use that is made of sub-ontologies, offering some reputation or popularity mechanism for meta-data.

11.8 Green field

The B2B knowledge base may be implemented directly in a business information (or Enterprise Resource) system. The concepts, defined by means of Definitions and Expansions then become object types in the system database. However, as in the course of the conversation more refined concepts may be defined, based on existing ones, the based-on relations must be retained. In traditional relational databases inheritance is not retained, as it is regarded as a technique at design time.

Queries over generic object types must also include specific objects that are based on the generic types. Queries over specific types should include generic objects that adhere to the criteria that make them specific. E.g. suppose originally Transport Equipment was defined as an object type, and several instances have been created. Transport Equipment includes pallets, boxes and containers. The size of the instances is stored. Suppose later on the concept Box is created, based on Transport Equipment and with limits to the Transport Equipment dimensions. Then a query over the Box object type should include other Transport Equipment with dimensions that fall inside the defined range for boxes.

11.9 Example

In section 1.5 (figure 1.3) we cited Lehmanns example dialog [3] as a challenge to be supported by the architecture proposed in this thesis. In this section is illustrated how the dialog may be supported by a B2B knowledge base. The dialog is conducted between two business information systems: the system of the US Chaplaincorps and a supplier of Explosives. The US Chaplaincorps has previously published an RFP for Votary Candles and Candle Holders.

A. Identify Parties
Bran: I am **BransonExplosives**. My **DUNS** number is **1234567.**
My encrypted signature key is **AAAAAAA**.
I respond to your **REQUEST FOR PROPOSALS #244** for **100,000 VOTARY CANDLES** and **10,000 CANDLE-HOLDERS**.
Chap: I am **USMC-Chaplaincorps-Procurement**.
My encrypted signature key is **BBBBBBBB**.
I accept that you are **BransonExplosives**.

Figure 11.6 Identify Parties

Identity can be verified using the certificates that come with a Public Key Infrastructure. The technology for such infrastructure is available, but organizationally the service providers are not yet world-wide interconnected. Electronic signatures may be set at message or utterance level and/or at network connection level. For identification purposes the highest stack-level should be used (so message level has preference over connection level), as lower level communication may have been delegated to service providers. Identity of sender and receiver of business information are however not primarily stated in the business information itself, but on the envelopes that convey it. The mechanism for encoding identity (e.g. by means of special administrative data elements) is dependent of the syntax solution used. UN/CEFACT has specified such set of elements in the Standard Business Document Header [4], which can be regarded as a set of requirements for a syntax solution. In the UN/CEFACT XML Naming and Design Rules [5] these requirements are translated into XML elements.

In other words, the identification of the business partners in a B2B dialog is positioned at the level of the envelope. The envelope covers the information that is included in the knowledge base.

B. Prior Accord
Chap: Have we dealt before?
Bran: Not directly. I sold training warheads to your parent system **USMC-LOGISTIC-BARSTOW** on **9/9/1996.**
Chap: **USMC-LOGISTIC-BARSTOW** just confirmed that to me, I accept it.

Figure 11.7 Prior Accord

Reputation can be verified by using governmental or private certification services, or by consulting specific reputation services that collect good and bad experiences from trading partners. In the example dialog the reputation claims and check occur prior to the agreement on a basic ontology. Presumably this policy has been chosen in order to avoid spam-like exchanges. Organizations are not prepared to negotiate ontologies with communication partners that later appear to be phony spammers. The reputation check then should be built in the technical communication protocol or in the syntax enveloping mechanism. If it is to be part of the business information itself it first must be defined as a normal business dialog (and confirmed by the partner). Here we assume that the Buyer checks the reputation of the Seller externally, after having identified him. With regard to

the knowledge base, this is an internal activity of the Buyer that needs not to be shared with the Seller.

Bran: Three ontological protocols were agreed upon: generic **MONEY**, **TIME** and a customized **SAFETY** agreement.
Chap: I don't have **SAFETY**; I inherit **TIME** and **MONEY** from **USMC-LOGISTIC-BARSTOW**, which have not changed.
Bran: **TIME** and **MONEY** have not changed for me either. Let's agree to use our earlier **TIME** and **MONEY** ontologies for our transactions.
Chap: Agreed.

Figure 11.8 Core ontologies

Common core ontologies can be exchanged using the initial knowledge base. This knowledge base can be repeated in the conversation, or some reference can be made to it. Whether the (meta-)information is repeated or referenced is a technical implementation. On semantic level the ontology is agreed upon.

C. Common Grounding
Chap: I have access to the **CCAT** core ontologies:
SPACE, PART-WHOLE, ABSTRACT-ALGEBRA, EVENT-OBJECT-PROCESS, CAUSALITY, SITUATIONS, REPRESENTATION, MEASUREMENT-UNITS and **DEEP-CASE.**
I have **CCAT** non-core ontologies **GENERAL THESAURI, DIGITAL SYSTEMS, INFORMATION SYSTEMS, GEOMETRY, MATERIALS, HUMAN-ACTIVITY, QUASIRATIONAL-AGENT, ENTERPRISE MODELS, TRADE ACTIVITIES,** and **ADDRESSES.**
I have **CYC** ontologies **TYPICALAMERICAN, COMMERCESTUFF** and **GOVERNMENTWORK.**
For English words I have **ROGET-TAGGED.** I have ...
Bran: I too have access to those **CCAT** ontologies except for **GEOMETRY, MATERIALS.** Of the **CYC** ontologies, I have only **COMMERCESTUFF.**

Figure 11.9 Common Grounding

The initial knowledge base can then be supplemented with knowledge (definitions of concepts) that is specific for the trade context. This may concern the products and services that are traded, geopolitical specifics or other concepts that may be relevant to the trade relation. Business partners translate their own policy into profiles that may be matched or negotiated using the mechanisms described in chapter 10. The concepts to be defined can be taken from ontologies that were agreed in the applicable industry or that were published by other companies or organizations.

This results in a B2B knowledge base with definitions of all basic concepts. The basic concepts can then be refined for the specific products and services to be agreed upon.

D. Term Definitions
Chap: My special **PAYMENT-TERMS** for procurement are **NEXT-QUARTER.**
Bran: I have only **EDIFACT PAYMENT-TERMS** as listed in Element 4279; there is no EDIFACT data code there called **"NEXT-QUARTER".**
Chap: I will define it for you in terms of our shared **TIME** and **MEASUREMENT-UNITS** ontologies. See the formal ontological definition of **QUARTER (EDIFACT** Data Element Value **2151:3M**). Any **YEAR** has **4 NONOVERLAPPING OFFICIAL TIME-PERIODS** of **3 MONTHS** each, called **QUARTERS,** consisting of the **JANUARY** to **MARCH** period, the **APRIL** to **JUNE** period,...
If an **INVOICE** is **RECEIVED** by us on a **DATE,** one **MONTH** is **ADDED** to that **DATE**; the resulting **DATE** occurs **WITHIN** a **QUARTER** and we **PAY** the **INVOICE** in **US-MONEY** by **MAILED CHECK** to the **SELLER** on the **LAST DAY** of the **QUARTER NEXT AFTER** that **QUARTER.**
Bran: Understood and Agreed.

Figure 11.10 Term Definitions

The definition of "Next Quarter" Payment Date is represented in a B2B knowledge base in table 11.23.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
100	10	00:10	B	Add	Def	Assert	D_ Date		has	Format	"YYYY-MM-DD"	1	1	1	B,S		All		
101	10	00:10	B	Add	Def	Assert	Q_ Date		has	Format	"YYYY-QQ"	1	1	1	B,S		All		
102	50	00:10	B	Prop	Exp	Assert	Invoice		has	Date	I_ D_ Date		1	1	S	Inst	Assert		
103	51	00:10	B	Prop	Exp	Assert	Payment		has	Date	P_ Q_ Date		1	1	B	Inst	Assert	100	
104	9	00:10	B	Prop	Def	Assert	P_ Q_ Date		has	Content	I_ D_ Date + 1 Month 1 Quarter		1	1	B	Inst	Assert		103
102	50	00:20	S	Acc	Exp	Assert	Invoice		has	Date	I_ Date		1	1	S	Inst	Assert		
103	51	00:20	S	Acc	Exp	Assert	Payment		has	Date	P_ Q_ Date		1	1	B	Inst	Assert	100	
104	9	00:10	S	Acc	Def	Assert	P_ Q_ Date		has	Content	I_ D_ Date + 1 Month 1 Quarter		1	1	B	Inst	Assert		103

Table 11.23 Definitions

The Buyer defines in Utt# 100 and 101 two date formats: P_ Date and Q_ Date. P_ Date is formatted as a normal ISO 8601 date (the CCTS Date data-type has a Supplementary Component that refers to an ISO 8601 format string). Q_ Date is formatted as a quarter, which is defined in ISO 8601 as YYYY-QQ. Then an Invoice. Date is defined as a P_ Date and a Payment. Date as a Q_ Date. The content of the Payment. Date is specified as the Invoice. Date plus one month plus one quarter. Payment is to take place within the resulting quarter. In Utt # 102 – 104 the Buyers' proposals are accepted by the Seller.

Bran: Your Request for Proposals requires **100,000 "VOTARY CANDLES"** and **10,000 "CANDLE HOLDERS"**; I can supply **100,000 "ROMAN CANDLES"** and **10,000 "CANDLE HOLDERS"**. What exactly is a **"VOTARY CANDLE"**?

Chap: A **"VOTARY CANDLE"** is a **CYLINDRICAL OBJECT** with a **"WICK"** which is to be **LIGHTED** and **BURNED.** Its **PURPOSE** is **BURNING** from one **END** to the other, thereby **RADIATING LIGHT** to be seen by **PERSONS.**

Bran: My **"ROMAN CANDLE"** is a **CYLINDRICAL OBJECT** with a **"FUSE"** which is to be **LIGHTED** and **BURNED.** Its **PURPOSE** is **BURNING** from one **END** to the other, thereby **RADIATING LIGHT** to be **SEEN** by **PERSONS.** Does my **"FUSE"** mean your **"WICK"**?

Chap: A **"WICK"** is a **PIECE** of **STRING** which is **LIGHTED** and **BURNED** at one **END** so as to **LAST** a **TIME-PERIOD.**

Bran: So is a **"FUSE".** My **"ROMAN CANDLES"** may comply with your **REQUEST FOR PROPOSALS**. What is **"VOTARY"?**

Chap: **"VOTARY"** means something which is **BROUGHT** to an **ALTAR** by a **PERSON** for a religious **PURPOSE.** A typical **VOTARY CANDLE** is made of **BEESWAX.** and it **BURNS QUIETLY** for **12 HOURS** to **36 HOURS.**

Bran: My **ROMAN CANDLES** could be brought by a **PERSON** to an **ALTAR.** A typical **ROMAN CANDLE** is made of **GUNPOWDER** and it **BURNS LOUDLY** in from **0.25** of a **SECOND** to **3 MINUTES.** My **"CANDLE HOLDERS"** are **METAL** and fit within your specified **PDES/STEP SHAPE** and **MATERIALS** definition for **"CANDLE HOLDER".**

Figure 11.11 Product definitions

The product definitions of Votary Candle and Roman Candle are represented in a B2B knowledge base in table 11.24.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
200	70	1:00	B	Prop	Def	Assert	Candle		has	Shape	Cylinder		1	1	B,S		All		
201	71	1:00	B	Prop	Def	Assert	Candle		has	Purpose	Light Radiation		1	1	B,S		All		
202	72	1:00	B	Prop	Def	Assert	Candle	Theme	make	Ingredient	Substance		1	1	B,S		All		
203	73	1:00	B	Prop	Def	Assert	Candle	Patient	burn	Duration	Measure		1	1	B,S		All		
204	74	1:00	B	Prop	Def	Assert	Candle		has	Part	Physical_ Entity		0	n	B,S		All		
205	204	1:00	B	Prop	Def	Assert	Votary_ Candle		has	Part	Wick		1	1	B,S		All		
206	75	1:00	B	Prop	Def	Assert	Wick	Patient	burn	Agent	Person		0	n	B,S		All		
207	76	1:00	B	Prop	Def	Assert	Votary_ Candle	Theme	bring	Agent	Person		0	n	B,S		All		
208	77	1:00	B	Prop	Def	Assert	Votary_ Candle	Theme	bring	Destination	Altar		0	n	B,S		All		
209	202	1:00	B	Prop	Def	Assert	Votary_ Candle	Theme	make	Ingredient	Beeswax		1	1	B,S		All		
210	203	1:00	B	Prop	Def	Assert	Votary_ Candle	Patient	burn	Duration	V_ Measure		1	1	B,S		All		
211	20	1:00	B	Prop	Def	Assert	V_ Measure		has	Measure Unit	Hour		1	1	B,S		All		
212	21	1:00	B	Prop	Def	Assert	V_ Measure		has	Min_ Facet	12 Hour		1	1	B,S		All		
213	22	1:00	B	Prop	Def	Assert	V_ Measure		has	Max_ Facet	36 Hour		1	1	B,S		All		
214	204	1:10	S	Prop	Def	Assert	Roman_ Candle		has	Part	Fuse		1	1	B,S		All		
215	75	1:10	S	Prop	Def	Assert	Fuse	Patient	burn	Agent	Person		0	1	B,S		All		
216	202	1:10	S	Prop	Def	Assert	Roman_ Candle	Theme	make	Ingredient	Gunpowder		1	1	B,S		All		
217	203	1:10	S	Prop	Def	Assert	Roman_ Candle	Patient	burn	Duration	R_ Measure		1	1	B,S		All		
218	20	1:10	S	Prop	Def	Assert	R_ Measure		has	Measure Unit	Second		1	1	B,S		All		
219	21	1:10	S	Prop	Def	Assert	R_ Measure		has	Min_ Facet	0.25 Second		1	1	B,S		All		
220	22	1:10	S	Prop	Def	Assert	R_ Measure		has	Max_ Facet	180 Second		1	1	B,S		All		

Table 11.24 Knowledge base

In Utt# 200 – 213 the Buyer defines a Votary Candle. A Votary Candle is made of Beeswax and burns during 12 to 36 hours. In Utt# 214 to 219 the Seller defines a Roman Candle. A Roman Candle is made of Gunpowder and burns during 0.25 to 180 seconds. Irrespective whether a "Wick" is equivalent to a "Fuse", the specifications do not match or overlap, as both the material, the candle is made of and the burn times are different.

E. Assess Mappings
Bran: I understand **PAYMENT-TERM: NEXT-QUARTER** since our definitions are now logically equivalent. This is a perfect mapping. My **CANDLE-HOLDERS** fully comply with your **PDES/STEP** specification.
Chap: Yes, agreed.
Bran: OK. Our strict definitions of **"CANDLES"** are logically inequivalent but not inconsistent. The concepts could overlap.
Chap: I require more than possible overlap for this **REQUEST FOR PROPOSALS.** I require an **"EGG/YOLK"** mapping reliability level 13" or better for the **VOTARY-CANDLES** concept. *[in EGG/YOLK reliability theory for data mapping (see [23]), level 13 requires at least an overlap between the typical instances of both concepts]*
Your typical candle **BURNS LOUDLY** in from **0.25** of a **SECOND** to **10 MINUTES;** my typical candle **BURNS QUIETLY** for **12 HOURS** to **36 HOURS.** The intersection of these typical classes of candles is empty, so the reliability of the class-mapping is less than EGG/YOLK level 13. Apparently I must reject it.

Figure 11.12 Assess Mappings

In a B2B knowledge base the absolute ranges of properties are specified, not the fuzzy range of a 'typical' representative of a concept. However, it would be possible to state typical ranges and use statistical measures such as the Egg/Yolk reliability measure. This is however not illustrated.

F. Reconcile Differences
Chap: My requirement for agreement on **"CANDLES"** precludes my accepting that your **ROMAN-CANDLES** are **VOTARY-CANDLES** because **EGG/YOLK** reliability level 13 is not achieved.
Bran: Will you accept the risk that my **"CANDLES"** are incompatible with your **"CANDLES"** if I offer them at a deep discount?
Chap: No. I will not accept that **ROMAN-CANDLES** means **VOTARY-CANDLES** at any price.

Figure 11.13 Reconcile Differences

Here metadata negotiation and price negotiation are mixed. This would not be possible in an environment where metadata is standardized beforehand. In a B2B knowledge base the two negotiation topics are treated similar and may be mixed.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
231	75	1:20	S	Prop	Exp	Assert	Candle	Instrument	has	Price	Amount		0	n	B,S		All		
232	231	1:20	S	Prop	Def	Assert	Cheap_ Roman_ Candle	Instrument	has	Price	Low_ Amount		0	n	B,S		All		
233	78	1:20	S	Prop	Exp	Assert	Votary_ Candle	theme	replace	Instrument	Cheap_ Roman_ Candle		0	n	B,S		All		232
231	75	1:30	B	Acc	Exp	Assert	Candle	Instrument	has	Price	Amount		0	n	B,S		All		
232	231	1:30	B	Acc	Def	Assert	Cheap_ Roman_ Candle	Instrument	has	Price	Low_ Amount		0	n	B,S		All		
233	78	1::30	B	Rej	Exp	Assert	Votary_ Candle	theme	replace	agent	Cheap_ Roman_ Candle		0	n	B,S		All		232

Table 11.25 Negotiation

In Utt#231 the Seller proposes a Price property to add to the Candle concept. This is accepted by the Seller at 1:30. The Seller also proposes to define a Cheap_ Roman_ Candle with a Low_ Amount Price. Presumably "Low_ Amount" has been defined as a data type previously. This definition is also accepted. But then the Seller proposes the Cheap_ Roman_ Candle concept to be a replacement of the Votary_ Candle. This is rejected by the Buyer.

G. Agree on Transactions

Chap: I require from you a **PROPOSAL** for **10,000 CANDLE-HOLDERS** only (no **CANDLES);** if it is satisfactory then I will send you a binding **EDIFACT**-style **"ORDERS"** purchase order; you will confirm with **EDIFACT** form **"ORDRSP".** Then you will ship me the **CANDLE-HOLDERS** in boxes bar-coded as **SHIPMENTS** with a **MANIFEST** message. Then you will send **EDIFACT "ADVANCE-SHIPPING-NOTICE"** and **"INVOICE"** to me for Payment. This will be done for each box of 100 **CANDLE-HOLDERS.** Payment terms will be **NEXT-QUARTER** as we agreed**.**

Bran: Yes, but I want to ship in lots of 1000 instead of lots of 100.

Chap: Agreed.

Figure 11.14 Agree on Transactions

In this section of the dialog the process is defined. In a B2B knowledge base the process is specified by means of preconditions. This is illustrated in table 11.26.

Realization

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
300	80	2:00	B	Prop	Def	Assert	Order		has	Product	Candle Holder		1	1	B	Inst	Order		
301	81		B				Order		has	Quantity	100_ Box_ Measurement		1	1	B	Inst	Order		
302	82	2:00	B	Prop	Def	Assert	Order Response		references		Order				S	Inst	Commit	300	
303	82	2:00	B	Prop	Def	Assert	Manifest		references		Order				S	Inst	Assert	302	
304	82	2:00	B	Prop	Def	Assert	Advanced Shipping Notice		references		Order				S	Inst	Assert	303	
305	82	2:00	B	Prop	Def	Assert	Invoice		references		Advanced Shipping Notice				S	Inst	Order	304	
306	82	2:00	B	Prop	Def	Assert	Payment		references		Invoice				B	Inst	Assert	305	
300	80	2:10	S	Acc	Def	Assert	Order		has	Product	Candle Holder		1	1	B	Inst	Order		
301	81	2:10	S	Rej			Order		has	Quantity	100_ Box_ Measurement		1	1	B	Inst	Order		
302	82	2:10	S	Acc	Def	Assert	Order Response		references		Order				S	Inst	Commit	300	
303	82	2:10	S	Acc	Def	Assert	Manifest		references		Order				S	Inst	Assert	302	
304	82	2:10	S	Acc	Def	Assert	Advanced Shipping Notice		references		Order				S	Inst	Assert	303	
305	82	2:10	S	Acc	Def	Assert	Invoice		references		Advanced Shipping Notice				S	Inst	Order	304	
306	82	2:10	S	Acc	Def	Assert	Payment		references		Invoice				B	Inst	Assert	305	
307	81	2:10	S	Prop			Order		has	Quantity	1000_ Box_ Measurement		1	1	B	Inst	Order		
307	81	2:20	B	Acc			Order		has	Quantity	1000_ Box_ Measurement		1	1	B	Inst	Order		

Table 11.26 Agree on transaction

In Utt# 300 the Buyer defines an Order for Candle Holders. He specifies in Utt# 301 that the quantity will be expressed in boxes of 100 pieces. This measurement presumably has been defined in the basic ontology previously. Then he defines that the Order is to be followed by an Order Response, a Manifest, an Advanced Shipping Notice, an Invoice and a Payment. Each document has the existence of an instance of another document as precondition (column 19). The Seller accepts all definitions, except the quantity. He proposes to express the quantity in boxes of 1000 pieces, which is accepted by the Buyer.

After these documents have been defined, they may be instantiated and the trade operations may start. The instantiations of the various documents (or transactions) are based on the definitions.

H. Agree on Data Required
Chap: Does your proposed **INVOICE** contain the **PARTYs,** their **ADDRESSes,** the **INVOICE-DATE,** some **REPRESENTATION** of the **PRODUCT,** a **SHIPPER** and a **PRICE** in **US DOLLARS?**
Bran: All but the **SHIPPER.**
Chap: Add the **SHIPPER** to your invoice form and I will accept it.
Bran: Agreed. **SHIPPER** is defined in the **TRADE-ACTIVITIES** ontology and in my local database meta-data; I know my **SHIPPERS.**
Which data do you normally include in your **EDIFACT "ORDERS"** purchase order form?
Chap: All the **EDIFACT "ORDERS"** fields with non-empty values.
Bran: All I need is **0030** (date-time segment), **0120** (party segment), **960-STEP** (item description segment, but in **STEP** terms)**, 980** (quantity segment), and **1150** (price segment).
Chap: OK I'll skip all the rest**.**

Figure 11.15 Agree on Data Required

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
320	85	3:00	B	Prop	Exp	Assert	Invoice		has	Buyer	Party		1	1	S	Inst	Assert		305
321	85	3:00	B	Prop	Exp	Assert	Invoice		has	Seller	Party		1	1	S	Inst	Assert		305
322	85	3:00	B	Prop	Exp	Assert	Invoice		has	Shipper	Party		1	1	S	Inst	Assert		305
323	90	3:00	B	Prop	Exp	Assert	Party		has	Postal	Address		1	1	S	Inst	Assert		305
324	83	3:00	B	Prop	Exp	Assert	Invoice		has	Issue	Date		1	1	S	Inst	Assert		305
325	84	3:00	B	Prop	Exp	Assert	Invoice		has	Product	Product		1	1	S	Inst	Assert		305
326	86	3:00	B	Prop	Exp	Assert	Invoice		has	Price	USD_ Amount		1	1	S	Inst	Assert		305
320	85	3:10	S	Acc	Exp	Assert	Invoice		has	Buyer	Party		1	1	S	Inst	Assert		305
321	85	3:10	S	Acc	Exp	Assert	Invoice		has	Seller	Party		1	1	S	Inst	Assert		305
322	85	3:10	S	Acc	Exp	Assert	Invoice		has	Shipper	Party		1	1	S	Inst	Assert		305
323	90	3:10	S	Acc	Exp	Assert	Party		has	Postal	Address		1	1	S	Inst	Assert		305
324	83	3:10	S	Acc	Exp	Assert	Invoice		has	Issue	Date		1	1	S	Inst	Assert		305
325	84	3:10	S	Acc	Exp	Assert	Invoice		has	Product	Product		1	1	S	Inst	Assert		305
326	86	3:10	S	Acc	Exp	Assert	Invoice		has	Price	USD_ Amount		1	1	S	Inst	Assert		305
327	83	3:10	S	Prop	Exp	Assert	Order		has	Issue	Date Time	1	1	1	B	Inst	Assert		300
328	85	3:10	S	Prop	Exp	Assert	Order		has	Buyer	Party		1	1	B	Inst	Assert		300
329	85	3:10	S	Prop	Exp	Assert	Order		has	Seller	Party		1	1	B	Inst	Assert		300
330	90	3:10	S	Prop	Exp	Assert	Product		has	Description	STEP_ Description		1	1	B	Inst	Order		300
331	91	3:10	S	Prop	Exp	Assert	Order		has	Price	Amount		1	1	B	Inst	Order		300
327	83	3:20	B	Acc	Exp	Assert	Order		has	Issue	Date Time	1	1	1	B	Inst	Assert		300
328	85	3:20	B	Acc	Exp	Assert	Order		has	Buyer	Party		1	1	B	Inst	Assert		300
329	85	3:20	B	Acc	Exp	Assert	Order		has	Seller	Party		1	1	B	Inst	Assert		300
330	90	3:20	B	Acc	Exp	Assert	Product		has	Description	STEP_ Description		1	1	B	Inst	Order		300
331	91	3:20	B	Acc	Exp	Assert	Order		has	Price	Amount		1	1	B	Inst	Order		300

Table 11.27 Agree on data required

In Utt# 320 through 326 the Buyer expresses his information requirements for the Invoice to be sent by the Seller. These are accepted by the Seller, who expresses his requirements for the Order in Utt# 327 through 331. The Buyer accepts those. The Quantity as information element of the Order already had been defined in table 11.26.

I. Agree on Formats
Chap: I can send my purchase orders in the usual **EDIFACT** format.
Bran: Don't bother. Just use a flat file with tagged fields: XXX, YYY, and ZZZ, comma-delimited and in any order.
As a military agency you use a 24 hour clock. Convert it to 12 hour for these transactions, 4 numeric characters followed by 1 alpha character: **HHMM{A/P}.**
Chap: Agreed.

Figure 11.16 Agree on Formats

J. Agree on Channels
Chap: I use the **XYZ** VAN service, or encrypted **MIME** email.
Bran: Use encrypted **MIME** email.

Figure 11.17 Agree on Channels

As for the negotiation dialog already some syntax is used, the same syntax may be deployed for the operations. As mentioned in chapter 2, syntaxes used for B2B communication converge to XML, using specific naming and design rules. A handshake protocol should be in place to agree on network connection type and syntax. If (for reasons of performance or otherwise) different transaction types need different syntax solutions or technical communication mechanisms, such requirements may be stated in a Business Header. Header information may be exchanged and negotiated in a technical protocol (such as the ebXML Collaboration-Protocol Profile and Agreement Specification [6]).

K. Agree on Liability
Chap: We will be bound by **CYBER-UCC-500 SCHEDULE 6** for government buyers. You will be bound to perform thereunder, and in addition to indemnify us against any and all damage claims by third parties.
Bran: No. We will not indemnify you for "any and all damage claims" by third parties; we will only be liable for performance, breach, and actual damages due to negligence in manufacturing, as is already provided by **CYBER-UCC-500 SCHEDULE 6.**
Chap:*[Checking perhaps with a person or an expert system]* Agreed.
Bran: We have agreed on everything necessary for our negotiated series of binding transactions. Let us commence. My **PROPOSAL** will follow.

Figure 11.18 Agree on Liability

The general conditions under which the trade takes place should be defined in the trade negotiation itself. A reference to such conditions (in this case CYBER-UCC-500 SCHEDULE) should be made when agreeing the trading terms.

The definitions of trade documents and information elements are the basis for operational transactions. See table 11.28.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
500	327	12:00	B	Add	Inst	Order	1_ Order		has	Issue	2011-07-26								
501	307	12:00	B	Add	Inst	Order	1_ Order		has	Quantity	10								
502	328	12:00	B	Add	Inst	Order	1_ Order		has	Buyer	B								
503	329	12:00	B	Add	Inst	Order	1_ Order		has	Seller	S								
504	300	12:00	B	Add	Inst	Order	1_ Order		has	Product	Candle Holders								
505	331	12:00	B	Add	Inst	Order	1_ Order		has	Price	USD 100								
506	302	14:00	S	Add	Inst	Commit	1_ Order Response		references		1_Order								

Table 11.28 Transaction

References

[1]. Visser c.s.: Assessing Heterogeneity by Classifying Ontology Mismatches, AAAI'97 Spring Symposium on Ontological Engineering, Stanford University, USA

[2]. Moore: Testing Speech Act Theory and its applicability to EDI other computer-processable messages, Proceedings of the 29th Hawaii International Conference on System Sciences, Volume 2: Decision Support and Knowledge-Based Systems, 1996

[3]. Lehmann: Machine-Negotiated, Ontology-Based EDI. In: Proceedings of CIKM-94 Workshop on Electronic Commerce, Springer, 1995

[4]. United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT Standard Business Document Header (SBDH)

[5]. United Nations Centre for Trade Facilitation and Electronic Business: UN/CEFACT XML Naming and Design Rules Technical Specification, version 3.0, 2010

[6]. OASIS ebXML Collaboration Protocol Profile and Agreement Technical Committee: Collaboration-Protocol Profile and Agreement Specification Version 2.0, 2002

PART III IMPLEMENTATION

12 Cases from various sectors

Summary

The mechanisms, described in previous chapters, are in this chapter applied to various situations. This is done in the form of case descriptions. Three cases are presented: a case for the exchange of electronic invoices, a case in which the characteristics of a product are negotiated and a case for open tracking and tracing in transportation networks. In each case the knowledgebase as designed in previous chapters is populated in the course of a business conversation.

The cases proof that the structure of the knowledgebase and the mechanism to populate it can be applied in various business environments.

12.1 e-Invoicing

The first case describes a simple situation in which two companies wish to exchange electronic invoices. As a basis for the information to exchange they use a schema, issued by a standardisation organization. However, they slightly customise the information structure, based on the specifics of the product that is invoiced and on the administrative procedures of the parties.

The Seller sells office supplies to the Buyer. The products are delivered at the desk of the employee who ordered them. The Seller is assumed to mention the name of the ordering employee and the cost account number of the employees' department on the invoice. Invoices are sent monthly.

The standard invoice looks as in figure 12.1:

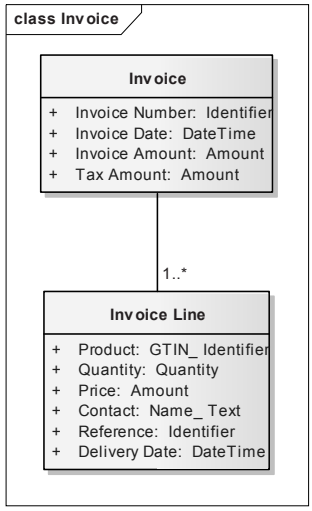


Figure 12.1 Standard Invoice Class diagram

This can be represented in a B2B KB table as:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
1	0	0:00	S	Add	Definition	Assert	Invoice		Invoicing	ID	Invoice Number	1	1	1	S	Instantiation	Order		
2	0	0:00	S	Add	Definition	Assert	Invoice		Invoicing	Date_ Time	Invoice Date		1	1	S	Instantiation	Order		1
3	0	0:00	S	Add	Definition	Assert	Invoice		Invoicing	Property	Invoice Amount		1	1	S	Instantiation	Order		1
4	0	0:00	S	Add	Definition	Assert	Invoice		Invoicing	Property	Tax Amount		1	1	S	Instantiation	Order		1
5	0	0:00	S	Add	Definition	Assert	Invoice		Invoicing	Part	Invoice line		1	n	S	Instantiation	Order		1
6	0	0:00	S	Add	Definition	Assert	Invoice line		Invoicing	Product_ Property	GTIN_ Identifier	1	1	1	S	Instantiation	Order		1
7	0	0:00	S	Add	Definition	Assert	Invoice line		Invoicing	Quantity_ Property	Quantity		1	1	S	Instantiation	Order		1
8	0	0:00	S	Add	Definition	Assert	Invoice line		Invoicing	Price_ Property	Amount		1	1	S	Instantiation	Order		1

Cases from various sectors

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
9	0	0:00	S	Add	Definition	Assert	Invoice line		Invoicing	Contact_ Property	Name_ Text		0	n	S	Instantiation	Order		1
10	0	0:00	S	Add	Definition	Assert	Invoice line		Invoicing	Reference _ Property	Identifier		0	n	S	Instantiation	Order		1
11	0	0:00	S	Add	Definition	Assert	Invoice line		Invoicing	Delivery_ Date_ Time	DateTime		1	1	S	Instantiation	Order		1

Table 12.1 Knowledge base

The specific Buyer requirements can be represented in UML as in figure 12.2.

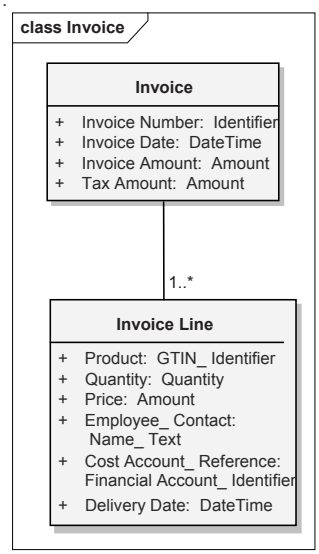


Figure 12.2 Specific Invoice Class diagram

To adapt the invoice to the requirements of the trade relationship, the following lines are added to the table:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
12	9	0:00	B	Propose	Expansion	Assert	Our_ Invoice line		Invoicing	Employee_ Contact_ Property	Name_ Text		1	1	S	Instantiation	Order		1
13	10	0:00	B	Propose	Expansion	Assert	Our_ Invoice line		Invoicing	Cost Account_ Reference Property	Financial Account_ Identifier		1	1	S	Instantiation	Order		1
14	9	0:00	B	Propose	Restriction	Assert	Our_ Invoice line		Invoicing	Contact_ Property	Name_ Text		0	0	S	Instantiation	Order		1
13	10	0:00	B	Propose	Restriction	Assert	Our_ Invoice line		Invoicing	Reference _ Property	Identifier		0	0	S	Instantiation	Order		1

Table 12.2 Invoice adaptation

Using the UN/CEFACT Core Component methodology, the XML Schema of the invoice is adapted and extended with the two new (further qualified) Business Information Entities. The cardinality of the original BIE's is restricted, so they disappear from the instances. At the Seller side the new attributes are stored in the database fields of the attributes they are based on, with an indication (type code) of their specific meaning. At the buyer side they were already included in the database structure, as these were buyer requirements.

After the proposals have been accepted by the Seller the new attributes are used in invoice instances.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
101	1	9:00	S	Add	Instantiation	Order	123_ Invoice		Invoicing	ID	123								
102	2	9:00	S	Add	Instantiation	Order	123_ Invoice		Invoicing	Date_ Time	20110804								
103	3	9:00	S	Add	Instantiation	Order	123_ Invoice		Invoicing	Property	100.00								
104	4	9:00	S	Add	Instantiation	Order	123_ Invoice		Invoicing	Property	19.00								
105	5	9:00	S	Add	Instantiation	Order	123_ Invoice		Invoicing	Part	123-01_ Our_ Invoice line								

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
106	6	9:00	S	Add	Instantiation	Order	123-01_ Our_ Invoice line		Invoicing	Product_ Property	87123								
107	7	9:00	S	Add	Instantiation	Order	123-01_ Our_ Invoice line		Invoicing	Quantity_ Property	10								
108	8	9:00	S	Add	Instantiation	Order	123-01_ Our_ Invoice line		Invoicing	Price_ Property	10.00								
109	12	9:00	B	Propose	Instantiation	Order	123-01_ Our_ Invoice line		Invoicing	Employee_ Contact_ Property	Johnson								
110	13	9:00	B	Propose	Instantiation	Order	123-01_ Our_ Invoice line		Invoicing	Cost Account_ Reference_ Property	5678								
111	11	9:00	S	Add	Instantiation	Order	123-01_ Our_ Invoice line		Invoicing	Delivery_ Date_ Time	20110801								

Table 12.3 Invoicing

12.2 Buying and selling rivets

The second case describes a situation where a machine manufacturer in the Netherlands is searching supply for specific fasteners on the world market. Both technical and commercial aspects of the supply are assessed and negotiated with a potential supplier in China. After agreement of the conditions (including the operational business process for delivering the fasteners) the delivery process is monitored.

The accepted way of defining technical products is to agree among manufacturers on product ontologies. ISO 10303 (STEP) [1] defines an ontology language that is used by industry working groups to assemble an ontology for each product type. The number of types of industrial products is however huge and industrial innovation causes dynamics in a pace that cannot be met by international standard committees. The method to agree bilaterally on ontologies, as defined in this thesis avoids the inertia of international standardisation. However, in order to reach convergence of ontologies, the method is to be used complementary to established ontological standards.

The case starts with specification of the product and negotiation and delivery of a spot order. Afterwards the process is changed to repetitive deliveries, based on the material requirements of the machine manufacturer.

Initially the (potential) buyer publishes his desire to purchase the fasteners. The main challenge for professional procurement is to specify requirements functionally instead of technically. If requirements are specified functionally, suppliers get a chance for

innovation: for solving a customer's problem easier or cheaper by means of a new technology.

In this case the buyer specifies that he needs a way to fasten two flat plates permanently. He specifies the thickness of the plates and the force the fastening must be able to resist. He also specifies the material properties of the plates. He indicates when he needs the fasteners. The buyer bases his definitions on the basic ontology as introduced in chapter 9.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
101	16	1:00	B	Propose	Definition	Assert	Fastening_ Event		Fasten	Patient	Object		2	n	B,S		All		
102	16	1:00	B	Propose	Definition	Assert	Fastening_ Event		Fasten	Goal	Joint_ Physical_ Entity		1	n	B,S		All		
103	16	1:00	B	Propose	Definition	Assert	Fastening_ Event		Fasten	Instrument	Fastener_ Mechanical Device_ Artefact_ Physical_ Entity		0	n	B,S		All		
104	16	1:00	B	Propose	Definition	Assert	Fastening_ Event		Fasten	Instrument	Tool_ Object		0	n	B,S		All		
105	4	1:00	B	Propose	Definition	Assert	Plate_ Physical_ Entity		Has	Approximate Shape_ Property	Plate		1	1	B,S		All		
106	4	1:00	B	Propose	Definition	Assert	Plate_ Physical_ Entity		Has	Thickness_ Property	Length_ Measurement		0	n	B,S		All		
107	4	1:00	B	Propose	Definition	Assert	Plate_ Physical_ Entity		Has	Material_ Property	Metal_ Material		1	1	B,S		All		
108	101	1:00	B	Propose	Definition	Assert	Plate_ Fastening_ Event		Fasten	Patient	Plate_ Objects		2	2	B,S		All		
109	4	1:00	B	Propose	Definition	Assert	Joint_ Physical_ Entity		Has	Force_ Property	Force_ Measurement		0	n	B,S		All		
110	4	1:00	B	Propose	Definition	Assert	Joint_ Physical_ Entity		Has	Detachability_ Property	Boolean		0	1	B,S		All		

Table 12.4 Product definition

At this point the buyer has defined a Fastening event with its properties (for brevity only few properties of the fastening event are included in the conversation). The seller reacts by accepting the proposed definitions. After acceptance the trading partners have a common understanding of the universe of discourse: fastening plates.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
111	16	1:05	S	Accept	Definition	Assert	Fastening_ Event		Fasten	Patient	Object		2	n	B,S		All		
Same for Utterances 102 through 110																			
120	4	1:05	B	Accept	Definition	Assert	Joint_ Physical_ Entity		Has	Detachability_ Property	Boolean		0	1	B,S		All		

Table 12.5 Acceptance

Next the buyer states the specification of the fastening he needs to establish within his production process. The prefix "My_" is introduced here to identify the buyer's process; note that any prefix could be used that uniquely identifies the specific fastening within the scope of the conversation.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
150	108	1:10	B	Add	Instantiation	Desire	My_ Plate_ Fastening		Fasten	Patient	My_ Plate_ Object		2	2	S		Offer		
151	102	1:10	B	Add	Instantiation	Desire	My_ Plate_ Fastening		Fasten	Goal	My_ Joint_ Object				S		Offer		150
152	110	1:10	B	Add	Instantiation	Desire	My_ Joint_ Physical_ Entity		Has	Detachability_ Property	False				S		Offer		150
154	109	1:10	B	Add	Instantiation	Desire	My_ Joint_ Physical_ Entity		Fastens	Max_ Force_ Property	200 N				S		Offer		150
153	106	1:10	B	Add	Instantiation	Desire	My_ Plate_ Physical_ Entity		Has	Thickness_ Property	5mm				S		Offer		150
155	107	01:10	B	Add	Instantiation	Desire	My_ Plate_ Physical_ Entity		Has	Material	Steel				S		Offer		111

Table 12.6 Buyer specification

The buyer now has specified that he needs to fasten two 5 mm thick steel plates. The fastening is to be permanent and must be able to withstand a force of 200 N.

Then the seller proposes to use rivets for the fastening. He defines a rivet according to ISO 13584 (PLib) [2] as a "cylindrical metal fastener with a preformed head at one end, whereas the head at the other end is formed during setting, such creating a non-detachable

joint". So a Rivet is a Fastener. According to SUMO a Fastener is a Restraint, a Restraint is a Device, a Device is an Instrumentality and an Instrumentality is an Artefact. The full name of a Rivet is therefore Rivet_ Fastener_ Restraint_ Device_ Instrumentality_ Artefact_ Physical_ Entity. At each level of the inheritance tree the specialisations (Artefact, Mechanical Device, etc.) are defined by means of their unique properties. For brevity here only Rivets are defined as a special type of Fastener and be simply named as "Rivet", not as "Rivet_ Fastener_ Restraint_ Device_ Instrumentality_ Artefact_ Physical_ Entity". Definitions of Fastener, Restraint, etc. are omitted.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
150	60	02:00	S	Propose	Definition	Assert	Rivet		Has	Approximate Shape_ Property	Cylinder		1	1	B,S		All		
151	61	02:00	S	Propose	Definition	Assert	Rivet		Has	Shape_ Part	Rivet End		2	2	B,S		All		300
152	151	02:00	S	Propose	Definition	Assert	Rivet		Has	First End_ Shape_ Part	Preformed Head_ Rivet End		1	1	B,S		All		300
153	151	02:00	S	Propose	Definition	Assert	Rivet		Has	Second End_ Shape_ Part	Post formed Head_ Rivet End		1	1	B,S		All	200	
154	64	02:00	S	Propose	Expansion	Assert	Rivet		Has	Shank Diameter_ Property	Length_ Measurement		1	1	B,S		All		303
155	65	02:00	S	Propose	Expansion	Assert	Rivet		Has	Shank Break Force_ Property	Force_ Measurement		1	1	B,S		All		
156	66	02:00	S	Propose	Expansion	Assert	Rivet		Has	Shank Length_ Property	Length_ Measurement		1	1	B,S		All		
157	67	02:00	S	Propose	Expansion	Assert	Rivet_ Resource		Has	ID nr	Identifier	1	1	1	B,S		All		
158	68	02:00	S	Propose	Expansion	Assert	Rivet_ Resource		Has	Price	Amount		0	n	B,S		All		
159	103	02:00	S	Add	Instantiation	Offer	My_ Plate_ Fastening		Fasten	Instrument	Rivet				B		Order		
160	104	02:00	S	Add	Instantiation	Offer	My_ Plate_ Fastening		Fasten	Instrument	Riveting_ Tool				B		Order		
161	151	02:00	S	Add	Instantiation	Offer	My_ Plate_ Fastening		Fasten	Goal	Rivet_ Joint				B		Order		
162	60	02:00	S	Add	Instantiation	Offer	My_ Joint_ Physical_ Entity		Has	Detachability_ Property	False				B		Order		
163	60	02:00	S	Add	Instantiation	Offer	My_ Rivet		Has	Shank Diameter_ Property	4mm				B		Order		

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
164	60	02:00	S	Add	Instantiation	Offer	My_ Rivet		Has	Shank Break Force_ Property	500 N				B		Order		
165	60	02:00	S	Add	Instantiation	Offer	My_ Rivet		Has	Shank Length_ Property	10 mm				B		Order		
166	60	02:00	S	Add	Instantiation	Offer	My_ Rivet_ Resource		Has	ID nr	123456	1			B		Order		
167	60	02:00	S	Add	Instantiation	Offer	My_ Rivet_ Resource		Has	Price	EUR 0.02				B		Order		
168	60	02:00	S	Add	Instantiation	Offer	My_ Plate_ Fastening		Fasten	Instrument	My_ Rivet				B		Order		

Table 12.7 Rivet specifications

After having defined the properties of My_ Rivet as an instantiation of Rivet, the seller suggests the buyer to use that type of Rivet for his fastening process. This occurs in line 168.

The buyer then states he wishes to buy a certain quantity of the rivets specified. To state this he uses the pattern as presented in chapter 9.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
200	7	2:10	B	Propose	Expansion	Desire	Delivery		Delivery	ID nr	Identifier	X	1	1	S		Commit	303	
201	7	2:10	B	Propose	Expansion	Desire	Delivery		Delivery	Date_ Time	Delivery Date		0	1	S		Commit		300
202	7	2:10	B	Propose	Expansion	Desire	Delivery		Delivery	Line_ Part	Delivery Line		0	n	S		Commit		300
203	7	2:10	B	Propose	Expansion	Desire	Delivery		Delivery	Address_ Location	Identifier		0	1	S		Commit		
204	7	2:10	B	Propose	Definition	Desire	Delivery Line		Delivery	ID nr	Identifier	1	1	1	S		Commit		303
205	7	2:10	B	Propose	Expansion	Desire	Delivery Line		Delivery	Quantity_ Property	Quantity		0	1	S		Commit		
206	7	2:10	B	Propose	Expansion	Desire	Delivery Line		Delivery	Product_ Property	Product		0	1	S		Commit		

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
207	200	2:10	B	Add	Instantiation	Desire	123_ Delivery		Delivery	ID nr	123	1	1	1	S		Commit		
208	201	2:10	B	Add	Observation	State	123_ Delivery		Delivery	Variable_ Date Time	123_ Delivery Date		0	1	S		Commit		
209	202	2:10	B	Add	Observation	Desire	123_ Delivery		Delivery	Line_ Part	123-1		0	n	S		Commit		
210	203	2:10	B	Add	Observation	Desire	123_ Delivery		Delivery	Address_ Location	87001		0	1	S		Commit		
211	201	2:10	B	Add	Observation	Desire	123_ Delivery Date		Delivery	Date_ Time	20090523								
212	204	2:10	B	Add	Instantiation	Desire	123-1_ Delivery Line		Delivery	ID nr	123-1	1	1	1	S		Commit		
213	205	2:10	B	Add	Observation	Desire	123-1_ Delivery Line		Delivery	Quantity_ Property	50000		0	1	S		Commit		
214	206	2:10	B	Add	Observation	Desire	123-1_ Delivery Line		Delivery	Product_ Property	123456		0	1	S		Commit		
215	207	2:10	B	Add	Instantiation	Assert	87001_ Address		Delivery	ID nr	87001	1	1	1	S		Commit		
216	0	2:10	B	Add	Observation	Assert	87001_ Address		Delivery	Street_ Part	Koggekade 208		1	1	S		Commit		
217	0	2:10	B	Add	Observation	Assert	87001_ Address		Delivery	City_ Part	Zwolle		1	1	S		Commit		
218	0	2:10	B	Add	Observation	Assert	87001_ Address		Delivery	Country_ Part	NL		1	1	S		Commit		

Table 12.8 Delivery

Note that in line 208 the data type of the Delivery Date is defined to be a variable. It needs to be referred to when the payment date is specified. The actual delivery date is determining the payment date when delivery has taken place.

The seller then states his quotation: he commits to deliver the rivets under the condition that the buyer commits to pay upon delivery.

Cases from various sectors

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
300	16	2:20	S	Add	Instantiation	Desire	123_ Payment		Payment	Patient	EUR 1000				B		Commit	303	
301	16	2:20	S	Add	Observation	Desire	123_ Payment		Payment	Date_ Time	123_ Delivery Date +30 days				B		Commit		300
302	16	2:20	S	Add	Observation	Desire	123_ Payment		Payment	Account_ Method	9876543				B		Commit		300
303	200	2:20	S	Add	Instantiation	Commit	123_ Delivery		Delivery	ID nr	123	1	1	1	B		Order		
304	201	2:20	S	Add	Observation	Commit	123_ Delivery		Delivery	Date_ Time	20090501		0	1	S		Assert		303

Table 12.9 Payment

The buyer accepts and promises to pay 30 days after delivery.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
305	300	2:30	B	Add	Observation	Commit	123_ Payment		Payment	Patient	EUR 1000				B		Assert	307	
306	301	2:30	B	Add	Observation	Commit	123_ Payment		Payment	Date_ Time	Delivery Date +30 days				B		Assert	307	305
307	303	2:30	B	Add	Observation	Order	123_ Delivery		Delivery	ID nr	123				S		Assert		

Table 12.10 Payment acceptance

Finally the seller ships the rivets and notifies the buyer and the buyer pays.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
308	307	3:00	S	Add	Observation	Assert	123_ Delivery		Delivery	ID nr	123								
309	304	3:00	S	Add	Observation	Assert	123_ Delivery		Delivery	Date_ Time	20090430								
310	305	3:30	B	Add	Observation	Assert	123_ Payment		Payment	Patient	EUR 1000								
311	306	3:30	B	Add	Observation	Assert	123_ Payment		Payment	Date_ Time	Delivery Date +30 days								

Table 12.11 Instantiation

After delivery, the buyer is satisfied and wishes to have rivets delivered on a regular basis. His call-off orders are based on the same conversation that he had with the seller to have the initial consignment delivered.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
400	200	3:00	B	Add	Instantiation	Order	124_ Delivery		Delivery	ID nr	124						Assert		
401	201	3:00	B	Add	Observation	Order	124_ Delivery		Delivery	Date_ Time	20090530						Assert		
402	206	3:00	B	Add	Instantiation	Order	124_ Delivery Line		Delivery	Product_ Property	123456						Assert		
403	205	3:00	B	Add	Observation	Order	124-1_ Delivery Line		Delivery	Quantity_ Property	5000						Assert		

Table 12.12 Repeat order

In this scenario it has been shown that a trade agreement and a product specification can be negotiated and agreed using the KB table structure. In advance buyer and seller did not know each other, the buyer even did not know what products the seller offers. Ultimately a blanket order is in effect and rivets are regularly being supplied to the buyer.

In chapter 11 it has been shown how communication as represented in the KB table can be implemented in a message protocol and in automated information systems.

12.3 Tracking and tracing

The third case describes the tracking and tracing of products and components from manufacturing through to disposal. It is assumed that products can be marked with an identifier that can be read automatically (e.g. by means of barcode or RFID). Products flow through an open environment: in advance (at manufacturing time) it is not known where products will be sold and by whom products will be used. Product information owners maintain databases with product information that are open to information suppliers and to stakeholders requiring information.

In this case it is described how flexible and dynamic B2B systems can interoperate to supply the information needed to stakeholders.

Most tracking and tracing systems are closed. They are configured for a well-defined purpose and a well-defined user group. Open tracking and tracing is multi-purpose and not pre-configured for a limited group of participants. In open tracking and tracing items (products, components, transport units) are uniquely identified with tags (barcodes or RFID transponders) that are automatically readable. A communication system exists that allows the uploading of identifiers that are read, together with additional not predefined information, and that answers queries on the item information for all kinds of purposes. Purposes may be logistic control, but also asset management, environmental and legal.

TraSer is an open source platform that supports open tracking and tracing. TraSer consists of a mechanism to uniquely identify items, without centralized control, and of a set of web services for the upload and exchange of item information. TraSer has been a project, partly financed by the EU (IST FP6). At the time of the project the TraSer web services were rigidly programmed to the requirements of various pilot environments. In this section of this thesis it is described how the TraSer infrastructure truly can become open and dynamic, using the mechanisms defined in earlier chapters.

The main challenge of an infrastructure such as the TraSer supported one is how to define the web services and how to make them interoperable if the application areas cannot be foreseen at the time the infrastructure is created. The services must constantly adapt to new requirements.

In the TraSer concept the information on each item is held by a server that is identified by a part of the item identifier. Item identifiers are of the format ID@URI. The URI is the identifier of the set of web services where the item information can be stored and retrieved. The ID is unique within the scope of the URI.

Whenever an item passes some checkpoint, a set of attributes are uploaded to the server under the key of the item's ID. The set consists of a number of basic attributes, such as date-time and location, but is extensible. For some items, for instance, the temperature should be uploaded as well. When items are assembled, the ID of the parts and the ID of

the assembly must be related to each other. Note that the information on the assembly may be held by a different server (with a different URI) than the information on the parts.

TraSer web services not only receive uploads and answer queries, they interoperate with each other. For instance, when a product is stored in a container, and the tag of the container is read at some checkpoint, the location of the container must be propagated to the server that holds the information on the product.

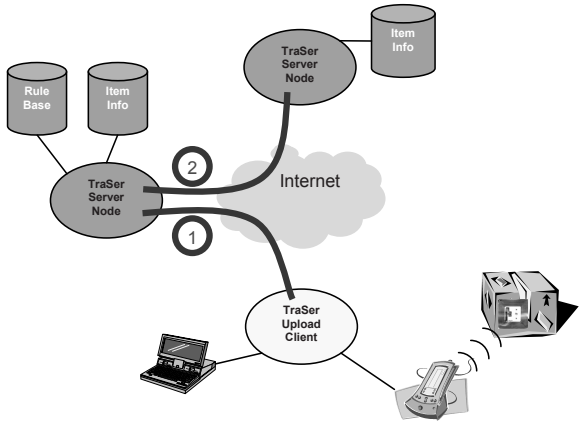


Fig. 12.1 TraSer architecture

When an item is scanned, a basic set of data is uploaded to the server that holds the information on the item. In the TraSer identification mechanism, the URI of the server is encoded, so the uploading application knows where to go. The basic set of information usually includes the location and the date-time of scanning. It should however be possible for the uploading application to add additional data, such as the temperature or the operation that has been performed on the item.

Two types of meta-information may be added: attributes (individual item properties) and associations with other item types. Additionally the item types need to be specialised. The item type determines what attributes and associations an item may possess.

A TraSer server initially supports a basic set of attributes that can be represented in a B2B knowledge base, see table 12.13.

Cases from various sectors

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
1	0	0:00	S	Add	Def	Assert	Item		has	Location			1	1	B,S	Def, Ins, Obs	Assert		
2	0	0:00	S	Add	Def	Assert	Item		has	Measurement			0	1	B,S	Def, Obs	Assert		1
3	0	0:00	S	Add	Def	Assert	Item		has	Association	Item		0	1	B,S	Def, Obs	Assert		1

Table 12.13 Basic TraSer knowledge base

The mechanism to define new attribute types is described in chapter 6. Basically the new attribute type is defined before it is instantiated. Definition of a new attribute type:

- defines a more restrictive data type
- defines the role of the attribute within the namespace of the role of an existing attribute

For instance, if at some location the temperature is relevant and needs to be registered for perishable items, the following is proposed to the server (table 12.14)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
10	2	1:00	S	Propose	Def	Assert	Perishable_ Item		has	Temperature_ Measurement			0	1	B,S	Def, Obs	Assert		1

Table 12.14 Temperature

For each server a client needs to define a specific attribute type only once. The client should remember (maintain a registry) which server supports which attribute type. The registry has the structure of a B2B knowledge base.

Once a new attribute has been accepted by a server, actual registration of items may take place.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
21	1	3:00	B	Add	Ins	Assert	123@flowcanto.com		has	Location	Zwolle								
22	10	3:00	B	Add	Obs	Assert	123@flowcanto.com		has	Temperature	5 CEL								

Table 12.15 Item registration

Much the same way new associations between items may be defined. Suppose it is needed to define a loading/unloading operation of items to/from a means of transport. Means of transport are items as well, identified with a ID@URI.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
30	3	4:00	S	Propose	Def	Assert	Item		load	Transport Means			0	1	B,S	Def, Obs	Assert		1
31	3	4:00	S	Propose	Def	Assert	Item		unload	Transport Means			0	1	B,S	Def, Obs	Assert		1
32	3	4:00	S	Propose	Def	Assert	Item		contains	Transport Means			0	1	B,S	Def, Obs	Assert		1

Table 12.16 Item association

Then, when an item is actually loaded, the information is exchanged according table 12.17.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
40	1	5:00	B	Add	Ins	Assert	123@flowcanto.com		has	Location	Zwolle								
41	30	5:00	B	Add	Obs	Assert	123@flowcanto.com		load	Transport Means	987@Trucking.com								

Table 12.17 Actual item tracking

Upload propagation is not explicitly defined by the client. Whenever a relation is defined between two items, such as assembly, loading or packaging, the server is to propagate each event to the server of the related item. In the case of table 12.17, this means that the

information that Transport Means 987@Trucking.com now contains item 123@flowcanto.com is propagated to the Trucking.com server.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Utterance #	Based on Utterance #	Timestamp	Uttered by Party	Action	Stereotype	With Intention	Source Concept Name	Source Role Name	Verb	Target Role Name	Target Concept Name	Part of ID #	Min Repetition	Max Repetition	To utter by Party	With allowed Stereotype	With allowed Intention	Precondition #	Transaction with #
50	1	6:00	B	Add	Obs	Assert	987@Trucking.com		has	Location	Zwolle								
51	32	6:00	B	Add	Obs	Assert	987@Trucking.com		contains	Item	123@flowcanto.com								

Table 12.18 Actual item tracking

TraSer clients are pre-programmed with the item attributes and relations they support. Supported attributes and relations are closely linked to the types of operation at the site of the client. Trucking operations support different sets of attributes from warehouse operations or production activities.

TraSer servers need to be able to adapt flexibly to the attribute and relation types clients propose. If the server uses an SQL database, definitions may be converted into ADAPT TABLE SQL statements as described in chapter 11.

This example illustrates how tracking and tracing can be established in a complete open environment, starting with a minimum of assumptions. Clients may upload any attribute of any event on any item type to servers they did not contact in advance.

References

[1]. ISO 10303 STEP, International Organisation for Standardization.
[2]. ISO 13584 PLib, International Organisation for Standardization.

13 E-business workstation prototype

Summary

In this chapter an XSLT based, flexible workstation is described, that allows small and medium sized organisations to participate in B2B communication. The workstation as it has been implemented, supports part of the architecture as defined in this thesis.

The workstation software is mainly executed in a browser. The XSLT script lets the XSLT processor transform any XML file into an XHTML file with XForms code, provided that the XML schema that describes the structure of the XML file is available.

An XForms capable browser then presents the XML file as a form on the screen.

The look-and-feel of the form is defined in a simple XML annotation file, that is not specific to the XML file. It interprets the elements in the XML file semantically. It may make use of CCTS annotation in the schema, or of any other semantical annotation.

13.1 Introduction

In chapter 11 is shown how middleware and integrated business information systems may be equipped to support dynamic B2B communication as described in this thesis. Many smaller companies however do not deploy such middleware or systems. Many larger companies do not use middleware either. To prevent a resulting dead-lock in the adoption process, a workstation, merely based on internet standards has been designed, that enables business people to participate in dynamic B2B communication processes right away. The workstation is designed in such a way that parts of the information process can gradually be delegated to business systems and middleware.

The workstation is based on XML that means that utterances are exchanged between business people as XML messages and metadata is represented as XML Schema. In order to define and manipulate semantics, and to manage semantic diversity, the mechanisms as defined in the Core Components specification (CCTS [1]) are used.

One of the mechanisms described in CCTS is the Context mechanism. CCTS defines two layers of building blocks: Core Components and Business Information Entities (BIEs). Aggregate Core Components represent very generic abstract business entities, such as “Party”, “Product” and “Delivery Event”. In a certain context (e.g. the Grocery wholesale business in Holland) these Aggregate Core Components are specialized to become Aggregate Business Information Entities (ABIEs) that may show up in electronic messages. Specialization may be performed in a hierarchical fashion: World – Europe – Holland for instance, or Consumer articles – Food – Groceries. Context dependent specialization rules should be developed by (local) standardization committees, or negotiated among the trading partners involved.

The mechanism proposed for the workstation does not use Context drivers, as suggested by CCTS, but enables trading partners to directly negotiate on message and process meta-information [2]. Such negotiation may follow the mechanisms as described in this thesis. The workstation may support such mechanism, by producing XML Schema's on the fly (this however has not yet been implemented).

Whatever mechanism is deployed, application of CCTS principles will lead to millions of different message schemas. That large set of schemas will constantly be adapted and extended. Businesses will need to implement slightly different sets of schemas for each trading partner. This means that application interfaces cannot be hard wired, but must be flexible. The same is valid for human interfaces, which is the subject of this chapter.

Many EDI systems are only at one side integrated to a back end system. At the other side dedicated client software is used to produce the messages and to show the EDI messages as forms on a PC screen. This is only feasible if the relation between the small, not integrated, and the big integrated company is stable and if the small company does not have too many EDI customers, because for each different message type the client software must be programmatically adapted.

With XML technology, browsers can show messages directly as a screen form by means of a stylesheet, another XML construct. The major obstacle however is that whenever a message schema changes, the stylesheet must be adapted as well. Stylesheets are written in XSLT [3], which is a powerful, but complex, programming language. Local customization of schemas and forms are not possible without reprogramming. In fact not only content of XML messages must be shown, but fill-in forms must be presented to the user for creating messages. This further complicates stylesheet maintenance.

Moreover, form behaviour and lay-out is very much depending on the local business process and on individual user preferences. In fact trading partners should completely be independent from each other in shaping their user interfaces. Tight coupling between schema and stylesheet violates this requirement.

13.2 Related work

To extend the ebXML applicability to small companies and underdeveloped countries, the UN started a project, called eDocs [4]. The eDocs project is to produce a set of electronic document schemas to be used in international trade, together with the Stylesheets that allow companies to print the standardized electronic documents or to show them as electronic forms in a browser. Stylesheets however are programmed one-to-one based on the message schemas. It is to be expected that when the eDocs library grows and specific documents are to support specific local regulations, maintenance of the stylesheets will become too burdensome.

Similar projects to produce standard stylesheets with standardized message schemas have been started in many governmental and business environments. Crane Softwrights Ltd [5] has produced a set of stylesheets to show OASIS UBL messages on screen or to print

them as PDF documents. Bals [6] proposes to use XSL-FO to present UBL messages. In Holland, GS1 has introduced an architecture with XML-stylesheets to show web-forms in the electro technical and plumbing sector [7]. In Holland and Denmark governmental bodies design stylesheets for forms to support electronic government services. In these projects advanced tools are used to produce the XSLT code [8], but again a produced stylesheet fits one schema and a schema leads to one stylesheet. Local preferences that would lead to different form lay-out or behaviour for the same schema are not supported. In Italy the Arianna project [9] does not take message schema as a basis, but a harmonized data model. From the model schemas and stylesheets are generated. The project generates e-government solutions for local Italian municipalities and other autonomous governmental agencies. Again, although programming burden is released, stylesheets cannot be adapted to user needs, but are imposed on the user by the governmental service.

In the Efficient project [10] a similar architecture is employed, but for B2B rather than for governmental processes. The generation of forms within the Efficient project is however only used for demonstration and prototyping, not for run time use. Bizdex [11] and Govdex in Australia are projects to facilitate structured communication among companies and with Governmental services, respectively. The projects concentrate on the architecture for registering and employing models, rather than on the user interfaces and their flexibility.

None of these initiatives attempted to support B2B (Business-to Business) or B2G (Business-to-Government) communication with flexible user interfaces that can independently be managed by one of the peers and that is tolerant for (small) changes in the message schemas.

13.3 Web-form solutions

For a flexible web forms solution the following requirements can be stated:
1. The form should adapt itself dynamically to the XML Schema
2. The form behavior should be set for information elements that may be used across schemas.

HTML includes a set of form controls. This set however induces a few problems [12]. First of all, the behaviour of the controls is governed by scripts, which makes the HTML code complex and hard to maintain. Second, initialization is different per control, so the processing of default or standard data is complex. Third, HTML forms do not have a deep hierarchical structure, which is common in B2B or B2G documents. Fourth, HTML forms rely on certain processing at the server side. In a peer-to-peer environment the only communication with the server is by means of the XML message as defined in the schema. No ‘form related’ processing may be expected from a trading partner.

W3C has published another specification for forms, called XForms [13]. XForms is a set of controls and mechanisms to bind the controls to elements in an XML message. XForms has a number of advantages over HTML forms. An XForms document (which is

represented in XHTML) defines the function of the form controls, not their actual appearance. This means that the same XForms document could be sent to a browser, a mobile phone or even a voice response system. The controls would be presented differently, but behave similarly.

XForms makes native use of XML and of the XML schema, which makes the binding to the message to be sent straightforward. XForms can completely be processed at the client side, although tools exist to translate an XForms document in a webserver to plain HTML which then is presented to the browser.

Unfortunately XForms is not (yet) supported by all browsers, notably not by Microsoft Internet Exporer (IE). On the market and in Open Source however a number of 'plug-ins' are offered to enable IE users to use XForms [14][15].

For users in small and medium sized companies (and in larger companies that have not yet enabled their ERP system for electronic messaging), it is of utmost importance that forms that they may use daily are easy to handle. Data that is not changing should be pre-filled and trivial data (like the system date) should even not be shown as it only clutters the screen. All or most forms within the company should be styled in a similar way. Behaviour of controls, the use of colours, tab-sequences, paging, etc. must be ergonomic. In addition to user friendly presentation, many companies require business rules and integrity checks to be built in into the form controls, such as maximum amounts, calculations, sub-setting of code lists, etc. These are private rules, not constraints that (may) appear in the schema.

13.4 Proposed architecture

In order to fulfill the two requirements in section 13.3 we developed a generic XSLT script that dynamically combines the structural message definitions in the XML Schema with annotations for the form behavior per information element that may occur in the schema. Those annotations are private to the form user and are contained in an XML file.

XSLT is a language, supported by most browsers, to transform an XML message into another XML message, HTML or any other structure. As an XForms construct is an XML (XHTML) message, it should be possible to transform any XML message into such construct. An XML schema (XSD) is also an XML message. So any XML schema, if it conforms to some more specific rules, can be converted into a screen form by means of XSLT. The more specific rules are offered by the UN/CEFACT XML Naming and Design Rules (NDR) [16] and CCTS [1].

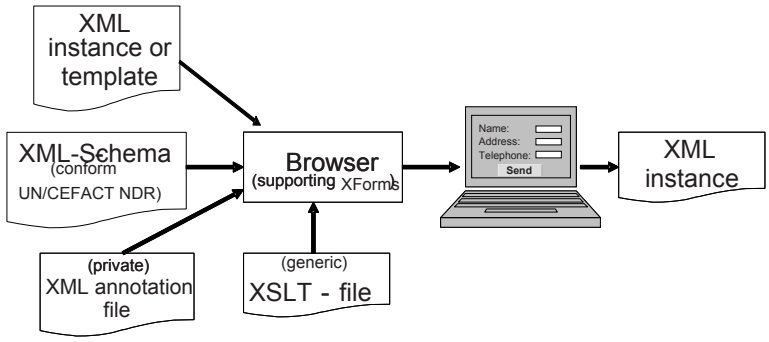


Figure 13.1 B2B Workstation

To prove the concept to transform an XSD into an XForms XHTML file, XSLT code was developed that does this job. It must be stressed that one generic XSLT file was constructed that converts *any* (NDR compliant) XSD into an XForms form. Doing so, concepts developed by Gropp [17], Brazier [18] and Garvey c.s. [19] were gratefully used.

As noted, the user friendliness of screen forms is important. Automatic conversion of an XML schema into a form is expected to lack friendliness. So it is necessary to introduce a mechanism for users to enhance the presentation features of the form. On the other hand, one cannot expect users to redesign a form, each time when a new (or slightly modified) XML message structure is agreed with a trading partner. The solution we propose is to make use of the hierarchical nature of CCTS Business Information Entities (BIEs), which are represented in the schema.

The user (or a super-user or systems manager) should maintain a simple XML file in which the presentation options are annotated of Business Information Entities that may appear in messages. Presentation options may include the look-and-feel of form controls, but also text of captions, default values, code enumerations and integrity checks. Whenever a more specialized Business Information Entity is agreed or imposed in a new message or a message to/from a new trading partner, the more specialized BIE inherits the presentation options of its parent. Users may then adapt the presentation of the new BIE by adding those options to the annotation file using some (e.g. WYSIWYG) editing tool.

For the proof of concept the structure of the annotation file has been kept as simple and straightforward as possible (see figure 13.2). Many studies have given suggestions how to specify user interfaces in XML messages, e.g. USIXML [20] or UIML [21]. It is very feasible to use one of those proposed structures as the definition of the form presentation. The only prerequisite is that the Dictionary Entry Names of the Business Information Entities are used as a key to the presentation definitions.

```
<core-component den="Contact. Electronic Mail. Text">
        <element class="core-component">
                <width value="24" />
        </element>
        <element class="content">
                <width value= "12" />
        </element>
        <element class="caption">
                <width value= "4" />
                <show value="true" />
                <text value="E-mail" />
        </element>
</core-component>
<core-component den="Invoice. Payment. Payment Means">
        <element class="core-component">
                <sequence value="09" />
                <class value="ABIE" />
                <width value="24" />
        </element>
        <element class="caption">
                <size value="big" />
                <text value="Betaling" />
                <class value="caption-big-left" />
        </element>
</core-component>
```

Figure 13.2 Annotation file snippet

Presentation is controlled conform the specialization hierarchy of Business Information Entities. Some Association Business Information Entity (ASBIEs, associations between or roles of ABIEs) could for example be named `Fresh_ Food_ Product. Urban_ Reseller_ Customer. Franchise_ Retail_ Party` (see figure 13.3 for an illustrative UML Class diagram of this case). The form presentation engine, as coded in the XSLT file, first searches the annotation file if presentation annotations exist for this ASBIE. If for some features (e.g. caption, tab-sequence) no annotation is found, the engine searches for annotations for the ABIE `Franchise_ Retail_ Party. Details`. Note that the ASBIE is the specific use or role of the (child) ABIE within the `Fresh_ Food_ Product` (parent) ABIE. Features that are not found in the ABIE are then looked up in `Food_ Product. Reseller_ Customer. Retail_ Party`, which is the ASBIE `Fresh_ Food_ Product. Urban_ Reseller_ Customer. Franchise_ Retail_ Party` is based on. Still missing features are taken from respectively the ABIE `Retail_ Party. Details`, the ASBIE `Product. Customer. Party` and the ABIE `Party. Details`. If by then still presentation features have not been defined, defaults that were built in the XSLT are used to lay-out the form.
In figure 13.3 the defined presentation hierarchy is illustrated in a UML Class diagram. The main advantage of this approach is that whenever a more generic form has been designed (say, including information of a `Retail_ Party`), a new, more specialized form (which includes `Franchise_ Retail_ Party` information) inherits such design. Only if the new form needs enhancement (e.g. a new caption for `Franchise_ Retail_ Party`), the user or his system manager may change that particular caption on that level. No other presentation options need to be redefined

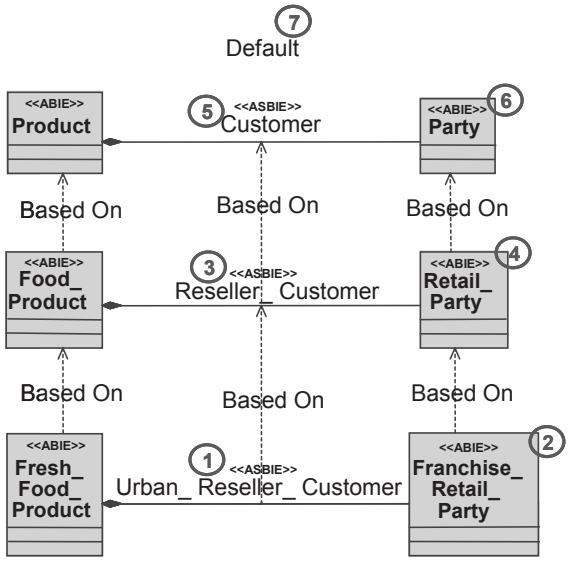


Figure 13.3 Presentation hierarchy

Some presentation options depend on the form document itself rather than on the BIE hierarchy. Colours, character fonts and border styles for example need to be consistent throughout the form. Therefore these features have been split: the feature function (e.g. 'normal', 'emphasis', 'alert') is defined at BIE level, the actual styling (e.g. 'alert colour' = 'red') is defined at document level.

13.5 Transformation file

The XHTML file that is shown by the browser as a screen form has a structure as in figure 13.4.

XForms structures consist of three parts:

- XForms Instances, that contain a template of the XML message that must be sent
- XForms Bindings, that bind the contents of the message elements to the form controls
- XForms GUI Elements, that define the form structure and the form controls.

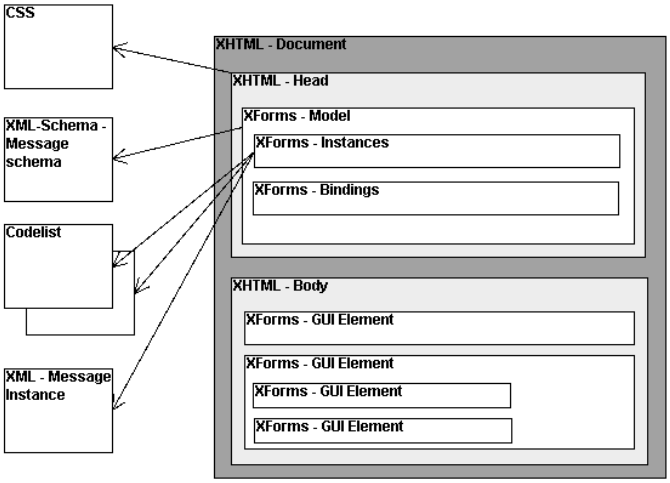


Figure 13.4 XForms XHTML structure

The structure of the XML instance message is defined by the same XML Schema that is input to the transformation process. The schema may refer to external code lists. The form is styled by means of a Cascading Stylesheet (CSS). The CSS may be generated by the transformation process by matching the Business Information Entities as defined in the Schema to the style definitions in the annotation file by means of the following XSLT code:

```
<xsl:template match="xsd:element" mode="style">
      <xsl:call-template name="generate-style" >
            <xsl:with-param name="element.root" select="@name" />
            <xsl:with-param name="element.root.type" select="@type" />
      </xsl:call-template>
</xsl:template>
```

Figure 13.5 Template to generate CSS-file

A UN/CEFACT NDR [16] compliant schema has a modular structure. ABIEs and data-types are globally declared. Each BIE or data-type definition has an annotation with all metadata of the element, including its Dictionary Entry Name. The transformation process uses this metadata to generate the right form controls and parameters. For example, the defined Business term is used as a default caption.

```
<xsd:element name="BasePrice" type="qdt:EuroAmountType">
      <xsd:annotation>
            <xsd:documentation xml:lang="en">
                  <ccts:UniqueID>UN00000012</ccts:UniqueID>
                  <ccts:Acronym>BBIE</ccts:Acronym>
                  <ccts:DictionaryEntryName>
                      Consumables_ Order Item. Base Price. Amount
                  </ccts:DictionaryEntryName>
                  <ccts:BusinessTermName>
                      Basisprijs
                  </ccts:BusinessTermName>
                  ...
            </xsd:documentation>
      </xsd:annotation>
</xsd:element>
```

Figure 13.6 Schema structure

CCTS defines ten basic Core Component Types or CCTs like "Amount", "Text", "Measurement", etc. Data-types are derived from these CCTs by restricting the values of the CCT components. CCTs are complex structures: the Amount CCT for example includes the Currency Code. For each CCT a separate transformation module is included in the XSLT, that is activated when a data-type is met which is based on that CCT.
The modular structure of the XSLT makes it easy to extend the functionality by adding more presentation features, like paging, calculations, control behaviour, etc. The proof of concept that has been produced to date only contains the most elementary features.

13.6 Meta data negotiation

The prototype described in the previous sections is using a predefined schema as input. In order to dynamically develop a B2B relationship the workstation should also be capable of manipulating the XML schema, using the mechanisms as described in this thesis. An XML schema is itself an XML structure that can be transformed by means of XSLT into XForms as well as an XML instance can. So in principle the form tool can also be used for manipulating XML schemas. In this section is described how the tool fits in the architecture as described in chapters 5 and 6.

The utterances implied by XML instances and schemas received from a business partner should be added to a B2B knowledge base. The knowledge base can be structured as an XML structure. See figure 13.7. The extraction of the utterances, their validation and inclusion in the knowledge base can be defined by an XSLT script. A standard XSLT processor then can do the job of extraction, validation and inclusion.

```
<Utterance number="10" timestamp="20110804T065200" basedOn="5">
      <UtteredBy>Seller</UtteredBy>
      <Action>Add</Action>
      <Stereotype>Definition</Stereotype>
      <Intention>Assert</Intention>
      <Source>Invoice</Source>
      <Verb>Claim</Verb>
      <TargetRole>VAT</TargetRole>
      <Target>Amount</Target>
      <MinRep>1</MinRep>
      <MaxRep>1</MaxRep>
      <ToUtterBy>Seller</ToUtterBy>
      <TransactionWith>1</TransactionWith>
</Utterance>
```

Figure 13.7 B2B knowledge base

Another XSLT script can extract the allowed utterances a user may send back to the trading partner. Those can be presented to the user in a menu. Each transaction (usually a number of utterances are bundled in a transaction) can be shown to the user as a form. The filled-out form is added to the knowledge base and sent to the trading partner.

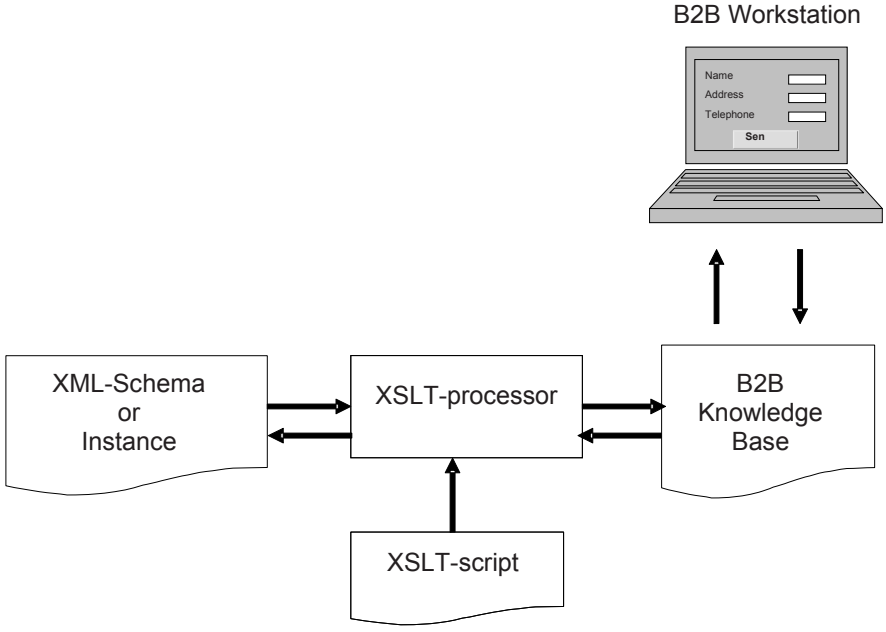


Figure 13.8 Workstation architecture

The above is true for instances, but also for schemas. A trading partner may send a new XML schema, with definitions of new concepts. Of course the (meta-)attributes as defined for lines in a B2B knowledge base must be present in the schema. Natively only

few of those attributes have been defined in XML schema. The other attributes are added by means of the `<Annotation><Documentation>` schema construct.

Thus small and medium sized companies with an XForms workstation can be a participant in flexible B2B processes, just like large companies with major ERP systems and middleware can be.

13.7 Conclusion and future research

By introducing flexible, form based B2B communication, the threshold for SMEs to connect to (ERP-) systems of larger trading partners (or to each other in exactly the same way) is lowered dramatically. The presented architecture and methodology is based on open standards and can be deployed for this purpose. Its concept has been proved. It has been implemented in pilots in the facility maintenance sector and in the fast moving consumer goods sector. Form-based B2B communication may also be used as a first implementation phase for larger companies, decoupling the application integration projects of trading partners. The methodology therefore may give a boost to the adoption of structured B2B and B2G business communication.

References

[1]. UN/CEFACT: Core Components Technical Specification – Part 8 of the ebXML Framework. Version 2.01. Adopted by ISO as ISO 15000-5. (November 2003)
[2]. Krukkert, D.: Matching of ebXML Business Processes; Report Project number IST 2001-28548 OpenXchange
[3]. W3C: XSL Transformations (XSLT) Version 1.0, W3C Recommendation 16 November 1999, www.w3.org/TR/REC-xml/, Last accessed 30 Oct 2006
[4]. www.unece.org/etrades/unedocs/, Last accessed 30 Oct 2006
[5]. UBL/Crane stylesheets: www.cranesoftwrights.com/resources/ublss/ (accessed on 30 Oct 2006)
[6]. Bals, K.: Using XSL, XForms and UBL together to create complex forms with visual fidelity, XML2005 Conference Proceedings, www.idealliance.org/proceedings/xml05/ship/1/ 04-02-01-new.PDF (accessed on 17 Oct 2006)
[7]. www.instalnet.nl (accessed on 30 Oct 2006)
[8]. CapGemini: De e-formulierenmachine in de Nederlandse context (in Dutch), www.e-overheid.nl/data/files/eFormulieren/eindrapportformulierenmachine200504.pdf (accessed on 30 Oct 2006)
[9]. Barone c.s.: Semantic of eGovernment Processes: a Formal Approach to Service Definition, I-ESA 2006 proceedings, Springer Verlag.
[10]. Eshuis, R.: c.s. Animating ebXML Transactions with a Workflow Engine (Efficient), Lecture Notes in Computer Science, Volume 2888/2003, Springer Verlag.
[11]. www.bizdex.com.au/ (accessed on 30 Oct 2006)
[12]. Dubinko, M: XForms Essentials, O'Reilly, 2003
[13]. www.w3.org/MarkUp/Forms/
[14]. BackplaneBX
[15]. XSLTForms
[16]. UN/CEFACT: XML Naming and Design Rules for Core Components. To be downloaded from www.disa.org/cefact-groups/atg/downloads/index.cfm. Last accessed 30 Oct 2005
[17]. Gropp, E.: Transforming XML Schemas, XML.COM, 2003, www.xml.com/lpt/a/1092 (accessed on 17 Oct 2006).
[18]. Brazier, D.: Stylesheet for NHSiS Schemas, private mail to H. S. Thompson, www.stylusstudio.com/xmldev/200012/post30080.html (accessed on 17 Oct 2006).
[19]. Garvey, P.: c.s. Generating User Interfaces from Composite Schemas, XML2003 Conference Proceedings, www.idealliance.org/papers/dx_xml03/papers/03-03-04/03-03-04.pdf (accessed on 17 Oct 2006)
[20]. Limbourg, Q.: c.s.: USIXML: A Language Supporting Multi-path Development of User Interfaces. EHCI/DS-VIS 2004: 200-220
[21]. UIML: www.oasis-open.org/committees/download.php/5937/uiml-core-3.1-draft-01-20040311.pdf (accessed on 30 Oct 2006)

14 Discussion and recommendations

Summary

In this chapter we assess the architecture that was designed against the requirements in chapter 3. The thesis is concluded with some recommendations for implementing the architecture.

14.1 Validation

In chapter 4 a number of issues were mentioned that lead to requirements to the architecture. In this section the architecture is assessed against these requirements.

Req1: Part of the architecture must be a protocol
The architecture concentrates on negotiating a protocol and is based upon business conversation analysis. However, as B2B communication involves private information systems, the way such (legacy) systems may be part of the architecture and participate in the conversation is also addressed. The architecture does not involve a centralized system that interconnects business partners, but interconnects the partners' systems. The structure of utterances and the rules governing the contents of the utterances is defined as a protocol.

Req2: The architecture must support negotiation of process flows and data structures
Negotiating the process and the data that is exchanged is an important aspect of the architecture. The negotiation may be carried out at runtime and is directly supported by the systems that form part of the architecture. Introduction of new concepts, attributes of concepts and relations between concepts may be negotiated, as process choreography can.

Req3: It must be possible to exchange intentions, not only facts
The utterances that are exchanged are in fact speech acts, with subjective values and private intentions. Processes may be take different viewpoints on the same 'facts' into account. Each utterance contains an intention.

Req4: The architecture must support adaptation of the process choreography
As negotiation can be performed at runtime, process negotiation can be reopened at any moment during the business relationship. So changing trust levels and control mechanisms may be supported by new process flows. During execution of the process, alteration of the choreography may be proposed (and accepted).

Req5: The architecture must support adaptation of the data structures
Process and data are integrated. Process steps are defined by pre- and post-conditions expressed in data definitions. The negotiation in fact defines the data and with it the process. During execution of the process, new concepts or new properties of concepts may be proposed (and accepted).

Req6: The architecture must be independent from the technological implementation
The negotiation mechanism described is completely technology independent. It may be implemented using legacy technology such as UN/EDIFACT and X.400, up to date technology such as XML and Internet or any technology to develop in the future. The architecture is conceptual; it needs to be mapped onto a technology. Mapping may be performed on legacy databases and on existing message libraries.

Req7: The architecture must not be specific for a certain business environment
The protocol is independent from a specific business sector and supports development of a large variety of practices, even bilateral specialization. The basic ontology proposed is focused on (generic) business processes, but the mechanism allows the manipulation of any ontology (e.g. statistics, government procedures or technological modelling). The architecture completely abstracts from the business context; each business is free to define its own ontology.

Req8: The architecture should contain instruments to enable enforcement
Commitments are clearly defined by the explicit use of speech act intentions. The proposed ontology is based on the fulfilment of commitments and obligations.

Req9: The architecture must not be specific for a geographic area
With the absence of central components the architecture is completely scalable. It is not limited to some (natural) language; the abstract protocol can be mapped to any local language. The architecture is not bound to any geographic area.

Req10: The architecture must be capable to use various technologies concurrently
The core of the architecture is a (virtual, distributed) knowledge base that can be connected via multiple channels concurrently. The mechanisms abstract from technology, they may be implemented in any technology.

Req11: The architecture should not assume prior agreement between trading partners
The initial ontology a business conversation is based on may be as small as a single entity ("Entity"), may be the initial ontology presented in chapter 9 or may be some industry specific ontology both partners understand. In any case there is no need for business partners to negotiate off-line on the business process or semantics. All negotiations may be supported by the B2B system. This is illustrated in section 11.9.

Req12: The architecture should be capable to include legacy applications
The B2B architecture as described in this thesis was designed to form the interface between business information systems. The mechanisms to negotiate meta-data and process flows are formally defined in a meta-model. Business systems, possibly assisted by middleware and on line services, are enabled to establish a connection with each other. In chapter 8 was shown that the knowledge base structure may be mapped on traditional modelling and exchange languages.

Req13: The architecture should be capable to be implemented in a legacy environment
As the architecture does not require a specific technology, it may be implemented in an environment with (rigid) legacy applications. The dynamics may be introduced by directing some messages to users, with the method described in chapter 13.

14.2 Conclusion

The paradigm shift as proposed in this thesis must make it to the minds of information managers and business users. Wide spread introduction of integrated information systems to support intra organizational processes took many years and their adoption lagged far behind the technological possibilities. The notion that computers and networks can be used to exchange structured information with all trading partners (not only the regular, trusted ones) will need a mind shift. Mind shifts cost time. The concept must be demonstrated at smaller scales before it can be introduced widely. Therefore the products and services that support it must facilitate a gradual transition. So they must be connected to traditional, rigid, EDI and XML systems as well as to dynamic and flexible B2B systems. A solution in which a ubiquitously used HTML browser is sufficient to connect to such B2B systems would speed up the adoption. In chapter 13 such browser based B2B workstation is described.

The mechanism as described in chapter 6 should be standardized. It is not necessary to immediately support all features. Intentions, for instance, may be treated as specializations of objects as it is the case today in many standards (e.g., a quotation then is a different business object as an order, instead being treated as an 'offered order').

Most important is that the mechanism should be proven in a pilot implementation.

More research (and consensus within the business community) is needed with regards to the initial business ontology. As stated in chapter 9, we propose to take REA as a basis. Many concepts have yet to be defined as specializations of REA concepts, and in the process, taxonomies of verbs and roles need to be developed. The definition of verbs and roles in this thesis is only used to illustrate the concept of building a business ontology by specializing existing concepts. To build such an ontology for a business domain must be the subject of another study. It is crucial, though. If definitions of concepts are not being formalized somehow, automating B2B systems on the meta level is not possible.

The best way to proof the concept, is to implement the architecture, as proposed in this thesis, in a service that offers instant interoperability to its subscribers. The service should build up and maintain a semantic registry, so concept definitions may be reused across business relationships.

ANNEX Meta-model of the knowledge base

In section 6.2 the B2B knowledge base was structured in a tabular format. That format is suitable for introducing the concepts and for illustrating the mechanisms that allow business partners to initiate and control their business relationship. In this section the structure of the knowledge base is further formalised.

The formalisation is done by means of constructing a meta-model. The meta-model is expressed in ORM. ORM forms the bridge between ontology languages, formal verbalization and data modelling. In fact an ORM model can be constructed by uttering statements in a (semi-) structured language, much as the B2B knowledge base is populated by means of utterances.

In a B2B relationship business partners must be capable of introducing new concepts or entity types. That means that in their conversation they manipulate the model. Manipulating models (that reside at MOF M1) means that a meta-model (residing at MOF M2) is populated.

To map the contents of a B2B knowledge base on ORM the description of the B2B knowledge base must be structured as an ORM profile, which restricts the set of ORM artefacts and operations. E.g. naming rules are defined that restrict the freedom how to name concepts and roles.

The business conversation results in the population of the meta-model (M2), adding new concepts and relations, and in population of the model (M1), adding and mutating instances. So the M1 model is changed during the operational process. In the B2B systems as described in this thesis, business partners build a model during their conversation. The model is immediately mapped to their (legacy) systems using middleware tools. Design time and run time are collapsed and cannot be separated.

ORM, like most modelling languages, depart from a different paradigm, namely the paradigm that modellers and domain expert together manually design the model, that later is compiled into a system. In that paradigm design time and run time are separated.

A B2B knowledge base represents the conversation between two trading partners. As has been shown in section 5.6 a knowledge base consists of utterances that are representations of speech acts. Utterances may be numbered consecutively. Trading partners are identified by means of some identification scheme.

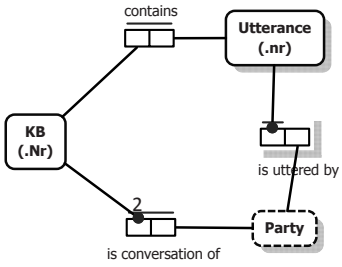


Figure A.1 Knowledge Base

Verbalized:
A B2B knowledge base represents the conversation between exactly two parties
A B2B knowledge base contains zero or more utterances
An utterance is uttered by a party

Each utterance is an assertion about an event in the real world or a decision or inference. Such assertion assigns a property to a concept or entity. As has been shown in section 5.7 an individual entity can be regarded as a concept with one instance, so at meta-level only the 'concept' artefact is needed. A property associates a concept with another concept or entity. The other concept is called the 'target' of the property.

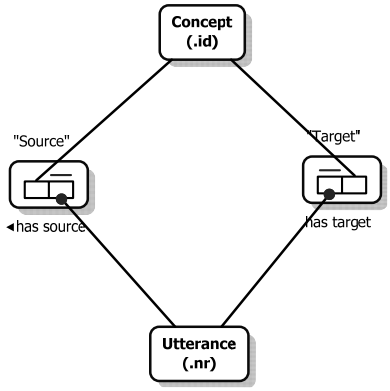


Figure A.2 Core utterance

Verbalized:
Each utterance refers to a source concept
Each utterance refers to a target concept

The property, an utterance is asserting, assigns roles to the source and target concepts, and contains a verb that qualifies the relation between the roles.

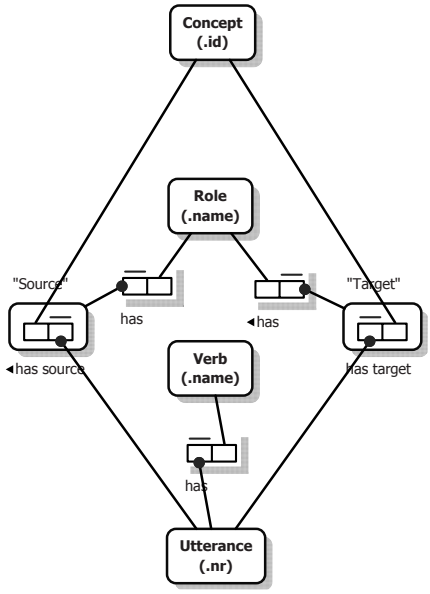


Figure A.3 Verb and roles

Verbalized:
Each Utterance contains a Verb
Each Source concept has a Role
Each Target concept has a Role

Each Utterance is based on a previously uttered utterance.

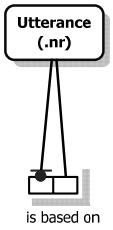


Figure A.4 Based on

Verbalized:
Each Utterance is based on another utterance.

Concepts, Roles and Verbs are also based on previously introduced Concepts, Roles and Verbs.

A party utters an utterance with an intention. An utterance is of a stereotype, that defines whether the utterance is a definition, .expansion, restriction, perception, state, instantiation or observation.

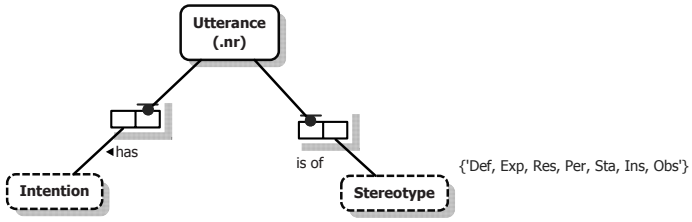


Figure A.5 Intention and stereotype

Verbalized:
Each Utterance has an Intention
Each Utterance is of a Stereotype

In an Utterance it can be controlled which intentions utterances may have that are based on the utterance (its subtypes). It can also be controlled which party may utter utterances that are based on the utterance and of which stereotypes those utterances may be. This can be modeled by constraints.

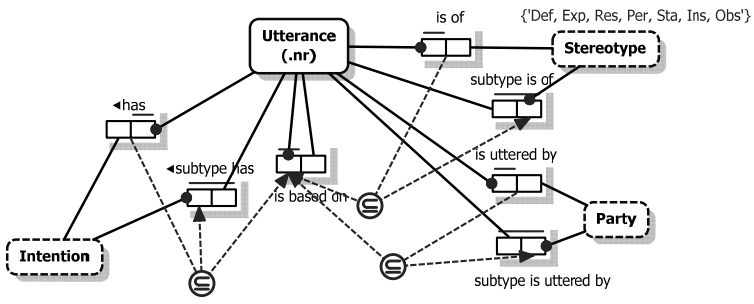


Figure A.6 Subtype intentions, parties and stereotypes

The constraints define that, e.g., the intention of an utterance, must be a subset of the intentions that are defined as allowable intentions of the subtypes of the utterance, the utterance is based on.

Concepts and properties are named. The names form namespaces for concepts and properties that are based on the named concepts and properties. Events, decisions and inferences typically result in more than one utterance. The utterances that result from an event belong together and form transactions.

Meta-model of the knowledge base

The complete meta model of an utterance is shown in figure A.7.

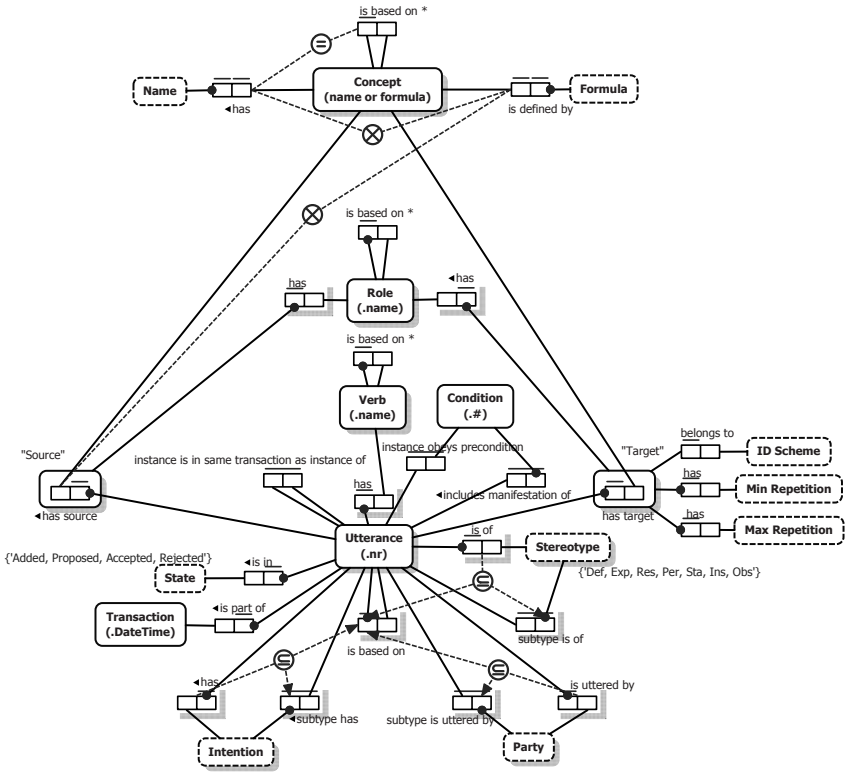


Figure A.7 Utterance meta model