Dottorato di Ricerca in Informatica
Università di Bologna, Padova
INF/01 INFORMATICA

An ontology-based approach to define and manage B2B interoperability

Nicola Gessa

7th March 2007

Coordinatore:	Tutore:
Prof. Özalp Babaoğlu	Prof. Paolo Ciancarini

Abstract

Electronic business surely represents the new development perspective for world-wide trade. Together with the idea of ebusiness, and the exigency to exchange business messages between trading partners, the concept of business-to- business (B2B) integration arouse. B2B integration is becoming necessary to allow partners to communicate and exchange business documents, like catalogues, purchase orders, reports and invoices, overcoming architectural, applicative, and semantic differences, according to the business processes implemented by each enterprise. Business relationships can be very heterogeneous, and consequently there are various ways to integrate enterprises with each other. Moreover nowadays not only large enterprises, but also the small- and medium- enterprises are moving towards ebusiness: more than two-thirds of Small and Medium Enterprises (SMEs) use the Internet as a business tool. One of the business areas which is actively facing the interoperability problem is that related with the supply chain management.

In order to really allow the SMEs to improve their business and to fully exploit ICT technologies in their business transactions, there are three main players that must be considered and joined: the new emerging ICT technologies, the scenario and the requirements of the enterprises and the world of standards and standardisation bodies.

This thesis presents the definition and the development of an interoperability framework (and the bounded standardisation intiatives) to provide the Textile/Clothing sector with a shared set of business documents and protocols for electronic transactions. Considering also some limitations, the thesis proposes a ontology-based approach to improve the functionalities of the developed framework and, exploiting the technologies of the semantic web, to improve the standardisation life-cycle, intended as the development, dissemination and adoption of B2B protocols for specific business domain. The use of ontologies allows the semantic modellisation of knowledge domains, upon which it is possible to develop a set of components for a better management of B2B protocols, and to ease their comprehension and adoption for the target users.

Contents

List of Figures

Chapter 1

Introduction

This thesis outlines a novel approach and an ontology-based architecture to manage Business-to-Business protocols. The aim is to show that Semantic Web technologies and the use of ontologies can be successful adopted to enhance the efficiency of B2B platforms. This architecture improves the B2B protocol management both for the protocol developers (and then for standardisation bodies) and for the target users, exploiting an ontological description of the protocol itself. In this chapter, first of all I will introduce the reader to the scenario of E-commerce and E-business and provide some basic definitions; in particular I will raise some general considerations about the supply-chain management, highlighting some issues to solve and goals to achieve in this context. Afterward, starting from the structure of the thesis, I will focus this introduction on my contribution and I will briefly show the structure of the proposed architecture.

1.1 B2B scenarios

1.1.1 E-commerce and E-business

The development of the relationships among various commercial realities has often represented one of the most important starting points for the progress of the human activities. New commercial relationships involve new surprising products, new providential collaborations, new unexpected possibilities for the growth of the communities, new ideas, new materials, and so on. The history is full of innovative solutions and of risky businesses: considering the last decades, several initiatives have been undertaken as soon ICT (Information and Communication Technology) has been introduced. Information technology has in fact given us a new perspective about the possibilities of development in almost every field of the commerce: starting from simple archives of data registering information and tracking commercial operations, to the adoption of complex real-time transactions among different enterprises, there is now an enormous variety of services, facilities and communities that have in some manner revolutionized the way to do commerce and business, leading enterprises, industries, agencies and organisations to deal with a new dimension of the market, the e-commerce and e-business dimension.

A short glance over the history of the IT shows us that one of the turning points in this process was probably the large spreading of the Internet, that becomes every year more and more reliable and efficient. The new electronic market is characterized by the lack of boundaries: it is potentially widespread all over each country, reaching almost whoever wants to be reached.

A possible (and wide-accepted) definition for E-commerce is: "the buying and selling of goods and services on the Internet, especially the World Wide Web". In practice, the term 'E-commerce' is often used in an interchangeable manner with E-business, but this is not correct. E-business (electronic business) can be defined, in its simplest form, as the conduct of businesses on the Internet. This widespread definition is surely more general than the E-commerce definition: E-business includes in fact not only buying and selling but also servicing customers and collaborating with business partners. Another definition could be retrieved for example in Wikipedia [125]: "Electronic business refers to any information system or application that empower business processes. Today this is mostly done with web technologies".

Together with the idea of the E-business, and the exigency to exchange business messages between trading partners, the concept of Business-to- Business (B2B) interoperability arouse. B2B interoperability is becoming necessary to allow partners to communicate and exchange business documents (as catalogues, purchase orders, reports and invoices), overcoming architectural, applicative, and semantic differences, according to the business processes implemented by each enterprise. In order to be really useful, a B2B interoperability solution must assess several aspect like security, reliability, availability, compatibility and consistency.

Business relationships can be very heterogeneous, and consequently there are various ways to integrate enterprises with each other. Moreover nowadays not only large enterprises, but also the Small and Medium Enterprises (SME) are moving towards E-business: more than two-thirds of SME use the Internet as a business tool [43]. The mid-sized enterprises have already closed the gap with large enterprises and the small ones have now the possibility to really exploit the to enhance their production processes. Any way, much work has still to be done, especially in those sectors dominated by (SME): B2B interoperability technology must solve and recompose many issues.

1.1.2 Supply chain management

One of the main context in which the interoperability problem is actively faced is that related with the supply chain management. In [101] supply chain management is defined as ”the combination of art and science that goes into improving the way a company can find the raw components it needs to make a product or service, manufactures that product or service and delivers it to customers”.

The basic steps in SCM are: design a strategy to manage all the resources, choose the suppliers to have the goods and services needed, make the products, deliver them to the customers, and finally receive feedback about problem and defects of the products. The reality of supply chain is very heterogeneous: depending on the enterprise, the same concept of supply chain can be very different.

In the new era of the Web and considering the new communication channels, many organisations must consider to deconstruct their conventional supply chains and to build an interactive E-business network to gain flexibility [97]. Surely E-business impacted supply chain management: [74] focuses on the role of E-business in supply chain integration. This paper considers four aspects to evaluate this impact: information integration is one of them, together with synchronized planning, workflow coordination and a new business model. These parameters represent also four subsequent steps to improve the interoperability of an enterprise system with external partners in a supply chain.

But what can we expect from a solution and architecture for supply chain management and integration? Which are the goals of a truly integrated supply chain? In some cases it is not clear which are the advantages in implementing and using software for SCM. At the beginning the companies viewed SCM software as part of their business architecture that would bring to ”cost saving” advantages. Actually proper platform for SCM can not only streamline production processes, but also create value for the enterprises. [64] discusses the meaning of ”value” of supply chain software: this paper highlights that software creates value when it brings to ROI (Return Of Investment), and examines how applications can create value for the supply chain. ROI occurs when the investment returns exceed the cost of the capital, but the value of supply chain software could vary depending on the perspective. [66] asserts that an integrated supply chain does more than reduce cost, but also creates value for the enterprises, the partners and the shareholders. In [98] the real leverage of a lean supply chain is in creating capacity of growth. This paper argues that the companies must realise that instead of expense reduction or profit enhancement, the main benefits of architectures and software for SCM consist in a new capacity to match with customer demands. [83] argues that research in supply chain management must accounts the context of the industrial society and considers some ethical, political and economic implications.

Future supply chain integration will surely benefit from development in ICT, in particular exploiting the Web as the communication mean. But other research fields will be crucial: the Semantic web, Web services and workflow management.

Business process integration among different and independent enterprises that co-operate along the supply chain is then considered a strategic issue for the industry, to have a more flexible and responsive way to interoperate with their partners. The basis of this integration is the definition of a common B2B (Business-to-Business) framework; in a message-based approach, it consists of a set of templates of messages to be exchanged (content layer), transport and security protocols (transport layer) and collaboration models (business layer)[11].

The messages are usually defined by standardisation bodies after a well-defined standardisation process; EDIFACT [60], in the past, and XML, more recently, are the reference technologies to exchange electronic documents, exploited by diverse standardisation processes which still present two main problems:

1. the business documents are complex to define, adopt and maintain, and standardisation life-cycle could be very complex. In [105] the need for standardisation life-cycle extensions is highlighted.

2. the standardisation processes related to B2B and ICT in general appear to be in a problematic phase; since the end of the ’90s the participation to the official standardisation bodies is decreasing [12] and one of the reasons is the (low) speed of the standard definition correlated with technology life cycles [104]. Many private consortia have been set up to overcome the problem but the results so far are not significantly different.

These difficulties are clearly worsened by the wide complexity and heterogeneity of the business application scenarios. The multiplicity of E-business models, production processes and services surely enhances the business relationships through new commercial paradigms [116], but on the other hand it presents different issues and priorities to solve [122].

Due to these problems the top-down standardisation has proved to be very inefficient. This has led to the creation of closed proprietary islands that are hampering the growth of the business. A more recent approach focuses the efforts on sectorial perspectives in order to limit the domain, improve the reactivity of the users and shorten the time to release [71]. Nevertheless this approach has resulted to be unsuccessful in some industrial sectors. For example, in the T/C (Textile/Clothing) sector, despite the huge potentiality and the need for such standards, the B2B document exchange is still considered a hampering factor [35]. In this sector the EDIFACT technologies never really caught on and still today new Internet-based systems are not spreading.

This is due also to the large presence of SMEs, and to the absence of market leaders that rule the whole sector and can impose technological adoptions. This scenario, common to other industrial sectors, requires the adoption of different approaches to the creation of common standards.

In [61] it is recommended to rethink the current standard-setting processes and also to focus on different technical means (i.e. support to an incremental approach, higher attention to simplicity and usability). The content layer architecture of a collaborative framework is strongly affected by these considerations; the vocabulary of business-terms upon which a document-based collaborative framework is set up represents the core of this layer. What is really needed is a better mechanism to develop and to adopt B2B document exchange, or, in other words, to develop B2B protocols.

1.2 The thesis

The thesis presents a novel - ontology based - approach and outlines an architecture that improves the definition and management of Business-to-Business protocols. The background of this work has been the study of the state of art in ICT technology to develop tools for E-business as well as the study of the existing interoperability framework and standards as well as the collaboration in the Moda-ML project for the growth of a document-based collaborative framework (tested in the Textile/Clothing sector) that facilitate the creation and the management of a sectorial standard.

The structure of the thesis reflects this path; it is in fact divided in two main sections: in the first section (chapters 2-5) I will provide a general overview of the history of B2B technologies and the evolution of interoperability architectures. In my perspective, the idea of enterprise interoperability is achieved only with the integration of three main aspects: the development of new technologies, the integration of these technologies into interoperability frameworks and the definition of common standards that can exploit them to improve the efficiency of business relationships.

Following this idea, at the beginning the focus has been in the study of a classic, abstract architecture for interoperability, and the main functionalities it should provide. Therefore the thesis provides an overview about the main trends in ICT technology development for interoperability, and about the main standardisation initiatives bounded with E-business. It is important to connect these two worlds, in order to identify the new emerging technologies that can act as support for the development of new efficient standardised interoperability frameworks. The use of XML technologies and, at the same time, the need to introduce a semantic description into the definition of interoperability frameworks represent briefly two of the main reference points in the work.

The second section of the thesis is focused on two distinct parts:

- the first part describes the implementation of an interoperability framework (developed within the Moda-ML project) for the Textile/Clothing sector. This activity, that has also resulted in a definition of an European Standard for the sector, has represented the first effort to make the wide plethora of SMEs (that are the backbone of the Textile sector) interoperable. The main need is to improve the competitiveness of the enterprises by providing them new tools that can speed up their business processes.

- in the second part, the thesis focuses on the need to develop new procedures and tools to manage business protocols and data formats that must be adopted to exchange business information: the starting point to this work is again the consideration that, in order to allow two different systems to interoperate, a language must be defined to exchange information between the systems. In the B2B scenario, this language is basically a B2B protocol that must be supported by both systems.

1.2.1 The experience of the Moda-ML project in the Textile sector

The development of the interoperability framework within the Moda-ML project was preceded by an analysis of the peculiarities of the target sector, the Textile/Clothing. So first of all the characteristics and the requirements of the SMEs have been identified; the characteristics of the production sector are:

- heterogeneity of the enterprises in the supply chain.
- high responsiveness of the supply chain to the market.

It is also relevant to take into account the low technological skill and limited financial resources of the enterprises. Moreover, the heterogeneity of the enterprises has led to the birth of a wide set of private specifications for the data exchange and the need for the definition of a common European standard. Considering these issues, the development of the framework for the sector has been based on three basic components:

- The definition of a common set of business documents, and an architecture for their management, to be adopted in business transactions. This set has been be the starting point for the definition of a standard for the sector. The definition of this set of documents tops on the implementation of a vocabulary of specific business terms.
- The implementation of a light-weight software that allows the enterprises to exchange the information, exploiting Internet transport protocols like smtp or http. The basic point is in the usability of such software and the documents.
- The definition of common business processes, and an architecture that allows its customisation, in order to face the heterogeneity of the sector.

As aforementioned, all these activities have been supported by two standardisation efforts to build an European standard: first, TexSpin[115] is an initiative promoted at a european level by CEN/ISSS (a branch of the European Committee for Standardization concerned with initiatives for the Information Society), co-ordinated by Euratex (european association of the industrial associations of the Textile sector). After the end of the Texspin workshop, the evolution of the standard has been taken by the TexWeave workshop, that ended in August 2006.

Considering the evolution of XML technologies, and the limitation of some interoperability solutions proposed, and also the urgent need to integrate in the most efficient manner the raising framework with the back-end application systems of the enterprises, the developers considered the idea to provide a semantic description for both the vocabulary, the documents and the processes. To solve this issue, an ontology builder has been developed to extract semantic information from the framework. The effort has produced an ontological description expressed using Semantic Web technologies.

1.2.2 An innovative ontology-based architecture for B2B protocol management

Considering the experience maturated cooperating in the Moda-ML project, I propose in this thesis a novel ontology-based approach to support interoperability frameworks improving the management of B2B protocols.

The aim is not to implement any specific B2B protocol for a specific business domain. In any case, this would not be really feasible and would result in a useless effort for a single person: this task should be assigned to more authoritative subjects, like standardisation bodies or enterprise consortia, rather then performed by external entities like research centers or universities.

On the other hand, the proposed framework and the bounded B2B protocol management infrastructure ease the work of standards developers and provide tools that can be easily adopted in every business scenario, also when the complexity of the business processes to implement and the low technical skills and economic resources of the enterprises represent a considerable obstacle.

The easy adoption in different production sectors is strictly related with the choice of a ”vertical perspective” in the building of a B2B protocol: the idea is that, especially in specific and heterogeneous business sectors, the framework generates a set of documents tailored to match the specific requirements of the enterprises. This avoids the complexity of those customisation mechanisms that must be used by target users to adjust generic data formats or documents to their own needs.

The first consideration in the thesis is that business interoperability needs the definition of proper business languages (that must be supported using standardisation tools) to exchange business information.

The second consideration in this thesis is the need to introduce a semantic layer within the design of an interoperability architecture, and in particular in the definition of the B2B protocol.

The third relevant point is that this semantic layer must not be extracted from a given interoperability framework, but must be considered as the first step in the definition of the framework itself. The semantic layer becomes the main reference for the definition of both the B2B protocol and any other component. One of the purposes of this approach is to improve the definition of vertical business standards, or, in other words, the idea is to support the development of interoperability platform for specific target sectors.

To coherently manage the needed information about the B2B protocol and considering also the definition of interoperability, I first prefigure an abstract model of an interoperability framework that is composed of three main abstract components (layers):

- The vocabulary component, that comprises the definition of the terms used to exchange information, and should specify as good as possible the semantics of the information. It must be specified (also considering the vertical approach adopted) that in general this layer can't aim to be a complete, universal and exhaustive knowledge representation, but should instead represent the semantics of a sectorial and specific domain knowledge that underlies the framework.

- The document component, that comprises the document structures used to exchange information between partners. This component is mainly devoted to manage the syntactical aspect of the B2B protocol, and fixes the data structure that can be exploited in the communication.

- The process component, that outlines the business logic and the collaboration scenario of the production sector, modelling the exchange activities that drive the enterprise collaborations. The processes define for example the data exchange sequences between business partners.

Each of these layers is defined to manage one of the three aspects of interoperability: in particular, the vocabulary layer concerns the semantic interoperability, the document layer concerns the syntactical interoperability and the process layer regards the structural aspect of interoperability (i.e. the definition of the exchange model between partners). Consequently, each of these layers will use different technologies (at the moment OWL represents one of the reference technologies for a semantic layer - other proposals are available - , XML Schema is a consolidated validation language to express data structure, whereas many different proposals (see chapters 4-5) deal with business process definitions).

It is possible to note that basically the documents are composed using terms, so the definition of a document layer is based on the definition of the vocabulary layer. The definition of a vocabulary of business terms represents the construction of the basic blocks upon which the B2B protocol will be built. So the definition of the vocabulary is crucial in order to make the framework meet the requirements expressed by the enterprises. In particular the structure of the vocabulary influences the maintenance, extension, usability, and scalability of the framework.

Following this idea, the process layer is based on the document layer, since the process specifies which documents are used in the business transactions, and how they are exchanged. In this perspective, this architecture is the basis for the novel approach to build protocols and common standards: in fact nowadays the design of interoperability frameworks often starts from the definition of the documents in term of syntactic structures and knowledge domain modelling is not considered: whereas a common vocabulary is provided (as in many E-commerce frameworks, or standards), there isn't a formal specification of the knowledge domain that underlies the vocabulary. In my vision the specification of the semantics of a knowledge domain represents on the other hand the first step that binds every other aspects of an interoperability protocol or standard.

Being the first layer I defined, the semantic layer is the starting point to seed the definition and implementation of other parts of an interoperability framework, for example it could be used to:

- define datatypes and document structures.
- improve business standard definitions, management and adoption.
- enable back-end application integration with the protocol.
- support the maintenance and development of the framework.
- develop user-friendly interfaces for protocol adoption and customisation.
- provide a basis in the context of Semantic Web applications (i.e. Semantic Web Service)

The introduction and management of the semantic layer is allowed through the definition of:

- A model, based upon and represented with a set of ontologies that define the structure to adopt and to extend while designing the semantic description of the knowledge domain, and a set of guidelines to design the specific domain ontologies.
- A set of software tools to generate from the semantic description the other components of the framework. Together with the definition of the semantic layer, it is relevant to define the syntactical specification for the target B2B protocol or, in other words for the datatypes to use in the electronic communication. The architecture includes an automatic mechanism to both generate the datatypes related with the ontology, and to link the semantic descriptions provided by the ontology with the syntactical definitions of the datatypes (that are formalised using the XML-Schema language).

More in detail, the functionalities of the proposed framework are provided by a strict integration of different components:

1. A Domain Ontology.
2. A System Ontology.
3. A Datatype library.
4. A set of Documents.

Exchange processes could be fundamental in the definition of interoperability mechanism between heterogeneous systems. In this thesis exchange processes are recognized as a relevant part of an interoperability framework to develop, but are not discussed.

In the following I sketch the role of the four components listed above.

The idea in the framework is to provide a simple model (represented by a part of the System Ontology - the KDMO, Knowledge Domain Model Ontology) for the definition of the Domain Ontologies. This model is used to differentiate in the ontology the main concepts to manage from their properties, allowing the identification of the basic bricks of the framework.

The Domain Ontology can be considered as that element that is plugged into the B2B protocol management architecture to generate all the other components. In another perspective, it is the parameter that characterizes the resulting protocol, depending on the target sector. The role of the Domain Ontology in the framework is then twofold: on one hand it must structure the knowledge of the domain that underlies the B2B protocol: this means exposing a sector-specific knowledge to the world by defining concepts through semantic relationships and particular constraints; in this vision the ontology mainly concerns that part of a well-defined knowledge that tends to remain unchanged for a long time. On the other hand the domain ontology should represent the basis upon which we can build a set of datatypes for the B2B protocol itself.

The Domain Ontology should be built up and filled starting from many information sources strictly related with the target domain. These information sources are represented both by document collections and human experts whose collaboration is crucial to develop a protocol that really reflects the needs and the requirements of the business sector.

While modelling the Domain Ontology, the central point is that, in order to generate a set of XML datatypes (and we want to generate these datatypes in a semi-automatic way), part of the Domain Ontology must be designed following the model defined in the KDMO (Knowledge Domain Model Ontology). Together with the use of the KDMO, some simple guidelines have been outlined to drive the design of the Domain Ontology.

The System Ontology represents that part of the framework that does not depend on the specificities of the target business domain. It is divided in two sub-ontologies: the KDMO (Knowledge Domain Model Ontology) and the DO (Datatype Ontology).

The KDMO allows the identification of the main concepts that must be managed by the framework, and to specify which properties must be used to described the concepts.

On the other hand, the Datatype Ontology is used to structure the data formats, to identify datatype specifications and to add some other information to manage datatype interoperability. The Datatype Ontology is used as the bridge between the semantic representation of the knowledge domain held in the Domain Ontology and the definition of a set of datatypes; it is also used to maintain information about the use context and the mapping mechanisms between different datatypes.

The proposed architecture comprises a library of (XML) datatypes. To manage this library, the framework prefigures a mechanism to generate a set of XML types, that are defined using XML Schema, to exchange the information previously modelled: once the knowledge domain has been described in an ontology by a domain expert, a tool builds and associates a set of XML types to the identified concepts and properties.

The generation of datatypes is limited to XML fragments: if the concepts in the Domain Ontology must be associated with a set of non XML fragments, these types are represented only using instances of the Datatype Ontology, that exploits specific defined properties to describe the formalism adopted.

Designing the architecture related with the generation of the XML datatypes, the thesis considers several issues:

- the definition of XML datatypes sub-sets.
- the choice of the programming style for the generated datatypes.
- the definition of transformation patterns to create XML Schema types.
- the design of the building mechanism of XML types.

In particular, the building phase is performed in two stages: an automatic stage, where there is a first formalisation of the datatypes and an automatic tool exploits as much as possible the definition of the Domain Ontology to deduce the proper basic structure for the datatypes; a manual stage, where a human can fix the so called ”syntactical” holes that are marked in the automatic stage in order to complete the building of the library. During the automatic stage the identified patterns are applied to transform those definitions of the origin ontology that are compliant with the KDMO in the datatypes library.

The datatypes library does not complete the design of a B2B protocol: to this aim a set of document templates must be defined. The document definition is performed on a semantic basis after the ontology definition: in order to define a document template to adopt within a business process, a human personnel identifies and selects the information that the document should carry.

This operation is performed exploiting the ontology definition and browsing the ontology itself with a software application.

Practically, the document building is not performed with the syntactical definition of the document templates, but with the semantic identification of the information that must be treated in the document. This mechanism strongly exploits the connection between the Domain Ontology and the Datatype Library.

Throughout the explanation of the proposed framework in the thesis, I will consider as an example, the definition of a document to exchange information about the characteristics of a fabric, the TEXSheet document. This document may be considered as the Identity Card for a specific fabric, and the information included in the TexSheet document result naturally very relevant in commercial transactions; often the enterprises have their own perspectives about the treatment of such information and the integration of such perspectives is needed. Moreover, in a real complex production sector like the Textile/Clothing one, both the amount and the complexity of the information available to describe a fabric could be considerable.

Part I

The context

Chapter 2

B2B integration scenarios

In this chapter and in chapter 3 I will provide (from [11]) a general overview of the history of the B2B technology and architecture evolution. [11] is a book that deals with the main aspects releated with the problem of the integration, and it can result really useful to skecth the scenario of the integration issues and platforms. This and the following section of the thesis summarize what I think is a fundamental introduction to this topic. Moreover, this book is a reference for a wide set of different and heterogeneous research initiatives, not only bounded with its author (that, in any case, cooperates in different relevant projects starting from the analisys provided in [11]). Just for a simple example, [11] is cited in:

- [10], that faces the introduction of an explicit semantics in EAI.
- [70], that outlines a generic framework and model for business interoperability (BIF) and describe the components for enterprise interoperability. This decription considers not the technical aspects for interopeability, but analysis business processes and strategy.
- [20] faces the problem of process mediation in the context of semantic web services, in the Web Service Execution Environment (WSMX).
- [102], that presents a workflow-based architecture to manage B2B interaction.

Clearly, the list is much more longer then the aforementioned. Anyway, from this short list it can be evaluated as the notions presented in [11] are been reused in different contexts.

Many other paper do not cite directly [11] but are strictly realted, in my view, with it.

The first section provides a brief description of the major types of integration technologies, whereas the second section describes more deeply the main architecture adopted in time to achieve B2B interoperability.

2.1 The evolution in the integration perspectives

Integration technologies evolved until now over about 30-35 years. This evolution was demanded by the enterprises that, since their information systems and their applications were more and more used to manage commercial data and information, needed new and valuable solutions to efficiently connect each other. The first, main, initiative to create a widespread integration framework was EDI[41], but afterwards more sophisticated technologies and framework were developed to solve more complex integration and interoperability problems; a new, highly dynamic, models and mechanisms of commerce were in fact rising in the world scenario.

This evolution is still in progress, and more and more trading partners, business processes, data and information need to be connected and integrated.

The history of integration is composed of different phases during which the enterprise requirements change, forcing the development and adjustment of new interoperability architectures. Naturally, enterprise requirements and technology solutions are strictly related, and they condition each other.

Along the whole history, three major types of integration technology can be identified:

- Application To Application (A2A) integration: deals with the integration of different back-end application systems within the same enterprise.
- Business To Business (B2B) integration: deals with the integration of different trading partners exploiting connectivity over network like Internet and the adoption of B2B protocols like EDI or others. This kind of integration overshoots the enterprise boundaries.
- Application Service Provider (ASP) integration: the effort in integration is managed by an enterprise that acts as a service provider (the ASP) hosting data for integration purpose on behalf of the subscribers, providing access to them within a pay-service.

These types of integration are in general all required by the enterprise: in many instances an enterprise has more than one back-end application system, and needs to integrate them with more than one business partner. In the following I will provide an overview of the approaches to solve B2B integration issues.

All these three different integration mechanisms can regard a single enterprise, since it is natural to suppose that it can have different back-end application systems to integrate, together with different trading partners with which it exchanges information. For this reason it is almost impossible to say which of two integration technologies, A2A and B2B, came first.

In any case, originally, the enterprises had up to one back-end system. When the need to connect these back-end systems between different enterprises arose, the first significative and wide adopted solution was represented by the EDI, that provided a set of standard B2B protocols, tailored for many different kinds of industries (Automotive, Tourism, Pharmaceutical, Textile/Clothing), to use in the business communications. In any way, each enterprise adopted (and still adopts today, naturally) its own data format to internally manage business data and information; on the other hand, to use the B2B protocol with an external partner, the enterprises had to translate their outgoing data into the EDI format,and the incoming EDI messages into their own format using an EDI translator. The messages were then exchanged using a value-added network VAN (see also chapter 5 about standards).

Hereafter the enterprises started to use no more only one, but several different back-end systems (i.e. Enterprise Resource Planning - ERP, Customer Relationship Management - CRM), that used different (proprietary) internal formats. These back-end systems needed both to be integrated with the other back-end systems and to exchange messages towards trading partners. To solve this complex scenario, the enterprises installed an A2A integration architecture (a point-to-point or hub-and-spoke one) to integrate the different application systems. Each of them used also a proper EDI translator to convert the internal data into the EDI format.

A further complication arose when, in the B2B scenario, other standards appeared together with the EDI. In this situation, those enterprises that needed to exchange information with different partners sometimes could be obliged to support different B2B protocols, depending on the partners. To manage this complexity, and this variety of standards, the enterprises needed different ad hoc translators, like the EDI one, that could translate the internal data formats in the right B2B protocol to use in the communication.

In this scenario, the enterprises managed different back-end information systems, integrated each other using an A2A technology, and also had different translators in order to use different B2B protocols to communicate with external partners. Moreover, the messages could now be exchanged over different networks, like Internet; the VAN is not more the main communication mean, and many different protocols (like FTP, HTTP, SMTP) must be supported, increasing software complexity.

Naturally, the installation and maintenance of such architecture was not really simple, and also not very cheap. In order to simplify this configuration, and also to meet the requirements of many small and medium enterprises that asked for a more ability in using the new technologies to do their business, the ASP solution was proposed.

In the ASP (Application Service Provider) model the enterprise data are hosted by another enterprise, the ASP, that manages them on behalf its subscribers, providing the requested connectivities towards other subscribers of the ASP (see later section 2.2.5).

In this scenario, if the subscriber has other local back-end application systems that need to be integrated with the hosted one, it need a protocol to communicate with the hosted one. The integration architecture thus must include an ASP connector. In this way, an enterprise could have to face three different integration mechanisms: the B2B integration with external partners, the A2A integration between its various local back-end systems, and finally the ASP integration to connect with hosted data.

This evolution has surely not concerned all the enterprises or business sectors, but in some manner these phases have recurred in different situations.

Considering the history of the integration architectures it becomes clear as the complexity of these kinds of architectures has grown more and more in order to tackle the three different forms of integration and the increased requirements of the enterprises. Basically this complexity involves also the installation, management and maintenance of different softwares, from different vendors.

In order to compose and remove this complexity, the valuable development of integration solutions must include the functionalities provided by the aforementioned architectures into only one simpler architecture, designed on well-defined set of concepts, that becomes the central point of the integration. Avoiding complexity results to be fundamental to make an architecture usable also by the Small Medium Enterprises (SMEs).

2.2 B2B integration architectures

The need for B2B integration technologies arose when it was clear that back-end application systems designed and built to operate in isolation was an obstacle to improve the efficiency of the business processes. In fact those applications were, and today remain, fundamentally heterogeneous, autonomous and distributed. In this situation the only way to exchange data was to manually re-insert data into the various different systems.

Automatic data exchange without manual intervention becomes then necessary to speed up business transactions; to this aim, different software solutions and architectures for B2B integration were considered and developed over time in several stages. In the following I will provide an overview of the main approach to design interoperability architectures.

2.2.1 Homegrown Integration

The first approaches to solve B2B integration issues where undertaken by those (big) enterprises that needed to integrate their back-end application systems with those of other partners to exchange data. In this case these enterprises implemented the integration functionalities themselves since no integration programs were available on the market from software vendors (mainly because this was not considered a profitable business).

With these solutions the integration technology was embedded in the back-end application system that had to be modified to called synchronously each other to exchange data. Asynchronous solutions, on the other hand, used an intermediate storage (database, file system or queuing system) to pass data. Back-and application systems using asynchronous solutions agreed on the intermediate storage location.

When it was clear to software vendors that enterprises needed solutions for their systems, integration products began to appear on the market.

2.2.2 Point-to-Point Integration

In this simple situation, an enterprise need only to install a dedicated B2B protocol adapter to connect the back-end application system with those of their partners or with its supply chain, without the use of a complete B2B integration architecture. Especially small enterprises in fact can consider to difficult and expensive the buy, the installation and the mainteinance of a complex architecture.

Naturally, the dedicated B2B adapter can support only a specific B2B protocol, but cannot provide other kinds of connectivity or services.

Point-to-point solutions solved the integration problem establishing a different connection for data transfer for each pair of back-end application systems to integrate. The integration applications extract the data from back-end application systems, transport it to the other system and insert it there. The data transfer can be implemented in different way:

- Synchronous communication. In this case, once the integration software extracts data from a back-and application systems, it executes the needed transformation on the data, and then it synchronously invokes the other systems to pass it the transformed data to it.

- Asynchronous communication using intermediate storage. In this case the integration software, once it has extracted the data from the source back-end application system, applies the needed transformation and stored the result on an intermediary storage like database or file system. Soon after, the integration software will take the transformed data from the storage and will insert it in the destination system.

  The two processes of extraction and insertion of the data are completely separated and independent and are executed from different parts of the integration software. In this way the two integrated back-end application systems do not need to be active simultaneously to exchange data. The transformation process could be itself a separated process within the integration system. The use of dedicated queues instead of database or file system implicitly imposes an order on the data exchanged during the extraction-retrieval process.

2.2.3 Hub-and-Spoke Integration

In many cases, enterprises hold several relationships with different trading partners. If the enterprises need to use a specific B2B protocol to communicate with each of its partners, the communication infrastructure could result to be too complex to be managed efficiently, especially for those medium-little enterprises that do not have enough economical and human resources to employ. Anyway, also for big enterprises, a cumbersome integration platform can be very expensive, inefficient and risky. An Hub-centered integration architecture can help to overcome these issues: in this case every communication between two different partners exploits a central hub as a transformation medium to convert each other the different B2B protocols.

Each enterprise exchanges the messages only with the central hub using a single specific B2B protocol. Then the Hub is liable to convert the incoming messages in the proper target B2B protocol, and to route them to the right destination. An enterprise has now only to translate a single B2B protocol (i.e. the format used for the incoming-outcoming messages) into the internal format of its system. With this configuration, the effort to manage several different relationships is notably reduced. Moreover, the hub can also provide different kinds of services, like logging mechanisms for the communication, or statistical analysis about commercial relationships (fig.2.1).

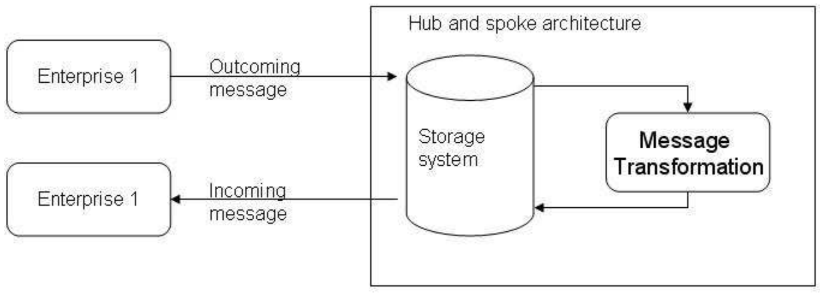


Figure 2.1: Hub-and-spoke architecture

Hub-and-Spoke integration introduces a central and common storage that is used for all the possible data exchange. Thus, for each pair of back-end application systems to integrate is no more needed a specific connection for data transfer, but can be used the unique connection with the central hub.

The central hub can receive data from every system, transform it in a proper way for the target system and than insert the transformed data in the destination. As in the point-to-point architecture, the three processes of extraction, transformation and insertion of the data could be completely separated and independent and they are executed from different parts of the integration software.

Since there is no an explicit connection between the two integrated back-end application systems, the sending system (spoke) has to specify in some way the target system, for example in the header of the message exchanged. With this approach, each spoke is aware of all of the other spoke partners in order to be able to address them through the central hub. Using a publish/subscribe schema to match the messages, each spoke can provide the hub its own requirements to receive data, and upon these requirements the hub can identify the proper target spoke for the received data. The addressing of the messages is based on the content of the message itself, the schema is called content-based routing.

2.2.4 Process Based Integration

Both the Point-to-Point and Hub-and-Spoke integration do not face two relevant issues:

- Multistep integration: during the message exchange it is not possible to involve a third partner in the communication.
- Business logic management: no addition activities (i.e. authorization) can be performed by the integration architecture between the operations of data extraction (receive) and data insertion (send).

In order to overcome this issues, the Process Based Integration extends the hub-and-spoke and point-to-point integration models adding process management functionalities in the form of workflow management system. In this way the integration system can insert the received messages in a workflow instance to determine the proper way to process the message itself, together with the other related messages. Thus the workflow instance determines the managing of the whole message exchange between the back-end application systems involved, that could be more than two.

The Process Based Integration is thus composed of a storage system, that is used to store data, and a workflow system, that extracts and inserts data into the storage system and transforms it, depending of the workflow execution. With this approach both multistep integration and business logic can be managed by the workflow management component of the integration system. This architecture addresses all the major drawbacks from the previous approaches: in fact it can "remember" the status of the workflow, providing a way to manage more complex business scenarios; business logic can be built into a workflow definition, using constructs to express, for example, conditional branching or parallel executions. The main issues related with this solution are:

- Back-end application systems can adopt different business message formats: workflow designed to deal with different formats can result too complex and cumbersome to manage.

- The workflow execution can heavily depend on the trading partners involved in the message exchange or on the B2B protocol used. Each partner can require the use of specific rules, forcing the definition and adoption of different workflows.

2.2.5 ASP Integration

Application Service Providers (ASPs) install back-end application systems and rent access to them to other enterprises (its customers), called subscribers. Enterprise data are hosted by the ASP, that manages them on behalf its subscribers, providing the requested connectivities towards other subscribers of the ASP. The subscribers pay a fee to the ASP and can access the back-end application system using Internet, for example via a web browser (fig. 2.2).

This is the principle of the ASP service, but there are many different topologies related to this architecture. In fact

1. an enterprise may have all of its application hosted by the ASP (that acts on behalf its subscribers for each integration service), or only part of them.
2. there are different mechanisms to interconnect the ASP and the enterprise: the hosted data can be accessed, retrieved or queried using a web browser, or exchanging messages between two different integration architectures, or also using a dedicated adapter that implements a specific B2B protocol.
3. the ASP service can provide the needed integration between different back-end application systems of the same enterprise, or can be used to make interoperable two different application-systems of different enterprises. When an ASP is used to interconnect two different application systems, the two involved partners do not know anything about the integration mechanism, nor the two applications are aware if the partner application is completely hosted by the ASP or not.
4. two different enterprises can also subscribe two different ASPs, having their systems completely hosted or not, that are then connected each other. In this situation the two enterprises can be integrated passing by two different ASPs that manage the integration on behalf their customers.

Hosted Integration
In different scenarios, the enterprises may want to exploit ASP integration services, but do not want to have their data hosted by the ASP; on the contrary, they require their data locally installed. In this situation, ASP are used only to provide integration functionalities between different subscribers. In general the enterprises connect their back-end application systems with the ASP integration architecture using a dedicated B2B adapter. The integration of the two back-end

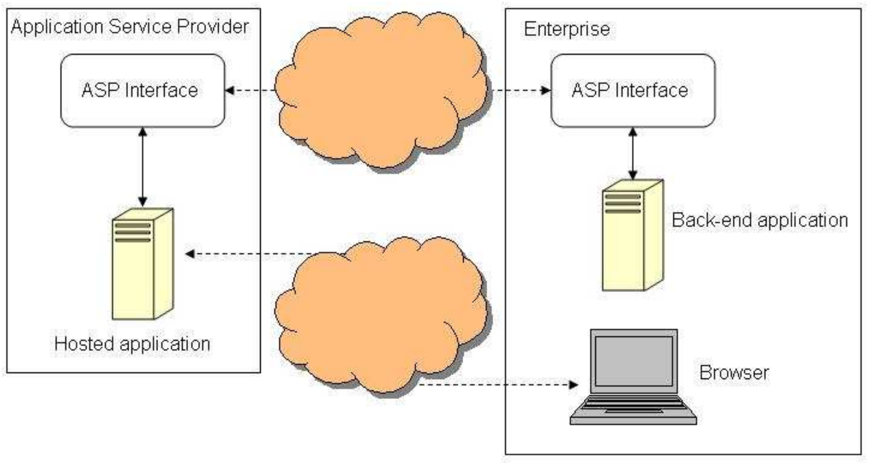


Figure 2.2: ASP architecture

application systems is then performed by the ASP server on behalf the two enterprises. In this configuration only the B2B integration server is hosted and not the back-end systems.

Reverse Hosting
Enterprises can also exploit the reverse hosting services by ASP: in this case the ASP do not install any back-end application system nor any integration architecture on its own. Both the back-end application system and the integration architecture reside in the enterprise site and the enterprise maintains than the whole control on the data. The ASP installs instead management software on the enterprise back-end system to manage the data on behalf the enterprise (the subscribers). In this way the customers of the ASP outsource the management of the software to an ASP.

2.2.6 A ”mature” architecture?

Nowadays a B2B integration technology architecture needs to encompass and manage may different issues of integration at the same time. In order to do this [11] sketches a ”mature” layered abstract architecture, composed of four main components:

- The user interface layer
- The integration logic layer
- The connectivity layer
- The persistence layer

Each component manages specific functionalities required by the integration process. I will better describe this abstract architecture in section 3.3.

2.3 B2B application scenarios

The design and development of integration architectures must obviously take care of their potential usage, that depends on the systems and the business models adopted by each single enterprises, and the commercial relationships to support. [116] provides a definition of a business model and a list of eleven models for electronic commerce, or in other words, the scenarios of electronic commerce that are now emerging beside the traditional ones (i.e. Value-chain service provider, E-shop, E-procurement, E-auction, E-mall and so on). These models represent the available mechanisms for the enterprises to do business exploiting new communication channels, like Internet, provided by the evolution of the Information Communication Technology (ICT). The adoption of such models and of the new technologies requires, from the enterprises, a (considerable) effort to reconfigure and reconstruct their information systems.

In this section I will summarize from [11] two typical use cases that a B2B integration architecture must be able to face.

Supply chain integration
In a supply chain two or more trading partners are connected in a chain exchanging business information, to implement or to support a specific production process or commercial relationship. The idea is that every partner manages a specific phase of the production process, and communicate using a network with its neighbours in a bilateral way; in this scenario not every partner involved has to communicate with all the other partners (although in some situation it can be the possible). Each trading partner can use a specific B2B protocol or network to communicate with its neighbours.

Marketplace integration
In a marketplace scenario, suppliers can access to a shared market where can offer their products. Then the buyers can access to the marketplace to see who are the suppliers and which products are on sale. The main scope of the marketplace is then to allow sellers and buyers to meet each other. In principle, the following business transactions and data exchange are performed outside the marketplace. More sophisticated forms of marketplace allow also to exchange business information (like a purchase order)and to perform business transactions within the marketplace, providing matching or agreement mechanisms. In this case, the trading partners do not need to have a direct connection with each other, nor a direct communication channel exists between the

buyer and the seller.

Chapter 3

A conceptual framework for B2B

3.1 Integration approaches

The history of integration is full of ideas and initiatives to provide better and better solutions to the A2A integration problems and to enhance B2B interoperability. During this history can be identified two main approaches building an interoperability architecture. In this section I will describe these approaches and an abstract model (from [11]) of an interoperability architecture.

3.1.1 The document-based approach

As said in chapter 2, the first endeavor to provide a valuable interoperability framework in the B2B field was done by EDI[41](I will describe EDI later in section 5.3.1). This approach can be labeled as a document-based approach for B2B interoperability. The main idea of the document-based approach consists in the definition of a set of complete business documents that form the basis for the B2B integration [27]. These documents are then exchanged between commercial partners and represent the B2B protocol. The commercial partners do not share any other kind of information except these documents, and do not access partner systems. Nor this approach takes care of the exchange mechanisms adopted. In general document-based approaches are associated with loosely coupled relationships. In other words, this means that enterprise systems are loosely coupled and largely independent each other.

3.1.2 The middleware-based approach

Other initiatives to solve the B2B interoperability problems highlight the role of the software tools and of the architectures used to exchange business information and to implement business processes.

The main idea of these approaches consists in the design and development of software architectures (middleware) that could strictly interconnect two different enterprise systems to exchange business data. They stress the aspect of software interoperability between the enterprise systems. The interoperability in this case is basically achieved defining standard interfaces and APIs for these architectures.

With the development of new advanced software solutions for interconnection, a new approach for B2B integration arose in this area, that provided mainly more efficient and secure mechanisms to exchange business documents. [65][9] show as we are now moving from the old monolith systems towards the so called component-based architectures.

As the name can suggest, a component-framework (like CORBA, DCOM, JavaBeans) is an infrastructure composed of different integrated components, each of which can be delegated to provided a well-specific service for the overall system.

All the components co-operate to implement the features required by the business processes or the business models, and can naturally be geographically distributed. In general these architectures do not provide any definition of B2B protocol or of business message formats, nor require the use of a specific one, but on the other hand lead to a tightly coupled interoperation paradigm, that results also to be really complex.

Finally, a novel approach is proposed by [134]. In this paper the authors observe that functional requirements of electronic commerce applications vary significantly by business domains and propose a domain-oriented approach for ECMS(Electronic Commerce Management Systems) development. The idea is to start from a generic ECMS upon which to develop more specific ECMSs tailored to support the need of precise business domains. For the author, ECMS represents the right evolution of E-commerce applications, that has to grow out of component-based architectures.

3.2 Basic concepts

In this section I will sketch the main concepts that recur in the interoperability research field. These concepts constitute the basic bricks used within an interoperability architecture, and represent the elements around which the components of the architecture are then designed, developed and deployed.

Endpoints
Integration architectures are used by enterprises to exchange and integrate in a useful manner business information. In this context, the enterprises, or the partners, or the back-end application systems represent the endpoints of the communication and the integration. Endpoints are then

the subjects of the integration, and provide also the access points to send and to receive the messages.

Message

Trading partners exchange data using messages. These messages can be transformed many times in order to be read and understood by the applications involved in the communication. A B2B message can undergo several phases, taking thus different structures. A message can be:

- A wire message: this is the message that is transmitted over the network, and contain both the business information and the transport headers.
- A clear text message: this is the wire message once it has been stripped from the transport information.
- Application clear text message: this is the message in the format the back-end application can understand.
- Application wire message: this is the application clear text message with all the transport information added.

Events

An event is a message that is received or sent together with all its metadata information (i.e. date, destination etc.). The life of a integration architecture is a sequence of events, in other words a sequence of messages that are sent, received, elaborated and stored.

Transformation and translation

In general, the messages used by two or more partners to exchange business information do not implement the data format of the back-end application systems, but adopt a specific one. Then, in order to process the exchanged information, when the messages is received by a partner, it must be transformed by the integration architecture before being passed to the back-end application. A transformation consists of a set of rules that must be applied to the incoming message to obtain a understandable message that can be processed and stored.

Business Processes

In general trading partners do not implement single independent communications during their business transactions: the message exchange in fact follows more complex pattern (many messages and activities could be related each other, like purchase orders and purchase order ack; invoices and catalogues are other examples of business information that are exchanged in several stages).

In order to manage these complex business scenarios, the partners adopt the definition of business processes that can regulate the message exchange. The automatic management of the processes is then entrusted to a specific component of the integration architecture, that performs the execution of the business processes.

Basically three types of business process can exist:

- the interface process, that describe the message exchange between different endpoints.
- the business process, that describe the internal process of an enterprise.
- the binding process, that bind the interface process with the business process.

Finally, it can be that these three different kinds of processes coincide, so only a single implementation and management of one process is required.

3.3 An abstract model of an integration architecture

The term architecture can assume different meaning depending on the usage context, and the perspective of the discussion; moreover this term is often used together with or in place of the term framework, whereas framework should be better used to suggest a logical structure or to classify and organize the descriptive representations of an Enterprise [133]. In this section I will depict a classic abstract architecture. This means that this architecture does not represent an implementation of a specific integration architecture, and a set of related applications and software components. On in the other hand this architecture outlines the basic structure and the abstract components that underlies an integration architecture.

This architecture is composed of four layers (fig. 3.1), each of which wraps a well-defined set of functionalities, that cooperate to provide for the overall functionalities. In the following I will describe the main components of each layer, outlining the functionalities each of them provides.

Like other layered models, components in the upper layer exploit functionalities provided by lower layers. The four layers are, from the bottom to top:

1. The persistence layer
2. The connectivity layer
3. The business logic layer
4. The user interface layer

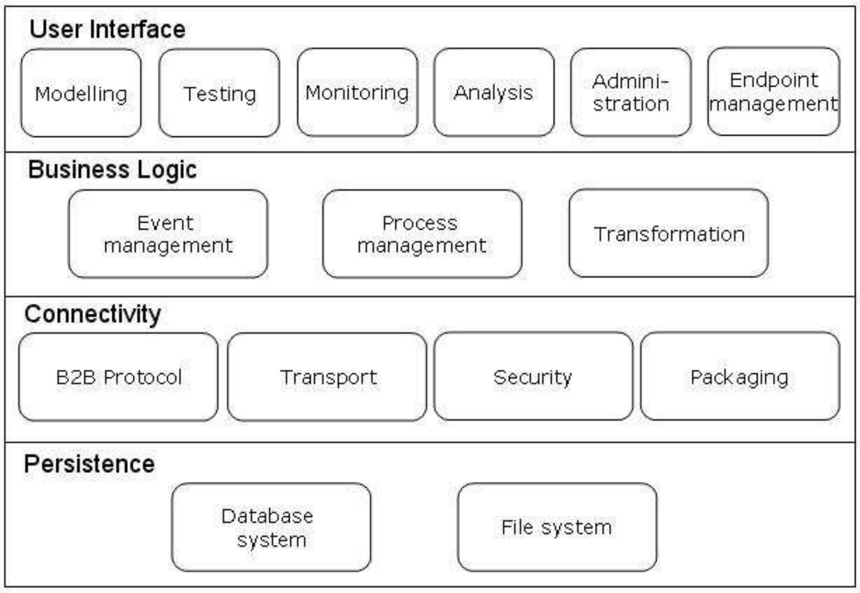


Figure 3.1: An abstract model for a mature interoperability architecture.

3.3.1 The Persistence layer

The persistence layer concerns of the reliability and consistency of the data managed by the architecture, and finally it does not provide any integration-specific functionalities, but allows the components in the upper layer to operate upon data. This layer is composed of standard, off-the-shelf products, like DB systems, file systems or persistence queuing systems.

DB systems implement various data models to store data (relational, object-oriented, etc), allowing the upper layer to insert, delete, update and select data into tables. DB systems may also support services like transactions management, reliability, information backup.

The persistent queuing systems implement a queuing behavior in managing data.

File systems are used to store data in files.

3.3.2 The Connectivity layer

The connectivity layer consists of those components needed to connect different integration applications of different trading partners on a network, as well as those components needed to connect the integration application to back-end application system, or an ERP system of an enterprises. These components are:

1. The transport component: this component implements the transport protocols used to exchange data on the network (e.g. on the Internet). In the integration application context,

these transport protocols can be considered as the basic elements needed to send and receive messages, and obviously they have to be implemented by all of the involved partners in the communication. Examples of these protocols are HTTP, SMTP, FTP, EDIINT - a standard to exchange EDI message (IETF standard). ebXML has released a set of specification, Messaging Service Specification (MSS), that define a support for the secure transport of messages. These specifications are, on their own, based on MIME, SOAP and FTP, SMTP and HTTP. RosettaNet standard provides to this aim a transmission infrastructure called RosettaNet Implementation Framework (RNIF).

2. The endpoint management component: it is used to manage the endpoints of the communication, and to identify the partners on the network. This component allows to create, delete and modify endpoint specifications.

3. The security component: this component provides the necessary security warrants on the communication, basically implementing a security standard. This component provides features like encryption and decryption of the messages, signature computation and check, non repudiation of the messages, authentication and authorization management. Among the developed standards in this area are for example XML Encryption and XML Signature (W3C recommendations), SSL (by Netscape Communication), XACML, SAML (OASIS deliverable), XML Key Management (W3C candidate recommendation).

4. The packaging component. If B2B messages are sent over a particular transport protocol, they can required to be packed to establish an application-to-application communication. The packaging component provides packaging functionalities for the B2B protocol components, in order to manage the message structure sent with an underlying transport protocol. Among the developed standards in this area are SOAP (W3C recommendation), MIME (rfc - IETF standard)

5. The B2B protocol engine: this component manages the B2B protocol, receiving and sending messages between trading partners and validating them according to the B2B protocol definitions. This component provides B2B connectivity exploiting the functionalities of the underlying transport/packaging components. This component may implement one or more of the several standards present in this area. Basically these standards provide the definition of messages and vocabulary to exchange data. There is a huge amount of these standards, that depend (and are in general developed) strictly upon a specific industrial sector; among these standards are, for example,: ACORD (by no-profit organisation for the insurance sector), EDI and UN/EDIFACT (UN/CEFACT standard), STEP(ISO standard for process plants), HL7(ANSI standard for the health care industry), OAGI (by no-profit organisation for both intra-enterprise and inter-enterprise message exchange),RosettaNet Dictio-

naries (RosettaNet consortium), SWIFT (company for message exchange between financial institution), UCC - Uniform Code Council - (for code identifier), UN/LOCODE (code for trade and transport location), UN/SPSC (classification scheme for products and services), D&B D-U-N-S Number (unique identifier for enterprise issued by the D&B company), UBL (OASIS standard, definition of both business document and business data component). I will describe some of these standards more in details in chapter 5.

6. The back-end application system adapter: this component provides the needed interface to bridge the B2B architecture to the back-end application system, or ERP system. This component obviously depends on the specific back-end application system used by the enterprise that has to be integrated. The adapter receives messages to be sent from the back-end application system and pass them to the integration application; on the other hand, it receive incoming business messages from the integration application and pass them to the back-end application system. It performs the needed transformation operation.

3.3.3 The Business Logic layer

This layer deals with the managing of the business logic. The related components cooperate to allow the right execution of the business processes. There are basically five components:

1. The event management component (to manage the various classes of events).
2. The process management component (to drives the process execution).
3. The transformation component (to transform the structure of the messages maintaining the same format, for example in case of different XML Schemas)
4. The translation component (to convert a message from a specific format in another different format)

While these different components can be distinguished in an abstract architecture, often these functionalities are embedded in a unique module or standard. Other abstract architectures gather all these components in a unique business logic component. In the following I list the main process standards related with the functionalities of this layer.

- Business Process Specification Schema (BPSS). BPSS is a process definition language that belongs to the set of ebXML standard.
- Business Process Execution Language for Web Services (BPEL4WS). This standard started originally by an initiative of IBM, BEA and Microsoft to provide a specification language for both private and public processes, and it is now a OASIS working draft (version 1.1)named

WS-BPEL [127]. BPEL4WS supersedes XLANG and WSFL, earlier proposed by Microsoft and IBM.

- OWL-S [91] (developed by a group of Semantic Web researchers) is a OWL-based Web Service ontology, which supplies Web Service providers with a core set of markup language constructs for describing the properties and capabilities of their Web Services. This ontology aims to ease the description of Web Services and their searching, discovery and composition.
- Partner Interface Processes (PIP). PIPs are defined within the Rosetta Net architecture[100] and represent (using UML format) well-defined domain-specific (basically for the electronic component supply chain) public processes: they basically specify the business message exchange sequences (i.e. the sequence for a purchase order).
- Business Process Modeling Language (BPML) [8]. BPML is defined within the BPMI (Business Process Management Initiative) initiative, a non-profit corporation that aims to promote and develop open, complete and royalty free xml-based standards to support and enable Business Process Management (BPM) in industry. In this context BPML consists of an abstract model and a XML syntax to define business processes, regardless of the target business domain.
- XML Process Definition Language (XPDL). XPDL is a WfMC standard [124] that defines an XML-based language to specify business workflows.

3.3.4 The User Interface layer

The user interface layer provides all those functionalities needed by the end-users (that can have different roles and skills) to configure, interact and manage the architecture. In the following I list eight components for this layer. Each of this components is basically implemented by an application module, and due to their very practical aims there are not standard related to them.

1. The modeling interface is used to provide a modeler with a set of tools and modelling constructs to model business events, business processes, data types, data transformations and translations, to define the final business integration model. Often the modeler can exploit a graphic interface to create and represent the business model.
2. The testing interface provides those tools to execute simulation tests upon the business model, to verify both the behavior and the performance of the integration architecture and the validity of the business model.
3. The monitoring interface is used to monitor the architecture during the execution of an integration model.

4. The analysis interface provides tools (like Data Warehouse tools) to analyse the final results of the integration model execution.

5. The administration interface provides the administrator all those tools to configure and administrate all the software components of the architecture. This component does not concern with integration functionalities configuration.

6. The endpoint management interface is used to define the endpoints and all their necessary attributes for the business processes.

7. The error-handling interface is used to define and manage system behavior in case of system failures.

8. The worklist interface is used to notify users about relevant events and to require human intervention to authorise special tasks.

3.3.5 A simpler model

A interesting survey on B2B technologies is [75]. In this paper a simpler abstract architecture for B2B application is outlined.

This paper identifies three main layers for B2B application:

- The communication layer: this layer manages the needed protocols for exchanging business messages between trading partners. In order to achieve a seamless integration, internal communication protocols used by the trading partners must be integrated with external ones (e.g. with translation procedure).

- The content layer. The communication between trading partners requires that each of them can understand the meaning of the business messages, in order to proper use them. This layer provides data format, data models and languages to structure and describe the information exchanged and fix a common and well-defined semantic to understand those information.

- The business process layer: this layer concerns the definition of common business processes between trading partners and their execution. This layer provides functionalities to publish and discovery trading partners, to understand the semantic of other business processes and to establish new business collaborations.

Within each of those layer, there are surely many other aspects to consider (e.g. security for communication), but this model highlights the three main components of an interoperability architecture. These components reflect in some manner also the world if the standardisation initiatives strictly correlated with B2B applications, that try to face the three main issues for business integration:

1. Definition of business data semantic. This definition regards the message payload.
2. Definition of business process semantic. This definition regards process execution, process transaction, partner profiling, partner agreement, partner advertisement and discovery.
3. Definition of common communication exchange means. This definition comprises transport, packaging and security issues.

3.4 Deploying an integration architecture

The definition of an interoperability architecture and of all its components is only the first step to integrate enterprise systems: this architecture in fact represents the interface of the enterprises towards the external partners. Anyway, this architecture has to be installed, configured, and the needed business scenarios to implement have to be modelled and formalised. Really, there is a complex phase of deployment to make working such architecture.

3.4.1 The modeling phase

In order to usefully exploit interoperability architectures and technologies, it becomes fundamental the proper definition of the business scenario that have to be managed. The modeling activity can follow many different approaches, and can depend considerably by the working perspective, vision and expertise of the modeler. Despite that, three main approaches in the definition of the business model to perform can be identified:

- The top down approach; in this case the more abstract concepts are first defined. This means that the overall business scenario would be depicted, and then, having an idea of the global process, the details of the integration mechanisms (like the structure of the messages) would be faced and bring into focus .
- The bottom-up approach; this is the opposite approach of the top-down one. First of all, every details that regards each endpoint of the integration is faced. Messages and events are defined before the processes that are the final modelling step.
- The abstraction based approach; this methodology is specific for B2B integration.The idea is first to provide a general definition of the enterprise business process, without considering the endpoint characteristics. In this phase the business process is formalized following the

perspective of a single enterprise. The business process is defined considering all of its aspects. Then the endpoints are considered: their characteristics are now included within the business processes defined before. Finally, the needed agreements with the trading partners are fixed.

3.4.2 Further functionalities - Profile and agreement definition, Advertisement and discovery services

Besides the aforementioned functionalities, and the related components needed to provide them, an integration architecture may provide further facilities that are not directly included in the abstract architecture outlined above: in some manner they do not properly belong to the interoperability architecture because they require the involvement of external entities, or , in other words, they require the extension of the architecture out of enterprise boundaries. These facilities are basically:

- Profile definition. This functionality allows trading partners to define an own business profile that describes their capabilities in support business processes and transactions.

  For example, the Collaboration Protocol Profile (CPP), that is part of the ebXML standard, describes the role of the trading partner in a business transaction, some information about it and various specifications for the message exchange. Web Service Description Language (WSDL), a W3C standard, to date represents the interface for Web Services functionalities, and specify input and output parameters for them. In some limited manner, WSDL could represent the profile of the WSs.

- Agreement definition.This functionality allows trading partners to define an agreement upon which business transactions can be established. The agreement includes the specification of transport protocols, security requirements, business processes, roles of the partners and so on. Collaboration Protocol Agreement (CPA) is the part of the ebXML standard devoted to the definition of partner agreements; it is used also to authorise the message exchange.

- Advertisement and discovery services. In order to establish new business relationships, there must be a mechanism to allow partners to meet each other and exchange information about the supported business activities. Advertisement and discovery services are used for this purpose.

  ebXML Registry (ebXML standard) represents the global place of an ebXML architecture where partners can store information for advertisement and discovery. This registry provides some tools to ease the managing of the contained documents (i.e. tools to approve or remove documents, or to better search and match them in the registry).

Universal Description Discovery and Integration (UDDI) has been created by an organisation called UDDI.org [118] in 2000[1] an now it is an OASIS standard(last version is 3.0) from 2002[2]. It is a complex set of specifications that define a "Web-based distributed directory that enables businesses to list themselves on the Internet and discover each other, similar to a traditional phone book's yellow and white pages" [119]. The structure of UDDI consists of a registry accessible via Web Services that contains XML descriptions (using the UDDI syntax) of potential business partners.

3.4.3 Maintenance of the framework: monitoring and improvement of business processes, change management

Monitoring business processes

Once an interoperability framework has been set up and configured, the business processes and business messages can be designed and deployed. From now the execution of the processes must be monitored both to detect errors and faults of the system, and to identify possible bad configured processes that can be streamlined and made more efficient, or critical activities that must be improved. This can lead to redesign the processes in order to have a faster response, less errore prone activities and optimize the use of the resource.

This is very a relevant phase in the life-cycle of the framework since it can result in a performance improving and therefore in economic advantages for the enterprises.

Basically, there are two mechanisms that can be implemented to perform the aforementioned activities:

- Monitoring tools, that can be used to constantly observe and trace the running of the processes and events, and the state of the system.
- Data warehouse tools, that can be used to analyse and to query base of aggregated data and the history of the process execution (that must be kept for the necessary periods of time) in order to extrapolate the performance of the system and to compare this resulting performance with the targeted performance.

Change management

In the business world processes and events are designed to be changed. In fact the business

[1]Copyright © 2000 - 2002 by Accenture, Ariba, Inc., Commerce One, Inc. Fujitsu Limited, Hewlett-Packard Company, i2 Technologies, Inc., Intel Corporation, International Business Machines Corporation, Microsoft Corporation, Oracle Corporation, SAP AG, Sun Microsystems, Inc., and VeriSign, Inc.

[2]Copyright (c) OASIS Open 2002-2004

world and commercial relationships are a very dynamic environment and continuously evolves; an interoperability framework must support this scenario in its changes.

On the other hand, it is not feasible to restructure an integration model from scratch [111], destroying the previous model asking an enterprises to re-built it business processes. In order to avoid such situation, changes must be not only planned in advance when possible, but must be thought to require the minimum change in the whole system. A well-designed architecture should support the management of the changes. An easy management of these changes can result to be fundamental for the good maintenance of the framework.

Changes can concern the basic standards (like exchange protocols, message formats) of the interoperability framework, or the business models implemented upon the standards (endpoint agreements). Surely the former ones are more critical to manage, since they can require the overall restructuring of the software architecture.

The main situation to address maintaining the framework are:

- The appearing of a new standard: continuously new standardisation bodies or initiatives are being set up, each of which with its innovative and resolutive solution.
- The introduction of a new version of a standard, or new product description. In this case must be evaluated the compatibility of the new version and the modifications to implement.
- A new agreements between trading partners.
- A new partner relationship to establish and manage.

Chapter 4

Technologies and research prototypes

In chapter 4 and 5 I will overview the recent developments and the solutions, coming from both the research literature and the standardisation bodies, that have been proposed to address the B2B interoperability problem ([29],[106] and [75] provide extensive surveys both on B2B interaction frameworks and on the research initiatives in the interoperability field). To this aim, three different perspectives can be adopted, all relevant in the overall knowledge of the research field.

These different perspectives consider respectively:

- Basic trends in Information Technologies. This view analysis the main technology results and the novel perspectives in ICT that can be useful to face the global issue related to the interoperability problem.

- Software integration platforms. Applications and research prototypes to enable enterprise integration and business interoperability are now developed by different open consortia, universities or research centers.

- Standardisation initiatives for B2B interoperability. Standards can be divided in two main groups: horizontal standards, that provide the definition of interoperability frameworks that, exploiting the new technology innovations, provide solutions for the main interoperability issues without addressing a specific business domain (and that can be a starting point for the definition of more accurate frameworks); vertical standards, that intend to provide well-defined solutions tailored for specific business domains. The idea is both to exploit technology solutions and to adapt general frameworks to meet specific business interoperability requirements.

This chapter will concern the first two points, while chapter 5 will face the world of standards.

4.1 Basic trends

4.1.1 The XML world

XML [130] is nowadays one of the most popular standard format for text management. Its first recommendation was published on 10th February 1998. Since then, it became more and more considered in almost every application and research field. XML derives from SGML, (ISO 8879), and its strength resides in its simplicity and flexibility that allow to adapt XML to may different use contexts. Born in the electronic publishing research context, it quickly resulted as a very efficient and powerful solution to solve problems of data interchange, also exploiting the potentialities of Internet and the Web; it has been introduced in the B2B context as the business document format and it is now one of the basic brick of almost every interoperability architecture.

XML is the progenitor of a large set of other XML standards based on it: among the more famous and used are: XML Schema, a validation language used to express document structure; XSL, a transformation language for XML document and to build style sheets for data documents; XPath, an expression language used to extract data from XML documents, and many others.

4.1.2 The Semantic Web

As I have outlined before, information integration is one of the basic steps in building interoperability architectures: this means finding a way to share documents and information among the partners that contribute in the production process implemented by a supply chain, or at least in a subsector of it. Such information can be represented using a wide set of data types.

Because of the different data formats, document structures and vocabularies of business terms adopted by each enterprises, usually two enterprises can not communicate each other without a mapping mechanism, or using a way to translate the documents. While this is the normal scenario among different enterprises, perhaps it could seem strange that this situation can occur also within the same enterprise, among different departments, but it is not unusual.

The differences in data formats can be divided in two types: syntactic and semantic differences. While syntactic differences relate to the structure of the data, and the way they are represented, semantic ones relate to the meaning of the data. [54] discusses the problems concerning the managing of message formats and contents in E-business communication: it provides an analysis about the EDI standard [41]. This paper argues EDI limits to solve electronic business communication problems, especially because of it results too cumbersome and not flexible enough to face the requirements of the new emerging market. XML is than pointed out as the basic technology for structuring the information to be exchanged.

On top of XML, a set of related technologies have been proposed to structure and to define well-formed documents. Among these technologies are the specifications of the DTD and XML

Schema syntax to define document schemas and models. But XML and DTD alone cannot define also the meaning of the documents, or, more precisely, cannot define the meaning of the terms that compose a document. Having a DTD, a partner receiving an XML document can only verify its adherence to the DTD, but cannot understand the significance of the document. Obviously this is not a secondary issue.

[113] depicts a future vision where machines can understand the semantic of the documents diffused in the web without human assistance. Starting from this future perspective, and to reach it, a number of initiatives have taken place to define specifications and to implement tools that can enable the so-called ”Semantic Web”. The idea is to express information contained into the (web) documents in a meaningful way accessible to the applications. The Semantic Web thus results as an infrastructure that provides semantic information about documents, on which different applications can be implemented.

The development of the Semantic Web involves many different research areas (knowledge representation, web technologies, knowledge engineering, database, etc.) so identify the Semantic Web with a particular technology is sure a misunderstanding. Nevertheless, it is possible to outline the Semantic Web as a framework composed of different components, providing different functionalities:

- The languages, that are necessary to express the semantic of web content.
- Accessible resources (metadata and ontologies) to describe conventional meaning.
- Application to manage semantic information.

There are many possible application scenarios for the semantic web, and e-commerce is sure one of them [63]. Still nowadays a large part of B2B transactions are realized using non-Internet networks, but this traditional mechanism can’t lead the electronic commerce to the development of its real possibilities and to the exploit of its full capabilities. It will be substituted by new web-based applications for business transactions, which will guarantee much more flexibility and adaptability. B2B Integration presents many open issues related to the semantic aspects of business transactions that are worth to be faced, analysed, studied, and solved [18]. The main problems to overcome in order to implement web-based frameworks for a real open e-commerce are:

- to find and compare different vendors and their offers.
- to exchange different data formats.
- to integrate different business logics and processes.

The Semantic Web intends as a possible solution to resolve some aspects of the interoperability problems and it can bring real advantages for electronic commerce [135]. It can become one of the necessary components of architectures to exchange business messages among different partners.

The concept of the Semantic Web tops mainly on the definition of ontologies: the ontologies extend syntactic interoperability to semantic interoperability by providing a source of shared, well defined and unambiguous terms. The main ontology languages specifically designed for use on the web was in the first instance DAML+OIL (that is the result of a merger between DAML-ONT and OIL [44]) and then its direct successor OWL. OWL exploits XML and RDF W3C standards adding the formal rigor of the description logic [53]. Together with DAML+OIL and OWL, many other languages exist to represent the knowledge. [17] presents a comparison on expressiveness between "traditional" ontology languages and web-based ontology languages. Since ontologies represent the basic mean for structuring and exchange information between different informatic systems, it becomes crucial to design, implement and maintain large and adequate ontologies: [80][1] tackle the problem of enriching ontologies, while [86] faces the problem of merging and aligning different ontologies.

Using ontologies, intelligent software agents can understand the semantic meaning of data and reasoning services (as FaCT or Racer) can resolve some inference problems: for example the logical descriptions provided with the ontologies could be used for automated matchmaking and brokering [51].

[76] argues that electronic commerce can be successful supported by the adoption of ontologies. It points out some companies that have already implemented ontologies; these companies consider the effort to build and maintain ontologies to be balanced by competitive advantages. In this scenario many organizations are arisen to support this trend. This paper claims that studying the use of ontologies for electronic commerce is a fruitful research area that can have a great impact on every person's life. [114][69] investigate how Semantic Web (and Web Service) can support a service description language that can be used to enhance business-to-business interactions and to provide service advertisement and discovery.

4.1.3 The role of Web Services

Web Services represent, on one hand, a future perspective for the development of interoperability web-based solutions that can leverage on the new emerging technologies. In the definition provided by the W3C, a Web Service [126] *"is a software system identified by a URI [RFC 2396], whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols"* In the context of the ecommerce, the Web Services can be used to provide the enterprises with a common way and a framework to define application-to-application services available via Web (again, XML represents the basis to construct and define the syntax - and then also the meaning - of messages to exchange).

Web Service technology defines three different [24], but correlated, necessary functionalities to implement in order to achieve interoperability among web applications. For each of these functionalities, there are specifications that are been developed and released. These functionalities are:

- Communication protocols. The protocols used for service interaction are HTTP (transport protocol), SOAP (packaging protocol), XML-based protocols for messaging and RPC, WS-Coordination (meta-protocol to maintain state between client and service and verify message exchange), WS-transaction (to provide reliability and guarantee transaction properties to any sort of interaction).
- Service description. In order to use a Web Service, a client must know which operations is supported by the service, which messages (in terms of input and output messages) and which data format the client can exchange with the service. Such information is described using WSDL (Web Service Description Languages), that provides the Web service's interface.
- Service discovery. In order to be invoked, a Web Service must be traced on the net by the potential users. UDDI (Universal Description, Discovery and Integration) provides a way to find service providers using a centralized registry of services.

Since the service description requires the unambiguous understanding of the semantic meaning of the description, semantic web languages (as DAML+OIL and OWL) are used to define proper ontologies for Web Services, like DAML-S [28] and OWL-S (directly derived from DAML-S)[33]. These ontologies should be used to describe properties and capabilities of Web Services; in fact, while WSDL describes how to use a Web Service, it can't express what a service does: in this perspective, DAML-S an OWL-S can be considered as the natural complement of WSDL. The structure of OWL-S, that directly derives from DAML-S and should represent the reference ontology for Web Service, is divided in three main parts: service profile (to advertise and to discovery a services), process model (to describe the service) and the grounding (to explain how to interoperate with the service).

Nevertheless the real challenges that Web Services framework are facing is to provide a way not only to advertise a single web service allowing its usage for a specific task, but to provide a way to compose and connect different services together, providing more and more powerful and flexible services [92].

Web service composition can be divided in two main folders:

- orchestration, to define executable business process that needs to collect and communicate with external services. In this case there is one actor that directs the execution of the process.
- choreography, where each partner of the business process can co-operate with the others describing its role within the process. In this case no one truly control the interaction.

There are a number of initiatives to provide XML-based language for Web Service orchestration and choreography:

- WSCI (Web Service Choreography Interface) is a choreography languages that is been introduced in 2002 by Intalio, SAP, and SUN. WSCI proposes a way to extend a WSDL interface definition for Web Services collaborations. The main extensions of WSCI are: order of the operations, exception handling, support for transactions, conversation identifier and time constraints.
- BPEL4WS (supported by Microsoft, IBM, BEA and other vendors) is originated from the convergence of two workflow languages: WSFL and XLANG. Its specification have been released in May 2003: it aims to describe how to coordinate Web Services within a process flow.

Considering Web Services as a framework to define, advertise and connect business processes, making them accessible within the same enterprise or across different ones, and the new efforts to develop service composition languages, it becomes necessary to focus the relationship between Web Services and business process management [72]. In this context it is also possible to compare DAML-S and BPEL4WS that have both broad and complementary objectives [79].

4.1.4 Agent architectures

Agents represent a novel paradigm for system and software programming. There are many definitions of an (intelligent) ”Agent”, but basically an agent is a software system able to act autonomously and flexibly in a changing environment [7]. According to the most common sense an agent is:

- Autonomous: an agent is able to act without the human involvement, and can take specific and contextual decision during its life-cycle.
- Communicative: an agent can communicate with other agents, with users and in general its external environment.
- Responsive: an agent must be able to react to external urges, adjusting to face the evolving situations

- Persistent: an agent maintain its own state and has a specific knowledge that it can use to take decision or to act.
- Mobile: an agent can move through different environments, changing collaboration partners, and facing new situations and issues.

In this scenario FIPA(Foundation for Intelligent Physical Agents)[45] is a relevant initiative (it is a no-profit organisation) to develop standards for interoperation among intelligent agents. Among the most relevant agent platforms are:

- Jade (Java Agent DEvelopment Framework) from Telecom Italia Lab (that also originate the Jade Open Source Community) based on the java language and compliant with FIPA standard;
- Grasshopper: it is a OMG and FIPA compliant platform based on Java and used in several European project
- Aglets, from IBM Japan, based on Java as programming language

4.1.5 Model Driven Standards

The evolution of software development went through different phases: one of the last steps in this process is represented by the model-based technology (named Model Driven Development (MDD) or Model Driven Engineering (MDE)).

This novel paradigm is supported first by the Object Management Group (OMG), that is now moving from the Object Management Architecture vision (OMA) to the Model-Driven Architecture (MDA). This novel approach considers the models as the first step developing software and information systems. The model represents thus the core for the definition, the maintenance and integration of the software. Transformation operations can then be applied to switch between different view of the same model. In this way the life-cycle of the software development is composed of different, successive transformations.

Upon the concept of model-driven development, many other related idea arouse as generative programming, software factories and so on, and many standard have been defined (MOF, XMI, UML, OCL). Nowadays, the main initiatives regarding model-based technologies are:

- OMG's Model Driven Architecture (MDA), that exploit MOF (Meta Object Facility), UML(Unified Modelling Language) and CWM (Common Warehouse Metamodel). There are also a series of Model Interchange standards based around XML (XMI), JMI (Java Metadata Interface), and CMI (Corba Matadata Interface).

- Microsoft's Domain Specific Modelling (DSM), the Microsoft approach to MDD within the architecture of Visual Studio.
- Model Integrated Computing (MIC).

4.1.6 Technologies for Workflow Management

Each activity within an enterprise, whether it is a production or management one, is inserted in an enterprise business process that in general involves different subjects and sectors of the same enterprise. A business process can then be defined as a collection of activities (performed by human personnel or by automatic mechanisms like software systems) to achieve a particular business object. Some examples can be the hiring of new employees or the definition of a purchase order.

These business processes must be represented is some manner. A workflow is a formal executable description of a business process. They are often specified using directed graphs that represent all the interactions that exist within the business process.

Since the workflows describe formally business processes that involve different entities, they can be seen as the programming language for enterprise application integration. The business process specification with a workflow language provides in fact a powerful mechanism to well organise and manage enterprise activities, allowing to have a tool to design a high-level abstraction of all interactions. Workflow management is done using a Workflow Management System (WfMS): a WfMS is a framework and a set of tools to facilitate the definition and the maintenance of the integration logic of an enterprise, and to control how to dispatch information among (human) participants of an administrative process. It also defines the business logic necessary to integrate heterogeneous and distributed systems.

The main benefits of a WfMS are (but obviously they depend on the specific framework used):

- Rapid process design and maintenance.
- Visual interface.
- High availability.
- Failure and exception handling.

Together with these benefits, there are also drawbacks:

- Expensive software licenses.
- Complex installation and management (they are no plug a play applications).

These drawbacks derive from the fact that a WfMS implements a complete middleware platform: a WfMS results thus to be a heavyweight platform, difficult to manage and maintain.

The strength of WfMS consists in the ability to make integration logic explicit and, in some cases, to hide the complexity of this integration logic behind a visual language and sophisticated development interfaces. These characteristics make a WfMS a relevant component in a framework for business integration. On the other hand WfMSs have proved to be most useful with repetitive well-defined processes, which are in many cases already managed using traditional and well-tested middleware. During the early 10 years many initiatives have facing the management workflow problem.

The Workflow Management Coalition (WfMC), founded in August 1993, is an international non-profit organisation devotes to the research and development on workflow topic, and its main aim is to define and to spread standards for interoperability and connectivity between workflow software. See section 5.4.2.

BPMI is another initiative that promotes and develops the use of Business Process Management. It aims, like WfMC, to establish standards for process design, development, management and maintenance. BPMI initiative stems from a non-profit corporation. See section 5.4.6.

Another language for process specification is PSL [73] (Process Specification Language), and its aim is naturally to provide a standard language for the process specification and application integration. PSL project stems from the collaboration between the National Institute of Standards and Technology (NIST) and the industry.

Finally, because of its role in the modelling area, it can be mentioned UML.

WfMC and BPMI are only two of the major protagonists in the scenario of the Enterprise Modelling, that is in truth composed of many other languages and frameworks. In this confused scenario of different approaches in defining workflow management frameworks also the role of Pi-calculus for a suitable formalisation of the workflows is an open question [103][95].

In order to contribute (partially) to resolve the problem of multiple enterprise modeling languages, the European Commission financed the Thematic Network Project (IST-2001-34299) called UEML (Unified Enterprises Modelling Language) [120]. The main object of UEML is to provide industry with a unified and expandable modelling language, which should serve as an interlingua between EM tools and applications (similarly as KIF was designed as an interlingua between different knowledge base management systems). The project is setting up an UEML working group, which activities plan the following steps: creating a European consensus on a common modelling language, building an UEML demonstrator to promote the initiative and to implement the complete UEML.

There are also several commercial products available for those enterprises that aim to manage their business processes using a WfMS. Among them are WebSphere MQ Workflow, from IBM, BEA Weblogic Integration and Microsoft Biztalk Orchestration.

4.2 Some proposed architectures from the literature

Really many research prototypes faced the interoperability issues. [15] presents an overview of some of them. Such prototypes provide different functionalities to enable business integration. This section summaries them exposing their main characteristics.

- CMI (Collaboration Management Infrastructure) provides an architecture to manage inter-enterprises workflows between tightly-coupled trading partners. It is composed of three engines that allow to establish relationships between the partners and to execute instances of concrete business activities (i.e. to execute workflows). Transport protocols and message format must a priori be agreed upon.
- eFLOW provides a mechanism to specify, enact and manage composite services. Such composite services are defined combining basic services or other composite services in a process schema that control the execution of the services. eFlow uses adapters to support services that adopt various B2B interaction protocols.
- WebBIS (WEbBase of Internet-accessible Services) defines a language to compose Web Services. WEbBIS uses ECA (Event-Condition-Event) rules to specify the business logic of the services, and adopts a mechanism to propagate changes of a service to other services that rely on it.
- WISE (Workflow-based Internet SErvices) provides an infrastructure to support process definition, enactment, monitoring and coordination in virtual enterprises.
- CrossFlow uses the contracts as a basic tool for cooperation. Using contracts, the partners can advertise a service in a matchmaking engine. A consumer can search for a service using a contract template via the matchmaking engine. If a match is found in between the provider's contract and consumer's contract, an electronic contract, together with a service enactment infrastructure is set up.
- Mentor-Lite tries to solve the problem of distributed workflow executions: the overall workflow is divided into sub-workflows, each managing specific activities that have to be performed within an organisation.
- SELF-SERV (compoSing wEb accessibLe inFormation and buSiness sERVices) defines a composition language for Web services using state charts, together with peer-to-peer model for their execution. This execution is coordinated by lightweight schedulers.

Many other research prototypes are now exploiting ontologies and semantic web technologies to improve the interoperability of the frameworks. One of the main purpose is to exploit ontology to map data format between heterogeneous systems. These efforts aim to provide the semantic description of the frameworks. A list of the most relevant follows.

1. [82], that is based on an ontology constituted of three layers: the Upper Domain Ontology, the Application Ontology and the Lower Domain Ontology. The interoperability is achieved providing a mapping between the local schema of a system and the common ontology. This mapping is basically constituted by a set of transformation rules.

2. [3] is proposed an interoperability architecture for the exchange of business documents between different enterprises in the Automotive industrial sector. The main idea is to translate the XML Schemas of the exchanged document into a OWL descriptions. Then the different ontologies will be integrated using the RICE reasoner in a unified merged ontology that can then be used as a source for the mapping between the different documents. Both the building of the merged Ontology and the translation of the business documents in the target documents is automatic.

3. In [88] is presented a multi-layers architecture that adopts ontologies as a mean to provide semantic mapping. The architecture is composed of three different layers, the Syntactic Layer, the Data Model Layer and the Ontology Layer. In a very similar way as Harmonise does, the interoperability between different frameworks is achieved thanks to a transformation process that led to a semantic model of the documents (using OWL) and exploiting a unified document ontology.

4. [68] describes more in detail which could be the rules to associate XML Schema constructs to RDF/OWL constructs, in order to build a semantic model of the documents. The transformation is performed via XSLT style-sheets. Once the model of the document is obtained, a mapping ontologies is used to transform the source semantic model in the target semantic model. The authors consider also SWQL as a language to query the ontology.

5. [93] presents a comprehensive ontology based framework (the CREAM framework) to improve semantic interoperability among heterogeneous information systems, allowing to identify semantic correspondances and conflicts in diverse data sources. The outlined framework exploits the definition of a shared ontology and a set of mapping schemas to integrate a set of local schemas. A set of semantic mediators are responsible for the semantic integration of the systems and to provides several services on the integrated data.

Chapter 5

The role of standards in E-commerce

One of the main actor in the definition of a really useful and usable interoperability framework is represented by the world of the standards. This chapter is not intended to be a huge, but however incomplete, list of B2B standards, standardisation bodies or standardisation initiatives; [108], for example, provides just an idea of the wide and dynamic world of standards. On the other hand, I will sketch some main topics of this argument, summarizing the main initiatives in this context, the advantages and the drawbacks that could be highlighted within standardisation processes.

5.1 Defining a standard

5.1.1 What is a standard?

As can be guessed, there are many definitions of what is a 'standard'. [107] can give an idea of the amount of these definitions. On the other hand, I report that one taken from [110]:

"According to BSI, a standard is a published specification that establishes a common language, and contains a technical specification or other precise criteria and is designed to be used consistently, as a rule, a guideline, or a definition".

The definition can be useful to stress the fact that a standard is based in some manner on an agreement between various and different subjects,(that could be also competitors among them self) that need a common platform to communicate, co-operate and so on.

In the context of B2B, the complexity, relevance, and the heterogeneity of the commercial activities require the building of a common interoperability agreement upon which new business relationships could be established and inter-company data exchange can be carried out. These sets of document formats, structures and transmission protocols, usually are called "collaborative framework". A complete definition of a collaborative framework is thus based also on the adoption of a standard.

In the complex scenario of collaborative frameworks (and standards) that characterizes the B2B context, it is possible to distinguish vertical collaborative frameworks (a type of collaborative framework addressed to a specific domain, i.e. an industrial sector) in respect of horizontal ones: Horizontal frameworks (examples are UBL, EAN.UCC,...):

- are not specific to a domain;
- contain basic semantics (and processes), to be extended before the real application;
- assign a relevant role (to design real word applications) to the implementation guides;
- are delivered by large organisations/bodies.

On the other hand, vertical frameworks (examples are Papinet, Rosetta.net,Moda-ML):

- are delimited to a domain;
- are focused, and support a variety of very specific business processes and messages; they usually have a strong data typing;
- are ready to use;
- are, often, delivered by ad hoc consortia, with strong commitment of the final users.

It is to be noted that, from this point of view, ebXML is a META-framework that offer a methodology to establish both kinds of frameworks.

5.2 Advantages and issues with standards

There is a discussion about the capability of the standards, and of the standardisation processes, to really contribute to the development of B2B integration solutions.

In fact, while the availability of standards related to technological aspects, such as message exchanges, is not critical (we can think about the case of SOAP and XML, for example, that are freely available and well recognised), the area of the standards related to the semantic of the data format, that are domain and application dependent, shows the weakness of the standardisation results. Fig. 5.1 represents this situation: the development of a new standard becomes more and more difficult as the standard itself is focused on specific business domains or applications

One of the unsolved problems regards the process of standard definition and implementation: the standardisation process life-cycles are too long if referred to that of products and technologies that they should rule[104]. In other words, at the end of a standardisation process the released specifications may result obsolete when compared with the new enabling technologies: canonical standardisation processes (such as ISO, W3C, OASIS) may last many years, whereas practical

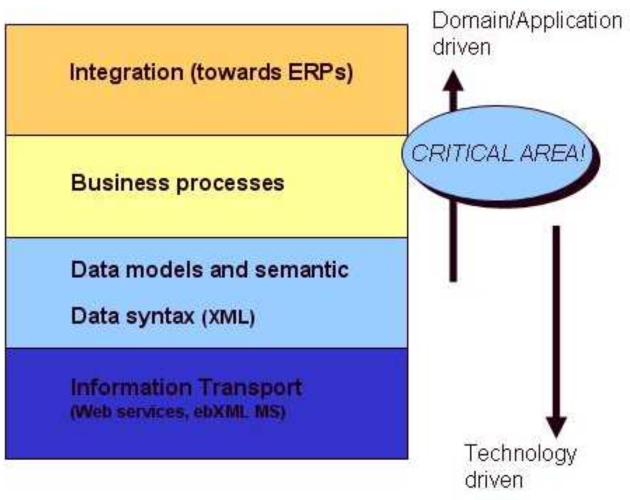


Figure 5.1: The stack of interoperability shows that the critical area for standardisation is that related with the semantic aspects, while the lower layers, related essentially to technology are not so critical.

solutions to face interoperability problems are needed immediately. These delays in the standardisation development, on one hand, damage the widespread adoption of the technologies and, on the other hand, discourage to undertake new standardisation processes for e-business. In the 90s private consortia were setup to overcome the problem but the results are not so significantly different.

Because of these difficulties, both technology suppliers and large final users, by themselves or through enterprise consortia, searched and developed ad hoc solutions based on the establishment of communities. This behavior has led various initiatives to flourish out of the standardisation bodies, and to the birth of more and more abstract standards (or the definition of meta-model or meta-standard) with the objective to prolong the life of the released specifications beyond the life of a single enabling technology.

A second problem is that business documents are complex to define, adopt and maintain, and standardisation life-cycle could be very complex (and furthermore extensions of the standardisations life-cycle should be considered [105]).

In general, in those sectors dominated by few large companies, one (or some) de jure or de facto standard, sooner or later, are defined as the result of an elitish standardisation process that involves the few dominant actors and their technological partners. But in other sectors, characterised by the dominant presence of SMEs, such process is virtually impossible, since it becomes extremely difficult to reach the critical mass of actors needed to set up a standard.

Fig. 5.2 tries to give a brief idea about the different kinds of supply chains. The aim is not to provide a detailed description of the different contexts, but just to put in evidence some fundamental differences.

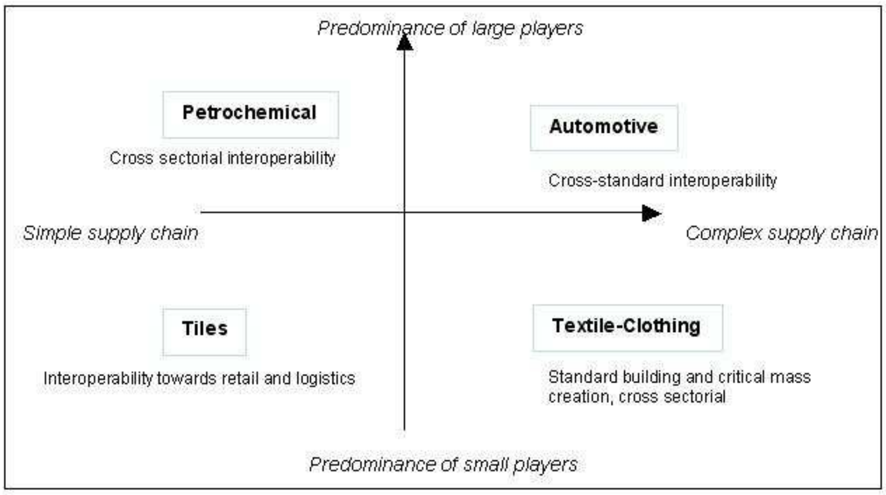

Predominance of large players
Petrochemical
Cross sectorial interoperability
Automotive
Cross-standard interoperability
Simple supply chain
Complex supply chain
Tiles
Interoperability towards retail and logistics
Textile-Clothing
Standard building and critical mass creation, cross sectorial
Predominance of small players

Figure 5.2: A possible representation of some peculiarities of different industrial sectors.

Within such a complex scenario, the continuous research of a trade-off between the completeness and correctness of a standard and its rapid development has raised the proliferation of both horizontal and vertical standards. While the so called horizontal standards belong mainly to the most recognized standardisation bodies (ebXML, UBL[6]), the vertical ones present a more heterogeneous panorama: some enterprise consortia have developed domain-based standards (e.g. RosettaNet, Papinet, Open Travel Alliance); others based their standardisation activities on standardisation bodies (i.e. Eurofer within CEN/ISSS); nevertheless in many other sectors there is not a well-defined agreement on a common vision and, often, the needed critical mass to build a standard is lacking.

5.2.1 Standardisation bodies

This subsection lists the main standardisation bodies. Public standardisation bodies are:

- UN/CEFACT - United Nations Centre for Trade Facilitation and Electronic Business[121]
- ISO - International Organization for Standardization[57]
- CEN - European Committee for Standardization[16]

Main open private consortia;

- W3C - World Wide Web Consortium[123]

- IETF - Internet Engineering Task Force[56]
- OMG - Object Management Group[89]
- OASIS - Organisation for the Advancement of Structured Information Standards[87]
- RosettaNet [100]
- BPMI - Business Process Management Initiative[7]

5.3 Standard overview

This section provides a brief overview of the most widely used B2B standards. It starts listing horizontal standards, and then it reports some relevant vertical standards.

Horizontal standards

5.3.1 EDI - Electronic Data Interchange

EDI is a UN/CEFACT standard born to define a common architecture for application-to-application transfer of business documents in digital format between computers, and to replace the hard-paper based communication. The document format EDI standard is composed of ANSI X12 and UN/EDIFACT standards.

EDI adopts the store and forward transport paradigm (in some manner it implements a peer-to-peer communication): before Internet was constituted, specific networks, called value added network (VANs), were designed to exchange EDI messages. Each partner must subscribe to a VAN to have a mailbox, and than can send messages to every other partner participating in a VAN. All the VANs are interconnected.

With the expansion of Internet IETF introduced EDIINT (EDI over the Internet), that allows to exchange EDI messages without the use of the VAN (thus avoiding the fee payment for the VAN). EDIINT is a true peer-to-peer system, without a storage medium, and, because of this peer-to-peer structure, it cannot provide those typical functionalities (e.g. authentication, authorisation, tracing, and other) of a VAN.

EDI standard is focused on document definition and transport mechanism. The original syntax of EDI messages was not based on XML, but prefigures the use of well-defined characters to structure the documents. An example of an EDI message is

```
ISA˜03˜0007777777˜01˜MYPASSWORD˜32˜007777777 ˜01˜614553816T\
GS˜TF˜MS501312˜614553816T˜20040420˜1015˜1010˜X˜004030\
ST˜813˜1101\
BTI˜T6˜050˜47˜MS˜20040420˜TRAV˜24˜007777777˜49˜12345678˜˜˜00\
DTM˜194˜20040331\
TIA˜5000˜˜1.0\
N1˜TP˜TRAVEL CENTERS\
N3˜755 FREDERICK AVE\
```

EDI has been extended towards the XML format [131]. An example of an XML version of an EDI message follows.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<?xml-stylesheet href="edi-lite.xsl" type="text/xsl">
<!DOCTYPE Book-Order SYSTEM "edi-lite.dtd">
<Book-Order Supplier="4012345000951" Send-to="mailto:orders@sgml.u-net.com">
  <Title>EDItEUR lite-EDI Book Ordering </Title>
    <Order-No>967634</Order-No>
    <Message-Date>19990308</Message-Date>
    <Buyer-EAN>5412345000176</Buyer-EAN>
    <Order-line Reference-No="0528835">
     <ISBN>0201403943</ISBN>
     <Author-Title>Bryan, Martin/SGML and HTML Explained</Author-Title>
     <Quantity>1</Quantity>
   </Order-line>
</Book-Order>
```

Another Internet-based EDI initiative is OBI (OpenBuy): OBI is a standard that aims to complement EDI; especially it concerns high-volume, low-dollar amount transactions. OBI adopts HTTP as transport protocol, ANSI X12 EDI for payload definition and SSL over HTTP for securing communication (it also manages digital signatures and digital certificates). Basically, OBI represents a step forward in terms of scalability, adaptability, and lower entry cost for the new users. OBI also provides a (very) simple protocol to manage a set of purchase activities in the business layer.

5.3.2 Workflow Management Coalition

The Workflow Management Coalition (WfMC) [124], founded in August 1993, is an international non-profit organisation composed of over 300 members that embraces software vendors, research groups, universities, and customers.

The research topic of the WfMC are the business processes and their executions, monitoring and interoperability, and the adoption and enhancement of workflow technologies.

The basic result consists in the definition of the Workflow Reference Model [52], a description of a workflow system architecture that attempts to construct an abstract view of the core characteristics of the business processes, separated from the technologies.

Its main aspects can be divided in three folders:

- A common vocabulary of terms to describe business processes.
- A functional description of the basic software components and the way they should interact within the WfMS.
- The definition of five interfaces to enable software components to communicate and exchange information using standardised mechanisms. This allows interoperability between different products.

One of the components of this architecture is XPDL (XML Process Definition Language), that is an XML language to describe the processes to execute by a workflow engine. This model has been adopted within some standardisation initiatives.

5.3.3 eCO

eCo is the e-commerce project from CommerceNet , that is ”a not-for-profit global community of leaders with a ten-year history of success. With a focus on understanding strategic information technology, CommerceNet helps companies improve business performance through the adoption of shared, interoperable business services.” [21]. CommerceNet was founded in 1994 by Dr. Jay M. Tenenbaum, with the aim to build on top of Internet an ”open network of businesses”, thus fostering the worldwide business collaboration.

eCO adopts basically a document-based approach for B2B interaction: it defines a generic framework composed of a set of core business documents that represent general-common information exchanged during business transactions and that can concern different application domains. In this manner eCO does not target vertical industry domains, but provides an horizontal interoperability framework.

eCO provides, regarding the content layer, xCBL 4.0 (XML Common Business Library) [129] that is the set of XML core documents that represent common interactions. eCO documents may be extended by business partners to solve more specific issues, related to a specific domain.

eCO does not provide a language to manage workflows: on the other hand, it uses xCBL also to provide business service descriptions. These descriptions are named Business Interface Definition (BID) and specify the business documents that can be exchanged by a business service; BID are also used to advertise business services. In this manner xCBL is thus used to describe both the content of the messages and the interfaces of the processes.

eCO separates the description of the processes from the process implementation. Using eCO, providing a new service meas basically to define a new interface, and to bind the interface with internal applications. Anyway eCO can not address the semantic heterogeneity of service descriptions, especially considering the wide number of E-commerce applications. Security mechanisms are optional for eCO.

5.3.4 ebXML

ebXML [38](Electronic Business using XML) is a set of specifications from UN/CEFACT and OASIS that defines a collaboration framework over the Internet to enhance interoperability between enterprises. The ebXML initiative started in 1999. ebXML framework comprises:

- the Message Service(ebMS), that provides a communication-protocol neutral, reliable and secure mechanism to exchange business messages. This specification defines the message enveloping and the headers (i.e. the message packaging) to use to transfer ebXML messages over a communication protocol. To this aim ebXML adopts and extends SOAP. An ebXML message can thus be delivered using both FTP, HTTP or SMTP as transport protocol.

- a set of standardised document schemas to manage the business logic: the Business Process Specification Schema, that is a process definition schema to define (public) business processes; the Collaboration Protocol Profile, that allows trading partners to specify the business processes they support; the Collaboration Protocol Agreement, that can be used to express an agreement upon business process execution between two or more trading partners that want to establish business collaborations. Differently from RosettaNet, business documents used during the transactions are specified outside the documents that describe the business process .

- a set of components upon which the exchanged business documents can be built. There are three types of components: core components (re-usable across different domains), domain components and business components (domain-specific components provided by sectoral industry). In practical, ebXML has provided only a poor support in the definition of these components. To overwhelm this limitation, the UBL initiative was born in mid-1999 (see section 5.3.7).

- a repository that is used to store trading partner information, profiles and product descriptions.

5.3.5 cXML

cXML (Commerce XML) [26] is a standardisation initiative born on 1999 with the cooperation of diverse companies (among which, for example, is ARIBA) to develop an XML business protocol for the exchanging of business documents. cXML includes ”documents for setup (company details and transaction profiles), catalogue content, application integration, original, change and delete purchase orders and responses to all of these requests, order confirmation and ship notice documents (cXML analogues of EDI 855 and 856 transactions) and new invoice documents.”

cXML does not cover all of the expected business transaction, but could be easily expanded. It adopts XML syntax and tries to overwhelm those limitations typical of the EDI frameworks, adopting a different angle in faced document definition. cXML specifications are composed by a set of DTDs and documentation for their usage available for free on the cXML website.

5.3.6 BPMI

BPMI (Business Process Management Initiative)[7] is a no-profit organisation founded in August 2000 with the aim to enhance business process management across enterprise boundaries in every industrial domain; BPMI is composed of a large set of members among which are IBM, Adobe, Intalio, Bea and many others. It releases XML-based standards. The main open specifications developed by BPMI are:

- BPML - Business Process Modeling Language. This is a meta-language for the modeling of business processes.
- BPMN - Business Process Modeling Notation. This specification defines a graphical notation to describe business processes. As it is claimed on the website, ”The primary goal of the BPMN effort was to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes.”
- BPQL - Business Process Query Language. This language is used to allow querying operations on business process instances and verify their execution state. This language tops on SOAP.

All these specifications concern specific aspects of a general Business Process Management Systems (BPMS).

5.3.7 UBL

UBL - Universal Business Language [117], aims to define a royalty-free library of standard electronic XML business documents (i.e. purchase order or invoice). These documents should represent the business language for commercial partners that intend to exchange business information in different commercial domains.

Since it is impossible to specify business documents tailored for every industrial sector, UBL defines a general vocabulary and a limited set of general documents representing common information, and a mechanism to extend both the vocabulary and the documents for well-specific needs. Thus, UBL provides:

- a library of reusable components to build business documents. UBL is the first standard implementation of ebXML Core Components.
- a small set of XML schemas that specify the structure of the most common business documents (28 documents in UBL 2.0).
- a mechanism to extend and customize UBL for a particular business domain.

The UBL initiative originated in efforts beginning in mid-1999 to create a set of standard XML ”office documents” within OASIS. The work of the OASIS OfficeDoc TC was set aside when OASIS and UN/CEFACT began collaboration on ebXML in December 1999. Interest in the creation of a standard XML syntax for basic commercial documents revived again in May 2000 with the decision in ebXML to omit a standard XML ”payload” syntax from the initial set of ebXML deliverables. The working group that came to be known as UBL began in April 2001 as a discussion group sponsored by CommerceNet and was established as an OASIS Technical Committee in November 2001.

UBL completes the ebXML framework providing the components that specify the standard formats for the business documents (while ebXML describes the business architecture to exchange these documents).

Vertical standards

There are many initiatives to provide vertical standards; they constitute a very heterogeneous scenario, more than horizontal standards; however, in general vertical standards provide support only for the definition of B2B protocols (i.e. they define only the document structure to exchange); Rosetta Net and Swift represent in this scenario two special, distinct cases because they provide also the necessary functionalities for the message exchange. In the following I describe RosettaNet (that is based on XML) and SWIFT. Many others, (like Papinet, Open Travel Alliance, Eurofer and so on) are not addressed in this survey. I describe also STEP, a relevant ISO standard to describe technical data of the products.

5.3.8 RosettaNet

RosettaNet [100] is a no-profit consortium founded in 1998 to develop standards for the IT (Information Technology), electronic components and semiconductor manufacturing supply chain and industries. The RosettaNet architecture is basically constituted of three main components:

- The dictionary, that is divided into the RosettaNet Technical Business Dictionary (RNTD), that provides a set of common properties that can be used to define products for RosettaNet Partner Interface Processes (PIP), and the RosettaNet Business Dictionary that contains terms to describe business properties.
- A set of well-defined domain-specific processes, that RosettaNet partners have to implement engaging business transactions. These public processes are called Partner Interface Processes (PIPs) and are specified using UML diagrams that explain the sequence of message exchange. Such use of the PIP avoids the need of an agreement mechanism to contract the business process specifications. Each PIP specifies both the business process, the roles of the partners and the documents (and their structures) exchanged during the process. The exchanged documents are built on top of the RosettaNet Dictionary.
- The RosettaNet Implementation Framework (RNIF), that specifies the packaging and transport mechanisms: its defines the envelop format (independent from the transport protocol) to be used exchanging specific business messages. Common Internet transport protocols are used to communicate over the net.

5.3.9 SWIFT

SWIFT [112](the acronym stays for ”Society for Worldwide Interbank Financial Telecommunication”) is a financial industry-owned co-operative born in the 1973 from an initiative of 239 banks of 15 countries that wanted to solve mainly the security and automatization problems of their communication.

Nowadays SWIFT community includes 7600 members (banks, broker/dealers, investment managers and so on) of about 200 countries. Its aim remains to provide both standardised business message definitions, functionalities, and interface software for secure message exchange in the financial community. In this way SWIFT provides all the necessary instruments to its users to adopt the standard and to enable interoperability between its members, their market infrastructures and their end-user communities. This standard addresses several aspects of the financial world: payments and cash management, treasury and derivatives, trade services and others.

With a more technical definition, we can consider SWIFT as a network of banks and financial institutions. SWIFT provides several products and services:

- Document message definition: quite obviously, SWIFT started designing, deploying and using a non-XML syntax to structure the messages (the last generation are FIN based standards), but it has now recognized XML as the new corner brick to build efficient interoperability solutions and the starting point to move towards open standards; then, SWIFT is now proposing XML-based version of its standard. SWIFT adopts a well-specific modelling methodology to define business messages.
- Connectivity services: the SWIFT's secure IP network (SIPN) and SWIFTNet Link (the SWIFT's mandatory software product for users of SWIFTNet services)
- A set of four messaging services: SWIFTNet FIN (the core store-and-forward messaging services), file exchange (SWIFTNet FileAct), structured message exchange (SWIFTNet InterAct)and browser based messaging(SWIFTNet Browse).
- A range of interface products to connect applications with SWIFTNet.

5.3.10 STEP

STEP is the acronym for ”Standard for Product Model Data”[109].The STEP project was initiated in 1984 by the ISO (International Standards Organization). Its main aim is to provide a comprehensive standard (the ISO 10303 standard) to describe how to represent and exchange digital product information and to produce one International Standard for all aspects of technical product data. In particular, the objectives were :

- Create a single international standard
- Implement this standard in industry
- Standardize a mechanism for describing data throughout enterprise life cycle.
- Separate data description from implementation to facilitate;
- Neutral file exchange
- Shared product databases
- Long term archiving

More in general, the ultimate goal is for STEP to cover the entire life cycle, from conceptual design to final disposal, for all kinds of products. The most tangible advantage of STEP to users today is the ability to exchange design data as solid models and assemblies of solid models.

Nearly every major CAD/CAM system now contains a module to read and write data defined by one of the STEP Application Protocols (AP's). These systems are:

- CAD (Computer Aided Design)
- CAE (Computer Aided Engineering Analysis)
- COM (Computer Aided Manufacturing)
- CNC (Computer Numerical Control)

STEP provides different Application Protocols (AP) built on the same set of Integrate Resources (IR's), that provides a shared definition for basic information. For example, AP-203 and AP-214 use the same definitions for three dimensional geometry, assembly data and basic product information (the AP-203 ”Configuration Controlled Design” is the most implemented protocol in USA, while AP-214 ”Core Data for Automotive Mechanical Design Processes” is the corresponding European version).

5.4 Semantic definition of the standand

The standardisation initiatives listed above provide the definition of documents, messages and data formats to use in B2B data exchange. These definitions are associated with the explanation of their semantics. While the data formats are formally defined, the semantics are in general expressed using guidelines and documentation written for humans. This documentation can comprises hundred and hundred pages and doesn't use a formal language. The semantics is then hardly computer-readable.

The earlier business standards, like eCO, provide basically only a set of commented xsd files, that represent the B2B protocol, together with a rdf or doc documentation that explains how to use the proposed documents. In such case there isn't a well-defined mechanism for the modellisation of the protocols, and this deficiency negatively impacts on the formalisation and usability of the protocols. While this kind of documentation could be somehow useful for a human (but in many case it results to be really cumbersome and difficult to understand for the target users), it is obviously non useful for automatic applications. The use of spreadsheets to add information on the structure of the documents doesn't solve the problem: in general they are used to represent in other way the syntax of the data format or business documents, but doesn't represent a clear formalisation of the semantics.

The same argomentation can be applied also to cXML that provides, for documentation purposes, XML instances as examples, and graphical representations of the defined schemas for the protocols. EDI is similar to xCML and XCBL, since provides the description of the syntax of the messages, togheter with a set of huge documentation to explain the use and the meaning of the structure of the messages.

The definition of the data models in STEP is performed with the EXPRESS language, that is the fundamental method to describe information models. In this sense it plays the same roles of XMLSchema. The EXPRESS language is used to model ”schema” that represent products. STEP uses also EXPRESS-G to provide formal diagrams that represent the data formats designed with EXPRESS. In any case, EXPRESS is again related with the structure of a message, not with its semantics, or with the concepts and the relationships of the knowledge domain related with the standard.

ebXML defines the semantics of its components in two ways: it defines the business processes with the BPSS, and uses the Core Components modelling approach to build business vocabulary. Being a meta-model for interoperability architecture, this is an abstract approach that must be implemented (in SWIFT or UBL for example).

The SWIFT methodology to develop standards is now adopting the XML format. Basically speaking, SWIFT adopts the ebXML view, that consists in the definition of a common repository of ”business objects” that represent the basic building blocks to use defining the data format. The definition of the data format follows a specific development process for the standards: start from a business case, a team of business expert identifies the business information and the scenario to model (mainly using UML notation) and than defines the data models to adopts. These data models are stored in central repository, that represents a core vocabulary. SWIFT uses wo different syntax for the messages, one of which is an XML syntax. To summarise SWIFT proposes a set of data format togheter with a wide human-readable documentation that describes deeply and verbosely the content of each message, and a vocabulary that lists the defined objects. In any case, there isn’t a formal definition of the semantics of the protocol that is related with the business messages/documents. The RosettaNet approach is similar to the SWIFT one: Rosetta NET provides simple vocabularies that are documented with pdf or with spreadsheets. The two RosettaNET vocabularies are really huge and detailed, but represents only a list of term used by the data format specified in the PIPs (that include the DTD for the specification of the structure of the documents).

In UBL the definition of the data types starts from a modellisation phase performed with UML diagrams, and with the definition of spreadsheets that represent the data formats. After this modellisation, the final xsd files, that represent both the reusable components and the standard documents, are built. The standard is released with set of documentation in HTML and pdf format, and rules for customization purpose. Moreover, the modellisation isn’t intended to be used by automatic application to perform data integration.

In order to formalise the semantics of business vocabulary BPMI adopts the SBVR (Semantics of Business Vocabulary and Business Rules), that is used to model vocabularies and set of business rules; SBVR defines a set of terms in which a business vocabulary should be expressed in.

This modellisation is the starting point to export the defined vocabularies in a machine-readable format like XMI.

Part II

Definition of a B2B interoperability architecture

Chapter 6

The business scenario of SME

The integration between systems and organisations of independent firms is a key factor in the modern competition. The objective to create an enhanced, knowledge based economy requires a seamless integration of co-operating firms.

The Information and Communication technologies have a relevant role to achieve this objective but it is clear that technology solutions are strictly interlaced with the peculiarities (and capabilities) of the real systems of firms.

During my work I have faced the problem of business interoperability, especially in a scenario characterised by a large presence of SMEs (Small and Medium Enterprises). This chapter aims to sketch the business scenario of SME, in order to finally draw some conclusions in terms of requirements and technological solutions.

6.1 Peculiarities of the SME scenario

New solutions for the B2B electronic commerce that are being developed could play an important role for the Small-Medium Enterprises, that try to fill the gap with the large enterprises (fig. 6.1 from [37]).

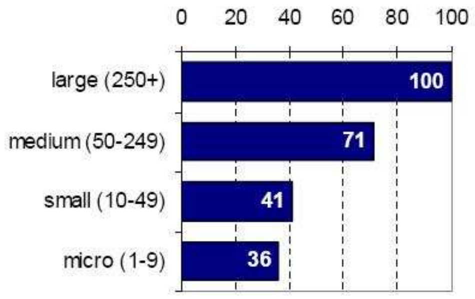


Figure 6.1: % of firms within a size-band the use some E-business solution.

On the other hands, it is important to identify particular strategic specificities about this production sector that require an improvement of the co-operation among the various actors of the production process. These specificities are:

- Heterogeneity in the supply-chain; as for the Textile-Clothing one, these sectors are composed of a large set of different and heterogeneous small and medium enterprises (SME), each of which are necessary to add high specialization and productive flexibility to the whole system. Differences consist not only in the role played inside the supply-chain, but also in technical infrastructure, informatic system, human knowledge, economic means available.
- Responsiveness of the supply chain to the market, following the fluctuation in fashion and season, and thus in the requested products. The need to produce a quick reaction to new and dynamic customer exigencies and to deliver rapidly the final products require a efficient way to synchronise every single component to minimize time lost and achieve a good response to the market demand.

The aim should remain to improve interoperability among independent industrial and commercial organisations by the definition of a common interchange language for expressing business messages in, an exchange protocol to deliver efficiently these messages among the various actors and a set of tools to allow a simple management and a constant maintenance of the established collaboration in order to update the system in accordance with the requirements of the market.

In order to produce a successful solution we consider to be important the following aspects:

- The requirements expressed by the enterprises, which the solutions refer to; we have to bear in mind that the proposed solution must fit the existing internal enterprise information system. It is not feasible to re-engineering the production processes, starting from scratch. Although it is evident as the new ICT and the electronic B2B are becoming essential in order to face the global competition, nevertheless the evolutionary approach, for both economical and practical reasons, has to be preferred to the revolutionary one [111].
- The computerization level of the enterprises and the technologies adopted for the internal data management; in order to obtain the consensus from the whole community the proposed solution must fit both the different informatization levels of their enterprises (that range from small firms with very poor information systems to more skilled companies with modern ERP systems) and the different economic and human resources available to adopt the new technologies.
- The number of reference standards defined from relevant standardisation bodies; these standards concern Internet communication protocols, message exchange formats among firms (SOAP, ebXML) and the languages available for creating documents and messages (XML).

- Being open-source and free; this is surely a good starting point to encourage the adoption of a solution, especially when the target is the SMEs and their technology providers that usually haven't got many resources available; flexibility is another aspect to consider in order to win support from ERP systems, legacy systems and SME.

6.1.1 The role of the standards for SME

A relevant issue to consider is that sectors characterised by the absence of market leaders and by a predominance of SMEs can tackle the construction of a common B2B community only through the adoption of commonly accepted standards [71]. This basically means that an initiative that aims to build a interoperability framework must necessarily give life to (and persist in) a standardisation process to create agreement about the developed framework. In this perspective it is worth to understand the hampering factors to the definition and adoption of standards.

The critical aspects about standards and standardisation processes related to the applicative, or semantic, level of interoperability [61] [12] can be resumed as follows:

- Time: the life-cycles in the standardization processes are too long if referred to that of products and technologies that they should rule [104][105];
- Resources: the extent of human and economical resources does not allow SMEs to participate or influence these processes;
- Usability: the specifications have poor usability (addressed to few expert readers);
- Adoption: the integration of the specifications with legacy systems and ERPs is difficult with low investments and technological skills; an incremental approach is needed.
- Implementation complexity: the complexity of the software to implement the specifications may become an obstacle to the integration with legacy systems and ERPs; this is highly relevant when the technology suppliers of the SMEs are SMEs in turn;

The methods used to face and solve each of these points could determine the success or the failure of the proposed solution. In the following we discuss more thoroughly such considerations.

Any new interoperability architecture must by definition, work with existing internal enterprise information systems. Usually for economic and practical reasons, there is little willingness to change these back-end application systems because this change could require the redefinition of all the stages of the production process.

Focusing on a narrow domain may help to overcome, or reduce, these difficulties; many vertical standardisation initiatives have been setup (for computer industry, automotive industry, paper industry, traveling, finance, etc.) to complement the horizontal frameworks (such as UBL, EAN.UCC, xCBL, etc) that address the need for a general and systematic vision of the B2B collaboration.

Recently, the ebXML initiative has established a new point of view defining a complete and systematic approach to the creation of inter-company collaborations with the aim to exploit the potentiality of XML as well as the legacy of experiences of the EDI community.

Since ebXML offers a meta-architecture and a methodology to establish domain specific collaborative frameworks, I consider to implement a sectorial framework through some of the ebXML architectural components and to deliver sectorial standard specifications through the participation to a standardisation initiative.

6.2 Requirements for a solution

From the analysis of the target sector it is possible to identify the basic requirements of an interoperability framework that aims to solve the interoperability issue of the enterprises. These requirements can also represent evaluation dimensions for interoperability framework as in [75]. These requirements are:

- Usability of the framework. This requirement refers to the degree of easiness with which the users can work with the framework. It embraces several aspects:
  - simple installation and configuration.
  - easy integration with back-end application systems.
  - ease to understand and to use for the users.
  - customisation of the functionalities.
- Modularity of the framework. This means basically that the functionalities of the overall framework must be provided by light-weight and independent modules. This allows
  - to easily adapt the framework to solve enterprise specificity, adding the needed components.
  - to have an incremental approach in the adoption of the framework, without upsetting the organizational structure of the enterprises.
  - to avoid cumbersome installation and adoption of useless components or functionalities.

  Modularity refers both to software components, documents and processes.
- Scalability, both in terms of number of enterprises that use the framework, and in terms of heterogeneity of the enterprise systems involved.

- Loosely coupling among the partners. The framework should not impose a strong dependence among the enterprises, especially in a sector where relationships are in general transient and very dynamic, can last short time, and must be set up rapidly. This involves also a low external visibility of the partners, that acts in some manner as black box. The enterprises can not be monitored by the partners.
- Easy to maintain and to upgrade. An interoperability framework, especially in the business sector, is a dynamic entities, and its specifications evolve continuously, and many versions can be released. This aspect allows an efficient and fast development of the needed standards, and an easy dissemination of the results among the target users.
- Secure and private. Obviously, business data must be protected by unsought accesses; there are several aspects related to security issues.
- Easy to interface with other interoperability framework. Often, there are several competing initiatives in the field of business interoperability, coming from national standardisation bodies, private consortia, private software industry, local districts of firms. Each of them can provide a specific vision of the interoperability problem, and consequently a specific solution for it. In this scenario, a solution should not claim to be the only right response for the enterprises requirements, and should not work in isolation. These frameworks must thus consider to build simple interface for connection with other frameworks.

Chapter 7

An interoperability framework for the Textile/Clothing sector

During my work I contribute in the the definition and development of a interoperability framework for the Textile/Sector. In this chapter I will describe the framework, outlining its components and its structure.

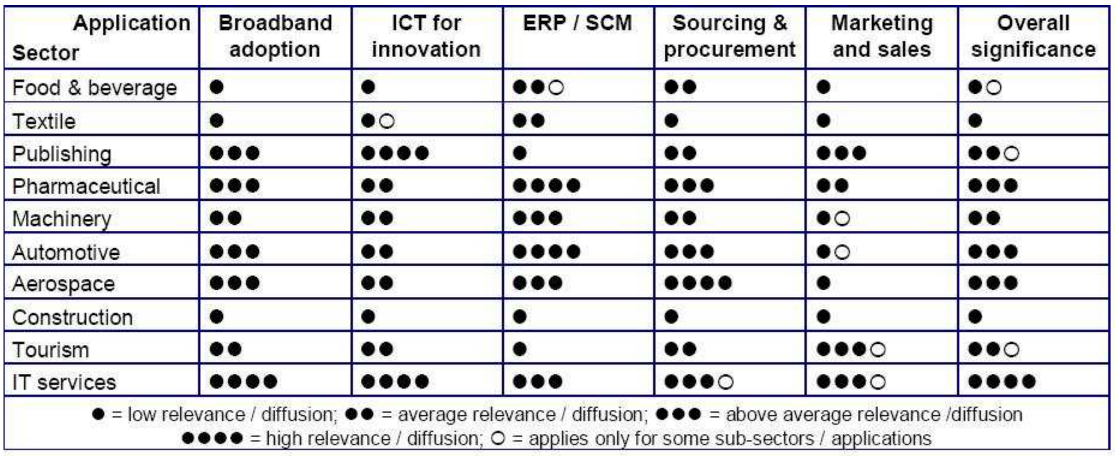
7.1 A brief view of the target sector

The production process within the Textile/Clothing sector is based on collaboration between a large number of small and medium sized enterprises (SMEs) to create and delivery items of fabric and clothing. Each of these enterprises is responsible for a particular aspect of the production process: such co-operation is regulated by the exchange of request/response messages necessary to carry out all the steps of which the supply chain is composed. The delivery timing of the final product is affected by the communication mechanism adopted within the supply chain. In the face of global competition, responsiveness is the key to success in this sector, especially because of the impact of fashion fluctuations on the sector. While most organisations use the Internet today, there are still difficulties on the adoption of new collaborative processes because of different attitudes.

This sector, and also in other sectors related to the fashion industry, also standardisation is an harder task with respect to other production processes, since the sophistication and specificity of the cooperation among the enterprises of the supply chain (mainly based on human relationships instead of Information and Communication Technology means) are very high and represent a peculiar competitive factor. Thus B2B integration is needed but the tools (recognized standards) to achieve this objective cannot be developed and introduced in the sector.

Up to the 2000 (at the start of the Moda-ML project), there was very limited experience of automating B2B processes. Most of the innovation was aimed at automating internal business processes. In many cases, even this was usually only within individual departments, leaving inter-enterprises processes to manual or at best semi-automatic management systems. The only existing interoperability solutions were based on EDIFACT technology, and EDITEX, a subset tailored for the Textile/Clothing supply chain. The inflexibility of these standards has led to solutions that could not be easily adapted to the specific needs of organisations.

The eBusiness Watch report on B2B addressed to the T/C sector [35][36] witnesses the difficulty of EDI based technologies in the sector: few installations, regarding only large companies and addressed to the relationships with large retail organisations rather than with suppliers and subcontractors; the table 7.1 (taken from [37]) summarized the limited adoption of E-business in the textile sector, also in respect other production sectors.



Application Sector	Broadband adoption	ICT for innovation	ERP / SCM	Sourcing & procurement	Marketing and sales	Overall significance
Food & beverage	●	●	●●○	●●	●	●○
Textile	●	●○	●●	●	●	●
Publishing	●●●	●●●●	●	●●	●●●	●●○
Pharmaceutical	●●●	●●	●●●●	●●●	●●	●●●
Machinery	●●	●●	●●●	●●	●○	●●
Automotive	●●●	●●	●●●●	●●●	●○	●●●
Aerospace	●●●	●●	●●●	●●●●	●	●●●
Construction	●	●	●	●	●	●
Tourism	●●	●●	●	●●	●●●○	●●○
IT services	●●●●	●●●●	●●●	●●●○	●●●○	●●●●

● = low relevance / diffusion; ●● = average relevance / diffusion; ●●● = above average relevance /diffusion
●●●● = high relevance / diffusion; ○ = applies only for some sub-sectors / applications

Figure 7.1: Relevance of E-business in different production sectors

The EDI (see also 5.3.1), Electronic Data Interchange, defined by an ONU Commission, was focused on the simplification of the international commerce. It was based on:

- a document structure for the international electronic commerce, already known as UN-Layouts keys (they were conceived for the hard paper communications, before the diffusion of information tools inside the firms)
- the ONU dictionary of the words for the international commerce (UN/TDED) and the recommended codifications (Terms of Payment, INCOTERMS)
- the ISO syntax for the electronic transfer of data within flat-files (ISO 9735).

The result was a complex and rigid technology, EDIFACT (EDI For Administration, Commerce and Transport [60]), where the customizations (EDIFACT subsets) were obtained with the suppression of not used parts of a common general structure that was (in theory) the ”least common multiple” of all the sectorial needs. This approach was valuable because conveys an ”universal” standard, but its application found a great diffusion only in some market sector because of its adoption was technically and economically justifiable only for large organisations and great amounts of data; however it represented a ”cultural fracture” for the business world that was still unprepared.

In the period spanning ’80s and ’90s, the TEDIS project (namely EDITEX [42]) developed the EDIFACT subsets for the European T/C industry. Despite the efforts, the diffusion of the EDITEX solution was bounded to few very large organisations. The take over of the Internet dampened the growth capacity of this experience, with-out offering a final solution to the enormous problem that the EDIFACT was facing. The problem has been tackled again with the diffusion of XML: tools to manage data structures and specifications (like XML Schema) have been developed in order to exploit the flexibility and human readability of the XML documents (a syntax much more flexible and quick than ISO 9735).

Internet has simplify the interoperability problem between different transmission networks (in EDIFACT the approach was typically based on private value added networks) and offers a capillary widespread network infrastructure that makes XML and Internet a winning scheme. On the other hand, the nature of ”metalanguage”, or ”metastandard”, of XML led to a sort of ”linguistic relativity”, because everybody has, now, the tools to build infinite semantic representations compliant with XML; this encouraged the ’do it by yourself’.

In this context the proprietary solutions, based on the ASP (Application Service Provisioning) architectures, like enterprise portals and Internet Integration Services, have known a wide diffusion and are directly competing with the Peer-to-Peer model of information exchange of EDI. Nevertheless they show evident limitations when facing complex networks of relationships that cannot be reduced to the hub-spoke model. This is the reason for a renewed interest in the so called XML/EDI paradigm that is based on standardised messages used in the Peer-to-Peer (P2P) architecture.

In this scenario, can be notable how, and under which circumstances, the experience of Moda-ML in defining, implementing and deploying an interoperability framework contributed to the raise definition of an European sectorial (pre-normative) standardisation initiative. The EDITEX unsuccessful experience led to prefer a more user driven approach and this is the beginning of the history of the Moda-ML project.

7.2 The Moda-ML project essentials

The Moda-ML (Middleware tOols and Documents to enhAnce the Textile/Clothing supply chain through xML)project started in April 2001 with the aim to define an interoperability framework for the Textile/Clothing sector, addressed to small and medium enterprises as well as to large enterprises; the long term objective was to contribute in establishing an European standard for data exchange in the T/C sector. The Moda-ML project ended in April 2003. At the beginning Moda-ML was a project and collected various research organisations (ENEA, Politecnico di Milano, Domina, Gruppo SOI, Institut Francais Textil Habillement - IFTH, University of Bologna) together with a representative set of leading Italian Tex-tile/Clothing manufacturers. It has been supported by the Fifth Framework pro-gramme of the European Commission within the IST (Information Society Technology) initiative (more information can be found in http://www.moda-ml.org) and took part in the cluster of projects about Agents and Middleware Technologies (EUTIST-AMI IST-2000-28221, www.eutist-ami.org).

Pressed between the need for common standard messages and the intrinsic difficulties in the standardisation processes the project adopts the following guidelines in the approach for the construction of a collaborative framework (a detailed description is in [47][30]):

- User driven: a bottom-up approach involving relevant actors since the beginning in order to have a wide consensus from the whole community; the process and transaction analysis and the modelling of the semantic related to the information to exchange (and the related codes) defined within the EDITEX experience were recovered and reinterpreted. The same happened, parallelly, in other national based initiatives in France (eTeXML) and Germany (e-Visit).

- Sectorial (vertical): focused on a well defined and narrow domain but compliant with an horizontal framework (ebXML) in order to assure the scalability;

- Dictionary centric rather than document centric: a dictionary allows to focus on the set of terms that are reused in many document templates with an improvement of the time to deliver usable results [30].

- Iterative: the starting point is a core of inter-company transactions; they must be analysed and implemented with the support of a group of industries and then proposed in a standardisation workshop; then further transactions (and business processes) have to be added to consolidate and extend the effectiveness of the framework and are the input for further standardisation initiatives (more details about the methodological aspects in [47]); it is worth to note that an open standardisation methodology, like the CEN/ISSS Workshop, might be a key enabling factor for the success of this strategy.

- Open specifications and open-source and free software implementation (not proprietary). This is surely a key point when aiming at encouraging a wide adoption to a large number of small and medium enterprises without large financial resources.

One of the first choices for the Moda-ML project was to employ a Peer-to-Peer architecture to allow each firm to communicate directly with its partners in the supply chain without using a central system or any form of service provider. For this purpose simple software modules and interfaces have been developed to allow users to access, define and deliver, in a friendly way, the documents needed for each step of the business process. The only central resource is a repository that is used to maintain the document templates that are shared between the partners and used during the interchange process.

The project has adopted the guidelines published by the ebXML initiative. We have chosen ebXML because of it resulted to be the more general methodology that completely faces the issues raised in the B2B field; it was one of the first initiatives providing complete specifications, especially to manage transport, security and reliability aspects; it combines the Edifact experience with the novel solutions introduced along with the XML technologies and appears to be technological implementation independent. Being general, public and free, the ebXML framework allows anyone who wants to develop a sectoral B2B integration solution for a well-defined business scenario (and, considering our needs, the framework proposes a peer-to peer exchange model that is well-suited for the real enterprise collaboration in the Textile/Clothing scenario). The main aim of ebXML is to support two different aspects of the interoperability processes:

- The semantic definition of the documents: ebXML proposes a set of ”core components” used to define the semantic value of a document. Differently from the traditional EDI approach, ebXML emphasises the importance of these components on the entire document structure, and this aspect gives ebXML more flexibility with respect to EDI.

- Several technical specifications on the communication protocols: Moda-ML follows completely ebXML transport specifications.

The two most important points of the Moda-ML initiative have been the definition of a vocabulary of business components, expressed as Business Information Entities and their related XML implementation, and the design of a middleware architecture based on the ebXML messaging service specification; both the document structures and the business processes have been considered, but a special focus has been on the definition of the business documents.

7.3 Dissemination and adoption of the project outcomes

The project achieved the planned results that can be summarized as follows[78]:

- Modellisation of some key collaborative processes.
- Definition of documents templates through XML Schema, User Guides, Dictionary of Terms (near to 400) by means of Document Factory Tools and Methods [14][50].
- Development of demo software implementing the ebXML transport. specifications.

After the conclusion of the Moda-ML project, in 2003, the partners (research institutions, software houses and textile companies) have maintained a technical group assuming the denomination of 'Moda-ML initiative' and have improved the links with other experiences in Europe (i.e. eTexML, e-Chain, TextileBusiness, T2T); the group is supporting the adoption of the results in the industry and is developing farther through the participation to CEN/ISSS standardisation initiatives.

The Moda-ML project has involved a large set of pilot users that have experienced the specifications and the tools released by the project. In the first phase the pilot users began to verify the capabilities of the documents to fit the information flows that were already managed via phone or fax. Sequentially they began to insert the Moda-ML documents in their real workflow and to adopt these documents in their business transactions with customers and suppliers.

For the experimentation purpose each pilot user used a different approach to integrate the XML documents into their company information systems: ad hoc development of the information systems, Oracle modules to manage XML, generalized XML to DBMS mapper, etc. During this phase we have ascertain that the introduction of the Moda-ML framework has suggested further evolution of the internal information systems, with the aim both to offer automatic support to the workflow execution and to add new services, previously not supported, to the customers.

In case the company information system was not able to completely support the flow of the documents, the companies have considered the opportunity to enhance their systems, while the first XML documents could be exchanged via e-mail.

The second important result has been that the adoption of the data exchange framework has resulted very inexpensive (in terms of both licence-free and human resources) for the industries. For example, in a firm already using an ERP system to manage purchase order and order response, the additional cost to import/export each new type of Moda-ML document has been estimated in about half person-day (one person days for the first one). On the other hand, serious investment has been required for internal systems if they were not ready for the collaborative workflows. In this case, sending new documents has been found to be much easier than receiving them.

The results of the project have furthermore given new perspectives and new opportunities to the technology suppliers: they have found resources and references to develop new skills, to offer a better service to their customers and, in general, to face the problem of system interoperability; moreover, they do not consider the Moda-ML solution as a ”competitor” proposal.

The first practical industrial benefits registered consist in the reduction of 80% of the costs of the operation supported by the Moda-ML framework and in saving of 40000 Euro/year for software maintenance. The Moda-ML project has demonstrated during the testing phase to have a good capacity to attract new potential user towards the results of the project: 3 consortia (110 firms) and two relevant players in Como and Prato are going to Moda-ML, and the Industry Trade association has been involved.

Presently there are some running implementations (between small clusters of enterprises) and a more general effort to achieve a critical mass of users; a large consortia of about 70 firms in Biella has experienced successfully the framework with benefits perceived in terms of a drastic reduction of errors, saving of costs for software maintenance and manual data entry and, most of all, improvement of the service to the customers. This activity provides the technical group a feedback that highlights some open issues:

- the effort to integrate external data into the internal workflows is much more higher than setting up the inter company exchange infrastructure; this is the main trouble;
- the interoperability model based on few common document templates to support many business transactions must be substituted with many, very focused, document templates, each of them tailored for a specific type of transaction; they are easier to understand and to be checked with standard tools based on XML Schema and reduce the risk of misunderstanding and misalignment between the partners [30];
- dealing with networks of industries comprising many SMEs it is important to ease the process of ’alignment’ of the systems: each industry cannot afford excessive costs to join a new partner and to tune its systems and organisational procedures.

Finally, the main success obtained by the Moda-ML is that its outcomes has been integrated into the final documents of the TEXSPIN [115] initiative. Promoted by Euratex (European Association of National Industry Trading Association of the T/C) and supported by the European branch of the ISO, CEN/ISSS (European Committee for Normalisation/Information Society Standardisation System)[15] the TEXSPIN initiative aimed to provide a framework for the (B2B) integration of the European Textile/Clothing/Distribution chain. This initiative ended in July 2003, and the final documents have been published in autumn 2003, leading to the specification document ”CEN workshop agreement” of the standardisation initiative TEXSPIN.

7.4 The Moda-ML framework

As our goal is to improve interoperability among independent industrial, commercial and business organisations, this involves defining:

1. a common interchange language for expressing business messages
2. an exchange architecture to efficiently delivery these messages
3. a simple management and a constant maintenance of the established collaboration in order to update the system in accordance with the requirements of the market.

The basic structures of the proposed framework are the vocabulary of XML business components, a set of public, modular document types built out of those components, and the message switching architecture for the delivery of documents.

Any widely used specification defined by relevant standardisation bodies must be considered during the design phase: Internet communication protocols, the protocols used for message exchanges between organisations (e.g. SOAP on ebXML) and languages for creating documents and messages (e.g. XML).

7.5 Implementing a business dictionary for the Textile/Clothing sector

The basic structure of the architecture is a vocabulary (the Moda-ML vocabulary) of well defined terms. The terms represent the basic business components and are defined as XML elements. Some components of these documents are specialized for particular needs, but many components are shared by all the documents: each component of the dictionary represents in fact a well-defined concept that can be specified in the messages. This organization of the dictionary makes it possible to perform the necessary distinction between the syntactical model, the semantic model and the transport model of the messages being exchanged. Public business document types can then be built starting from this set of business elements and upon them in a modular manner, defining rules and constraints to express the interrelations existing among the concepts they represent. Also the structure of these document templates is contained in the vocabulary.

7.5.1 Impact of the requirements

Together with leading industries and trading association we have examined the requirements of the Textile/Clothing supply chain and found a few particularly relevant requirements for any document modeling.

Readability

The T/C sector is characterized by the existence of a lot of small and medium sized companies with poor ICT skills (and with technology suppliers that are SMEs as well). This emphasizes the need for both document readability and specification understandability. In the past, in particular regarding EDI systems, and in some early translations into XML, the emphasis was on the efficiency and normalization, to obtain compact representation of normalized data to be exchanged between hosts, no matter about naming rules and human readability. In the following an example of an early opaque XML document provides no hint as to its actual meaning without a complete documentation effort.

```
<?xml version="1.0"?>
<ORDER RefNo="0001">
<BGM>128576</BGM>
<DTM1>19970812</DTM1>
<RFF IDType="CT" FileID="652744" Line="112"/>
<NAD Of="SU" EAN="6012345678900"/>
<LIN LineNo="1">5012345678900</LIN>
<QTY>900</QTY>
<DTM2>19970812</DTM2>
</ORDER>
```

Nowadays, since the data storage is getting less and less costly, our focus is not on efficiency, nut rather on human readability, in order to facilitate setup and maintenance, and to improve flexibility of systems.

Ready to use: XML-based document models.

A second consequence of the small sizes and the low ICT skills of the actors is the need to build a ready-to-use framework. This means basically to define simple data models without ambiguity or, in other words, to limit the degrees of freedom in implementation. Such data models should be already implemented and represented in a specific technology (XML).

Some existing and past approaches (the BSR initiative for example, and also UN/TDED, the ONU dictionary for the international commerce) adopt a technology-neutral representation of the terms of the e-Business. The problem with these approaches is that too little semantics can be encoded in XML-based specifications and thus such specifications allow for different interpretations and consequently for different, often incompatible, implementations. All this hampers system interoperability. In our approach, we still adopt XML as the basic format for specification.

To avoid ambiguity the terms should be constrained by 'facets' to limit their features and values. An example is the definition of the precision or the allowable range of values for the numeric data via XML Schema; another example is the large use of enumerations to, a priori, define the set of allowable values; yet, the more recent UBL specifications, for example, are expressed using XML, but use only a part of the features available via XML Schema.

Our aim is to avoid the situation in which XML documents are accepted into an enterprise information system as ’valid’, but contain data values that the internal systems will not be able to manage. On the contrary, we want to assure that the notion of compliancy with the specifications is as much as possible close to the notion of interoperability between enterprise information systems (about the relationship between compliancy and interoperability in ICT standards see [40]).

7.5.2 Modelling choices

Many process centered document models

The business processes of the T/C sector are quite different from those of other industrial sectors,because of timeliness, like some others ’fashion’ sectors, and complexity of the production processes; so we believe that sector specific and highly specialised data models (XML Schemas) could result much clearer and more understandable by users and developers. The familiarity with the concepts and the terminology of their own processes is meant to reduce misunderstandings and errors and to create a lower cultural threshold for the adoption of the solution.

In this perspective we target to reduce the complexity of the supply chain with the design of many specialized models of documents rather than with different implementation of the same documents.

Moreover, each of these documents need to be, from the viewpoint of implementation, self-contained and independent. This approach is opposite to that of the most known EDI and XML/EDI implementations (EDIFACT, EANCOM, EAN.UCC, UBL).

Two examples of the effects of this choice are the following:

- Differentiation by treated goods: XML documents within the same process but related to different kinds of goods were considered different (for example Fabric and Yarn purchase) because of the necessity to describe different properties of the goods and (for example) different packing instructions.

- No shared documents between different transactions: some transactions exchange similar information, but if the bulks of functionalities (and mandatory information) are different they cannot be supported by the same document model; for example, ’purchase’ from a catalogue of products (purchase order, based on product identifiers) is different from the request to produce something whose specifications must be included in the order (manufacturing order). On the contrary, reusing common blocks of information inside different document models is strongly pursued.

It is worth noting that the risk of an exploding number of templates is avoided given the limited kinds of goods (”fabric”, ”yarn”) to be exchanged; in fact the implementation to support the processes (related to purchase of fabrics and yarns, manufacturing orders and subcontracting for darning, dyeing and finishing) has led to the development of about 21 document templates.

Standard orientation

The proposed framework is close to standardisation activities and results such as:

1. the W3C Recommendations for using XML for EDI;
2. the ISO Naming Conventions (standard ISO IEC 11179 [58] that were adopted also within ebXML);
3. the reference to the recommended codifications (Terms of Payment, INCOTERMS); they were selected from the ONU dictionary for the international commerce (UN/TDED);
4. the ebXML architecture
5. the analysis of the business processes and of the related information in EDITEX (EDIFACT implementation for the textile clothing sector [42]).

The adoption of these references ensured a good affinity of the dictionary with the ebXML approach, although straightforward compliance cannot be demonstrated.

Maintenance and fast development

Finally, the large number of dictionary items to be managed, in multiple languages, recurring in many document templates (but combined in different processes), has led to the need to build dictionary tools that:

- support both the collaborative process description as well as the document structures and semantics;
- speed up the maintenance and design process, assuring fast and error-free release of the XML Schemas and related User Documentation;
- facilitate the reuse of semantic blocks (Aggregated Information Entities)

7.5.3 Dictionary structure

Content model

The data model of the dictionary is based on two sets of entities: the entities related to the representation of the processes and those related to the document templates and business information entities.

The dictionary is composed of the Business Information Entities (BIEs) (as defined in [58]). Within our model they are organised as follows:

a) we define two types of semantic units: the SIMPLE Business Information Entity (representing an ”atomic” piece of data) and the COMPLEX Business Information Entity (representing a ”molecular” data composed of diverse component units up to the full document template);

b) in order to keep things as simple and flexible as possible, for each semantic unit we create both an XML-type and an instance. In general a COMPLEX BIE may have children that are ’optional’ (cardinality 0..N) or mandatory (cardinality 1..N);

c) in order to exploit the richness of the XML syntax, we have differentiated the Simple BIEs, which carry an effective content, into ELEMENTS (for data) and ATTRIBUTES (for meta-data);

d) the formal definition and control over the content of Simple BIE is obtained by applying ”facets” (e.g.: maximum length, minimum inclusive, enumeration or code list) to the ”built-in simple types” defined in the W3C Recommendation (e.g.: string, integer, decimal, date).

It is worth to note that alternative constructions (different sets of children) are allowed when an enumeration type can be replaced by a free text describing the same concept. The use of free text, although not completely forbidden, is strongly discouraged, while the adoption of a set of predefined codes (enumerations) is preferred in order to reduce risks of ambiguities and errors. The codes are either international (ISO, UN/CEFACT, etc.) or self-defined whenever international coding is not available.

Message structure and naming conventions

A very hierarchic and compact data organisation was chosen for the dictionary . At the highest level a message is made of COMPLEX BIEs that are either:

- common to all the documents (e.g. TERMS) and thus reusable in a large set of document types;
- depending on the goods category (for example the txtCOMINFO section, dedicated to commercial info as packaging instructions) and thus reusable in a smaller set of document types;
- depending on the pair ”goods category, type of document” (for example toBody, related to order of textile) and thus specific of the document type.

In turn each of these complex BIEs can contain other complex BIEs, but only of equal or greater generality. So, as a thumb rule, generality grows top-down and reduces bottom-up.

The naming convention reflects this degree of semantic generality to clearly identify the generality of each BIE with respect to the document type and requires the adoption of meaningful tags in order to improve the readability.

Namespaces are used to clearly identify the different releases of the set of documents: each version corresponds to one namespace. For example xmlns:ml="http://www.moda-ml.net/moda-ml/repository/schema/v2004-1" clearly identifies that the XML Schema belongs to the 2004-1 version of Moda-ML; until today five versions have been released, three of them publicly available.

The dictionary entries

The last foundation of our design is the independence of our Basic Information Entities (BIE) from the final Schema representation. In fact the unique repository of all the BIE is the on-line dictionary (that is public and freely accessible); from those elements, Schemas are computed dynamically to implement the different templates (that are publicly available as well).

This allows a separation between the dictionary (semantic aspects) on one hand, and the final XML Schemas (syntactic aspects) on the other .

In general the same entry can be used in different Schemas with different hierarchical positions, usage conditions (e.g., required, conditional, optional) or number of repetitions, providing that the "generality rule" above is maintained.

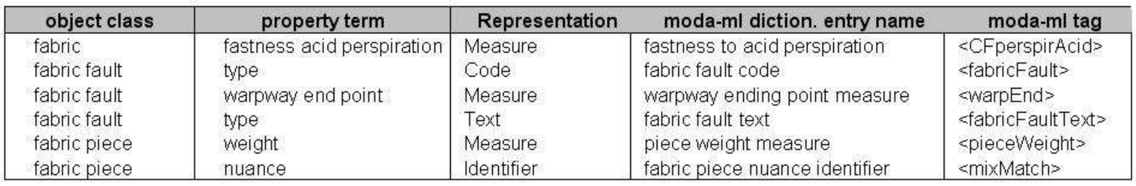
Each Dictionary Entry, be it a Simple BIE - Element, a Simple BIE - Attribute or a Complex BIE, is identified by

- a unique code (e.g. 35-755-04)
- a unique "tag" (e.g.: specDate)
- a unique name (being an ordered triple (object class, property, representation) as they are defined in [58])

Code and Name are interrelated, i.e, in the example above, 35 identifies the Object Class = product, 755 identifies the Property = specification and 04 identifies the Representation = date.

The adoption of the ebXML recommendation about the [58] specifications eases the maintenance and development of the dictionary: terms related with the same physical object or the same kind of property are easily selected from the dictionary (fig. 7.2).

It should be noticed that the Representations allowed in our Dictionary (as created from the original XML built-in Types) coincide with the "Approved Core Component Types" of ebXML.



object class	property term	Representation	moda-ml diction. entry name	moda-ml tag
fabric	fastness acid perspiration	Measure	fastness to acid perspiration	<CFperspirAcid>
fabric fault	type	Code	fabric fault code	<fabricFault>
fabric fault	warpway end point	Measure	warpway ending point measure	<warpEnd>
fabric fault	type	Text	fabric fault text	<fabricFaultText>
fabric piece	weight	Measure	piece weight measure	<pieceWeight>
fabric piece	nuance	Identifier	fabric piece nuance identifier	<mixMatch>

Figure 7.2: Term association with Object class

Managing semantic diversity

The need to define very general documents types, typical of EDIFACT, leads to manage the ”semantic diversity” through the combination of a general object (e.g.: Party) with specific qualifiers (e.g.: Party Qualifier assuming values such as ”Buyer”, ”Seller”, ”Consignee”, etc.). The result is the compression in the size of the data dictionary, but on the other hand this has often proved to be an obstacle to the clarity of the language. Furthermore, today, this approach bears some problem to fully exploit the XML Schema potentiality in data validation.

For instance, consider the fragment:

```
<measure application="physical dim" dim_type="length">10</measure>
<measure application="physical dim"  dim_type="weight">1</measure>
```

This fragment is obviously particularly difficult and convoluted to read; but the main problem is that it is very difficult, when using XML Schema, to check separately the range of values of ’length’ and of ’weight’, since the XML-Schema syntax does not allow the definition of constraints of an element as a function of the value of one of its attributes.

On the contrary the fragment

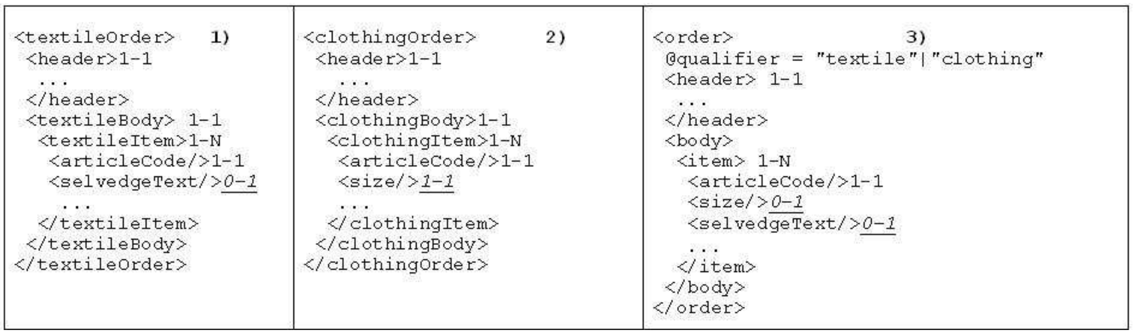
```
<length>10</length>
<weight>1</weight>
```

is clearly comprehensible and allows to define different ranges of values using XML Schema. In this way we can find a lot of standard libraries capable to check more closely our documents and, therefore, we avoid the need of implementing ad hoc software from rules found in the implementation guides.

Specialised document rather than general

Analogous issues arise when different optional elements are in similar documents.

In the example (fig. 7.3), we might create two similar document types (textileOrder and clothingOrder) or one generic Order including all the optional elements. The textileOrder has an optional ”selvedgeText” while the clothingOrder has a mandatory ”size” element.



```
<textileOrder>    1)
 <header>1-1
  ...
 </header>
 <textileBody> 1-1
  <textileItem>1-N
   <articleCode/>1-1
   <selvedgeText/>0-1
   ...
  </textileItem>
 </textileBody>
</textileOrder>
```

```
<clothingOrder>       2)
 <header>1-1
   ...
 </header>
 <clothingBody>1-1
  <clothingItem>1-N
   <articleCode/>1-1
   <size/>1-1
   ...
  </clothingItem>
 </clothingBody>
</clothingOrder>
```

```
<order>                  3)
 @qualifier = "textile"|"clothing"
 <header> 1-1
  ...
 </header>
 <body>
  <item> 1-N
   <articleCode/>1-1
   <size/>0-1
   <selvedgeText/>0-1
   ...
  </item>
 </body>
</order>
```

Figure 7.3: 1) Specialised textile order. ”selvedgeText” is optional. 2) Specialised clothing order. ”size” is clearly mandatory. 3) Generic order. It is not clear whether the indication of ”size” is always optional or it depends upon the value taken by the order qualifier; ”selvedge text” is not excluded for ”clothing” order as well as ”size” for ”textile” order. Only the Guidelines can help.

The example demonstrates that forcing few document structures to express too many content types leads to an increase of the complexity of the paper-based documentation and reduces the efficacy of automatic validation of the messages. The adoption of validation languages to express ’co-constraints’ (i.e. Schematron) could solve the problem but they are not really as spread and accepted as XML Schema).

Schema generators

The choices regarding the use of namespaces, the scope of the single elements, the role of type declarations versus instances can exploit the definition of different programming ”styles” of XML Schema (such as ”Venetian blind”, ”Russian doll”, ”Salami slices” , ”Garden of Eden” [22]).

In our approach the Schemas are automatically developed from the dictionary by the generator according with the ”Venetian blind” model, that emphasizes compactness and readability. The choice to maintain all the elements of the Schema in one unique file, without spreading different elements in different files, characterises our Schemas with respect to other frameworks like UBL or EAN.

A potential negative effect is the repetition of many elements in different Schemas, with the risk of a difficult and expensive maintenance; but this risk is completely overcome in Moda-ML by the use of the automatic Schema generator based on the dictionary.

The positive effect, on the other side, is that everything related with a document model is completely contained in a single Schema, with higher comprehensibility and a reduced risk of error in the configuration of the applications. Only some very large tables related with enumeration types (for example codes representing the currencies) are left externally to the document schemas.

7.6 The document factory and the Message Service Handler

The effective implementation of the vocabulary is done using a database application that provides a sophisticated description of the defined basic components. This database collects any information on the semantic blocks needed to build the document types. Such information include the name of the XML elements, their description and the associated properties such as data format, length, range of permitted values and so on. The vocabulary further specifies a root element for each document and all the relations existing among the elements (such as sequence order, cardinality and so on).

The database is designed also to maintain the information about the business processes of the framework. A set of application will then re-create the complete set of rules (an XML Schema) for each document type by starting from the root element and following the defined relations. Moda-ML also provides a set of XSLT style sheets to create HTML pages off the XML instances so that the document content can be visualized in a readable manner even if using a simple Web browser. The Vocabulary represents the core of the management of every aspect related to the Moda-ML document types, schemas and instances. We call this approach the XML document factory. This architecture is depicted in fig.7.4

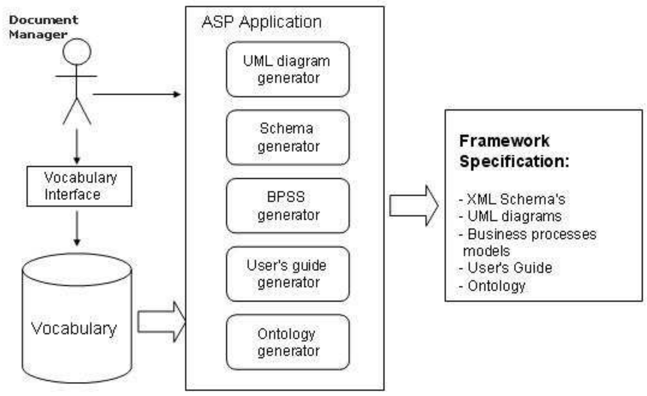


Figure 7.4: The Moda-ML document factory architecture

Together with the vocabulary, the framework provides the necessary tools to exchange Moda-ML documents. These tools are collectively called the message switching system. The message switching system implements a transport protocol based on ebXML messaging service specifications; since the Textile/Clothing sector is composed of various kinds of enterprises, each characterised by a different level of technological sophistication in its information systems, it becomes fundamental to create simple software modules that can be made publicly available, providing an easy and low-cost integration with complex legacy information systems within skilled companies. The main component of the Moda-ML message switching system is the Message Service Handler (MSH), that acts as an email client, sending and receiving Moda-ML documents as attachments to email messages: it takes care to validate Moda-ML documents and it uses SMTP as its transport protocol. In order to enhance the functionalities of the MSH, we have considered security aspects for authentication and non-repudiation of Moda-ML messages.

Moda-ML staff and the pilot users can automatically generate the XML schema and the user guide on every message, even in course of definition; generating XML schemas or user guides produces documents that can be immediately downloaded. A generic user seeking information about the message usage can only download all the developed versions of the schemas and the users guides. These operations are all ASP applications executable from the web.

7.7 Extracting a semantic view from the vocabulary

In this section I describe the approach to provide a semantic description of a the Moda-ML framework outlined before. Especially, this semantic description regards both the general concepts inherent the production processes and, more precisely, the data model implemented by the framework. The aim is two fold: from one hand, we want to ease the management, usability, maintenance and efficiency of the framework and to improve its powerfulness; on the other hand, semantic description could be the bridge to connect and to integrated more efficiently the business model views that are local to the enterprises with a global external view common to the whole business sector. In this perspective semantic descriptions and the bounded technologies are intended as infrastructures that act as a support for standard management, and not as a substitute for them.

7.7.1 From the dictionary to the ontology

The last developments [29][106][75] about interoperability have highlighted the relevance of the semantic aspect for the interoperability problem [54]; also abstract models for interoperability [11] should plan modules for semantic description management. The semantic description for the framework is particularly relevant in the context of the document-based interoperability frameworks, that define as basis for the B2B integration a set of complete business documents.

In order to efficiently elaborate the instances of the business documents, a deep and exhaustive knowledge of the meaning of the documents is needed. In particular, an automatic elaboration of the documents requires a formal description of the semantic associated with the data and with the terms contained in the documents. The study of semantic aspects for interoperability joins with the Semantic Web vision [113], where machines can understand the semantic of the documents diffused in the web without human assistance; the idea is to express information contained into the (web) documents in a meaningful way accessible to the applications. The Semantic Web thus would result as an infrastructure that provides semantic information about documents, upon which different applications can be implemented. In the e-business context, formal descriptions could be provided through the definition of an ontology that represents that implicit concepts and the relationships that underlie the business vocabulary, and consequently the documents; [96] provides an example of a conversion of a controlled vocabulary in an ontology.

As aforementioned, the main information ”domain” described by the vocabulary are:

1. the collaboration scenarios (business processes).
2. the general concepts treated by the framework and pointed out in the business documents.
3. the structures of the documents.

We have 1) integrated the framework with the semantic description of each of these domains 2) exploiting a software component (developed within our software architecture) that can extract in an automatic manner the needed information from the vocabulary. These domains basically represent three semantic areas to include within an ontology. We expect the ontology to support data integration between the enterprises, standards and B2B protocol, to facilitate framework development and maintenance, and to acts as the interface towards the semantic web.

In the field of ontology development it does not yet exist a well established design approach [62]: it in general results to be an iterative process, where the ontology being developed is continuously evaluated in order to drive and to improve the development process itself. In our approach, we started identifying the basic concept we want to extract from the vocabulary. Then we created a draft ontology to identify the best structure to model the final ontology. Finally, we have developed the tool that build automatically the whole ontology following the pattern fixed in the drafts.

The developed ontology reflects naturally the structure of the vocabulary; for each of the semantic area listed above, we identified specific basic concepts used as cornerstones to build the final ontology. We have then mapped these basic concepts to a set of OWL superclasses; they represent the roots of a set of corresponding sub-ontologies that describe these superclasses together with their subclasses and their relationships with other classes or sub-ontology; each of these basic concepts represents also a starting point for the browsing of the ontology. For example, we have fixed the basic concept of *Process* that has been mapped in the superclass *BusinessProcess* and in the corresponding sub-ontology *BusinessProcess* (that is described in the BusinessProcess.owl file). The final ontology is constituted then by all of these subclasses.

With this approach, we have generated a modular ontology, in which each basic concept (and the associated sub-ontology) can be managed independently from the others and is identified by its own namespace; we have moreover streamlined the designing process and the structure of the ontology, and ease its maintenance and its future development. The following sections deepen the content and the structure of the semantic areas listed above.

The collaboration scenario ontology

This area models the business scenario of the target business sector. It includes the following basic concepts:

- Process: A generic process of the supply chain
- Activity: A generic activity within a process
- Document: A generic business documents exchanged between two actors; it corresponds to a single transaction within an activity.
- Actor: A business partner involved in a business activity
- Good: The good treated in the business process

Each of these concepts is then specialised in subclasses. Each process, activity or document is characterised by a type, and refers to production good; the ontology represents the taxonomy of these concepts, together with the relationships between the processes, the activities and the documents. Finally, each document is linked with a component that represents the root of the XML documents. In this way we connect the world of the business scenarios with the world of the business document structures.

The document ontology

This semantic area defines the structure of the terms defined by the vocabulary and used within the business documents (in the following we call them components), and consequently reflects also the structure of the business documents themselves. Anyway, the designed ontology does not model the syntax of the documents: this task is achieved through a proper XML Schema for each document; we have rather decided to highlight the relationships between XML types, elements and attributes, and the semantic content of the documents, that is strictly related with the business sector the documents are bounded to (to this aim this ontology exploits the component ontology described later). This means also that those layout elements of the documents, (like "Header" or "Body") used for presentation or formatting purposes are not included in the ontology. The basic concepts of this area are:

- Document: A generic business document exchanged between two actors.
- Component: An XML element or attribute used by the documents.
- Type: A generic XML type.

We basically model the idea that a component, regardless it is an element or an attribute, is an instance of an XML type. We have then created the class Component to represent a generic component (element or attribute) that can be used to build a document. Each Component maintains a relationship with the class XMLType (see fig. 7.5). We thus maintain separately both the XML types and their instances. The XMLElement class has been provided with a set of property (like "hasAlwaysChild") used to maintain the relationships (and the hierarchic structure) between the various elements. Finally, each document is linked with a specific component that represents the XML-root of the document.

The component ontology

This semantic area contains the generic concepts and properties that represent the "world" described by the vocabulary and treated by the framework, relating them with the XML components; this is surely the most interesting area for the semantic description of the business sector. A ISO/IEC 11179 compliant vocabulary basically maintains a set of Data Element, each of them composed of its Object Class, Property Terms and Representation Terms. The three dimensions used to characterise the components of our vocabulary, that represent the basic concepts for its description, are then:

- Object: An abstract concept related to the business.
- Property: A property of an object, described by a component.
- Component: the component (an XML element or an attribute) used by the documents; the Representation Terms is finally given by the types of the components.

For example, our vocabulary contains the component (that finally is mapped to an XML element) FabricCompos. This component is used to describe the property "*fibrous composition*" of a "*fabric*". In the vocabulary, the component is then directly related with the general concept Fabric and with the property "*fibrous composition*".In this way, the framework maintain a list of triples Object-Property-Component. Each object of the ontology has specific properties with which is linked using the relation hasProperty; obviously, each object can have many properties. Fig. 7.5 depicts the conceptual model implemented to express the relationship between an Object with its Property and its Representation (the Component).

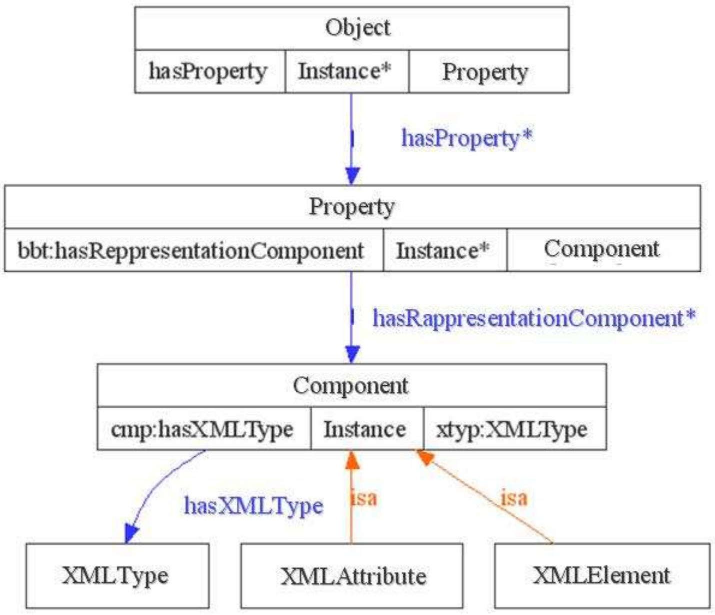


Figure 7.5: The modelling of Objects, Properties and Components in the Moda-ML ontology

We can now consider the final structure of the resulting global ontology is depicted in fig. 7.6 It shows:

- the three semantic areas described in the ontology.
- the subdivision of the areas in sub-ontologies, depending on the basic concepts described by each of them (each square represents a sub-ontology). Each sub-ontology is identified by its name together with its namespace.
- the links between the sub-ontologies (the defined relationships are represented by the arrows). The sub-ontologies are connected defining for the superclasses some ObjectProperties that have their domain local to their sub-ontology, and their range defined in an external sub-ontology.

The image highlights as there are some overlaps between the semantic areas: for example, the BusinessDocument class appears both in the Process Area and in the Document Area. These overlaps represent the contact points of the different semantic domains of the ontology, allowing to interconnect all the ontology in a common vision of the framework.

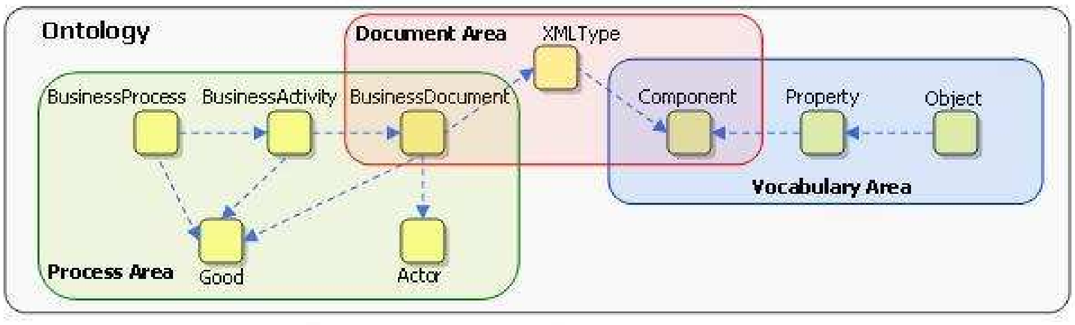


Figure 7.6: The global structure of the Moda-ML ontology

The integration with external ontologies

A relevant aspect that we have considered in this activity is the interaction of the resulting ontology with other ontologies: it, in fact, is not thought to work in isolation respect other knowledge domains, nor we claim to model everything within our ontology. The main aim that underpins the ontology development is the creation and specification of common models that could guarantee the needed interoperability among heterogeneous systems. In particular we have identified two initiatives for the definition of middle and upper level ontologies that can be exploited to integrate different semantic descriptions:

- The SUMO ontology [84], that defines very general concepts, like time and number.
- The MILO ontology [85], that defines more specialised concepts. It defines for example concepts like product, order or delivery, tailored for the business, but also generic terms that can be used in specific sectors, like fabric, that obviously regards the Textile/clothing sector.

In our ontology the set of the basic concepts mapped in the superclasses (like process, good or documents), and the concepts enclosed in the vocabulary area, that include the generic concepts treated by the framework, represent our interface towards middle level ontologies like MILO. OWL provides to this aim some properties and constructs that can be used to associate our concepts with those defined in an external ontology: for example equivalentClass or equivalentProperty, seeAlso or sameAs.

7.7.2 The ontology generator

Once we have defined the structure of the target ontology we aim to produce, we developed a software tool able to generate in an automatic manner the OWL files starting from the information held in the database (i.e. starting from the vocabulary definition)[46]. This allows us to produce outright a semantic description (the ontology) of our framework each time we update, integrate, or extend it. The ontology generator consists of a set of software modules. It will be used periodically, basically each time a new version of the framework is released, and thus works as a batch application that does not require a specific user interface. We just fixed some parameters to drive the generation of the ontology (for example, we can set file names, ontology names, namespeces, name of some main classes and so on). The ontology generator has been developed upon the Java architecture J2EE: each software module consists of Java classes. This choice is justified by the availability of useful and strong OWL API for this architecture. In particular, we have exploited the Protegè OWL-API developed by Holger Knublauch at the Stanford Medical Informatics(SMI), and adopted by the OWL Plugin of Protegè. The development of the application (fig. 7.7) has involved:

1. the development of the DictionaryConsole Javabean that extract data from the database.
2. the development of the OntologyManager Javabean that builds the ontology and produces the OWL files using the Protegè OWL-API.
3. the development of a web interface using servlet and JSP pages that allows the staff to remotely access to the tool.

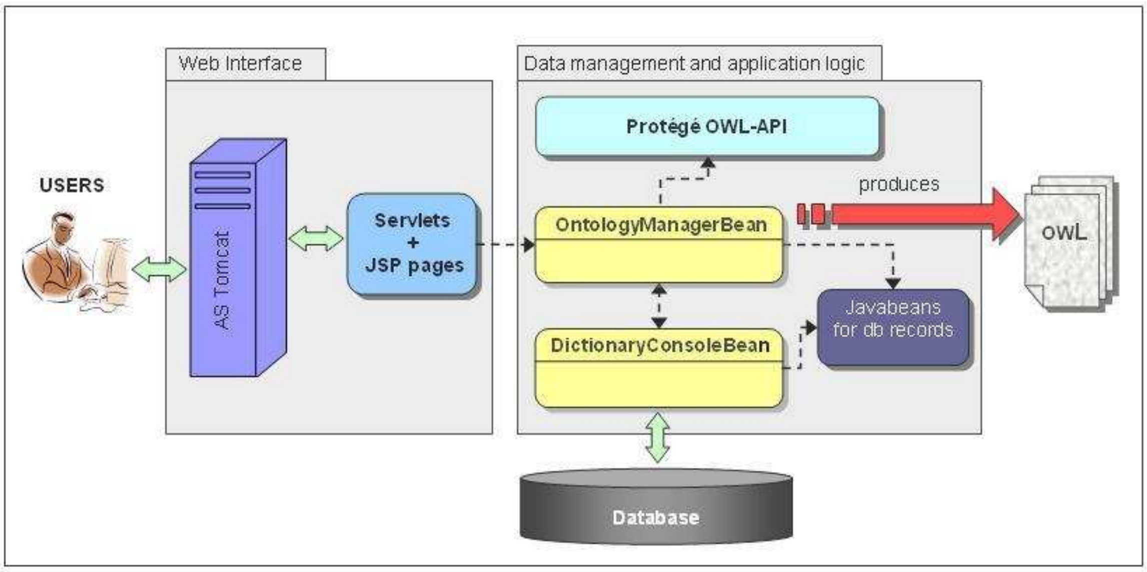


Figure 7.7: The architecture of the ontology generator

The last version of the framework defines a vocabulary composed of 494 terms, a set of 32 business documents, and specifies 8 business processes for the enterprises. The generated ontology includes 373 classes, 44 relationships and 1098 instances.

We consider also these results as a premise for further developments in three areas:

- mapping and integration mechanisms to interface each other heterogeneous data models defined by different subjects, like standards and ERP systems;
- study on the complementary approach of the definition of business vocabularies starting from an ontology;
- linking of the ontology with other (external and/or upper level) ontologies.

7.8 Business process and profile management

The proposed framework defines both business messages and the way to exchange them, but aims to provide also a mechanism to improve the workflow management, the integration and the interoperability among different partner of the Textile/Clothing supply chain, and the agreement about interaction mechanism. In order to achieve this aim, a standard formal definition is needed to describe business processes. This definition ease inter-enterprises collaborations and commercial transactions, especially spreading information about business processes.

Moda-ML business processes are specified using BPSS documents. Every BPSS is generated directly from the Moda-ML dictionary. Using a web interface and an HTML form it is possible to select a process and to build, from the implicit definition contained in the vocabulary, the corresponding BPSS.

During the development of the tool to create BPSS document we have found two different DTD to validate our BPSS: one found on www.ebxml.org, and the second on www.oasis-open.org. Also the examples we have found on the web sites to understand the correct use of the BPSS were not coherent: these examples propose different ways to structure the XML elements and adopt differently a XML schema or a DTD to validate the BPSS. Finally we decided to adopt the DTD for the version 1.01 of BPSS provided by the official web site of ebXML.

Moda-ML process definition is not completely compliant with ebXML specification: Moda-ML processes are composed of one or more activities each of which can consist in the exchange of one or more Moda-ML messages. Differently from ebXML (and to better fit with really situations), Moda-ML message exchange is not subdivided into a ”request-response” atomic activities .

ebXML uses the BPSS as the starting point to generate a CPP document. As defined in [23], a CPP (Collaboration Protocol Profile) ”defines the capabilities of a Party to engage in electronic Business with other Parties. These capabilities include both technology capabilities, such as supported communication and messaging protocols, and Business capabilities in terms of what Business Collaborations it supports”. A CPP document describes an enterprise and the role it can carry out inside a business process, but does not allow to define more specifically the information that can be managed by the enterprise itself. The ebXML definition of CPP results to be not flexible enough to tackle the requirements of the T/C supply chain.

Together with the generation of BPSS documents, the Moda-ML project takes care to produce CPP documents to describe the collaboration profile of the Moda-ML users. Moda-ML refers to 2.0 version of the CCP, available on the official site of ebXML (www.ebxml.org).

On the other hand the Moda-ML approach to define the enterprise profile aims to be more flexible than the ebXML one. In the Textile/Clothing sector the business collaborations can change depending on the final product that has to be realized. Moreover the enterprises in general are not well disposed towards sudden and drastic changes of their ”well-proved” management systems. They surely prefer a gradual approach in adopting new mechanisms to manage business processes, to exchange documents and contact their partners. It is not feasible to impose completely new transport mechanism or to propose absolutely unusual and unfamiliar documents.

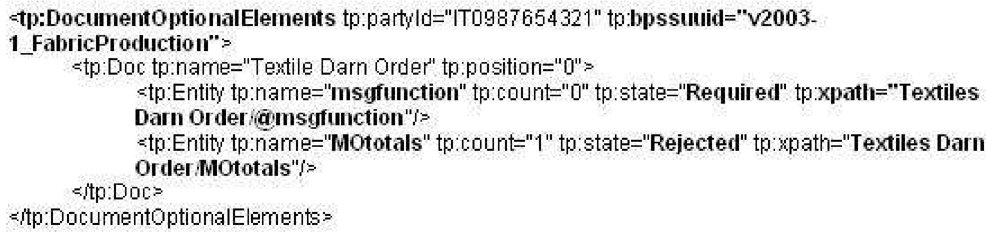
These practical issues conditioned the way to compose CPP documents: in defining their profile, Moda-ML users require:

- To choose a specific sub set of documents to exchange in a particular business process. The T/C supply chain is composed of a large variety of partners that want to interact according to common processes, but can implement only a sub set of the exchange activities inside the processes. In general, there are many enterprises that, depending on their internal organisation, do not know how to manage a certain kind of information, and how to insert the related electronic business documents inside their internal workflows.

- To specify which part of the business documents they can manage. Moda-ML messages contain different kinds of information. Depending on the enterprise, this information can result as absolutely indispensable, optional or instead not required. Moda-ML document schemas provide a flexible structure of business documents to reflect enterprise requirements, providing a mechanism to produce business messages customized for each situation.

In order to provide a mechanism to solve the first issue, we have considered three possible approaches: modify the CPP DTD for an enhanced version of CPP documents, use of the ”SimplePart” element to specify the message type to be sent for a particular activity as a ”completely arbitrary type”, or use the ”CanSend”, and ”CanReceive” elements to specify the transport method to deliver business documents. Finally we have chosen to use the CanSend and CanReceive elements to point out different ways from MSH to exchange the messages. In this way a partner of the supply chain, that does not want to use the Moda-ML framework (i.e. MSH) to exchange a particular business message, can point out, using these two elements, an alternative mechanism to deliver the message (i.e. phone or fax or whatever). Using ebXML terminology, in our CPP a user can specify inside a Binary Collaboration the supported sub set of Binary Transaction Activity, avoiding the constrain to manage the whole Binary Collaboration (that represents the business process) defined.

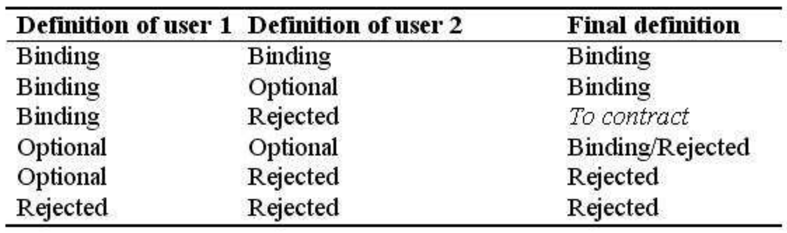
To solve the second issue, we decided to modify the DTD of the CPP introducing a new element to allow the users to specify how she/he considers determined parts of the message. In this way a user editing his/her own CPP can define which information she/he judges to be binding, optional or rejected inside the business message that have to be exchanged. A brief example of the use of this new element is depicted in fig. 7.8.



```
<tp:DocumentOptionalElements tp:partyId="IT0987654321" tp:bpssuuid="v2003-
1_FabricProduction">
        <tp:Doc tp:name="Textile Darn Order" tp:position="0">
                <tp:Entity tp:name="msgfunction" tp:count="0" tp:state="Required" tp:xpath="Textiles
                Darn Order/@msgfunction"/>
                <tp:Entity tp:name="MOtotals" tp:count="1" tp:state="Rejected" tp:xpath="Textiles Darn
                Order/MOtotals"/>
        </tp:Doc>
</tp:DocumentOptionalElements>
```

Figure 7.8: A brief example of the new element

This information will become fundamental for a successive definition of the Collaboration Protocol Agreement (CPA) document: the table 7.9 can be used as a reference to match those message parts that are customisable to define a final message.



Definition of user 1	Definition of user 2	Final definition
Binding	Binding	Binding
Binding	Optional	Binding
Binding	Rejected	*To contract*
Optional	Optional	Binding/Rejected
Optional	Rejected	Rejected
Rejected	Rejected	Rejected

Figure 7.9: Possible matches between different requirements on same part of the message

Following the guidelines of the project, that developed all of its services like web applications, the tool to generate the BPSS has been implemented as a set of VBScript dynamic web pages. Once that every element concerning the new business processes and the relative business documents has been specified and its description has been included in the vocabulary, it is possible to automatically generate the related BPSS starting from such description 7.10. Using an appropriate web interface the Moda-ML staff can select a particular process and then, by a simple click, generate the XML files. The BPSS documents are then stored in a directory publicly available via web for Moda-ML partners (Fig. 7.10).

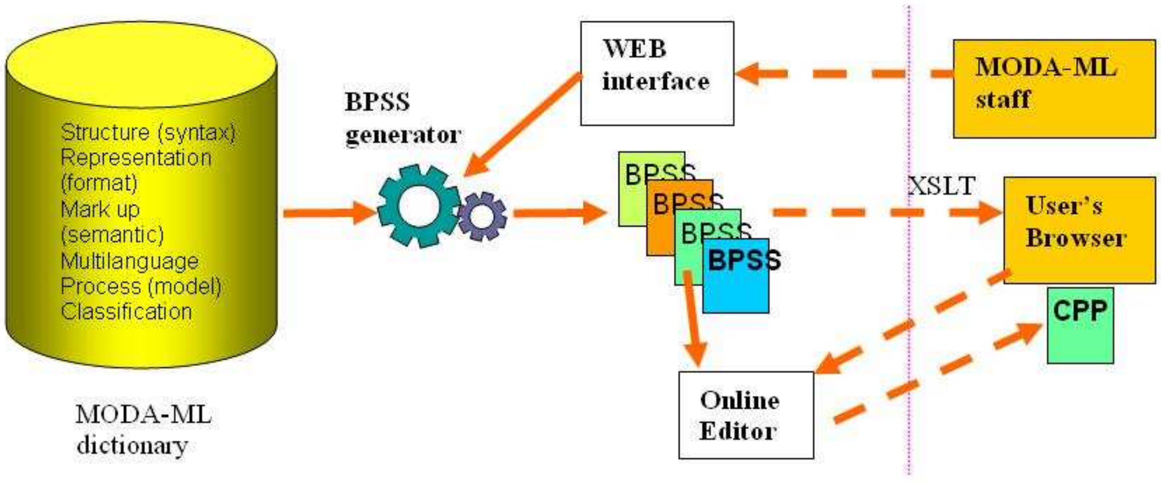


Figure 7.10: Moda-ML architecture for BPSS and CPP management

The two main aims of this tool are:

- to have a standard-formal definition of the processes supported by the Moda-ML framework.
- to maintain the vocabulary as the only central component of the framework to update during its growth.

Besides the BPSS generator tool, we have implemented an on line editor for CPP documents. In fact, starting from the BPSS it will be possible for each actor to edit his own Collaboration Protocol Profile (CPP), that will be used as reference for the role of the actor in the relevant process. The possibility for every enterprise to tailor the message set defined in the BPSS and to customize interaction parameters on its own capabilities is related with the basic requirement of the Moda-ML project that aims to face the heterogeneity of the Textile/Clothing sector.

From the user's point of view, BPSS and CPP implementation can enhance the understanding of the services provided by each enterprises, and the analysis of possible interaction mechanisms.

Naturally the work in implementing BPSS and CPP documents targets to a future definition of CPA documents. As explained in ebXML specification, CPA documents represent the agreements achieved by two different partners in the supply chain to do electronic business. This agreement stems out from the possible matching between the enterprise profiles expressed in CPP. The future step will be the analysis of (semi-automatic) tools to build CPA documents.

7.9 Lifecycle of the framework

To understand the experience of Moda-ML and the subsequent standardisation initiative TexSpin and TexWeave it is worth to examine the phases of its design and implementation (see fig. 7.11). These phases represent the evolution during its lifecycle. This analysis aims to point out which could be the right guideline developing a B2B solution that could be widespread adopted in specific business domain. Basically, the idea is that technology research, software development but also standard definition should be carried as more as possible in concert.

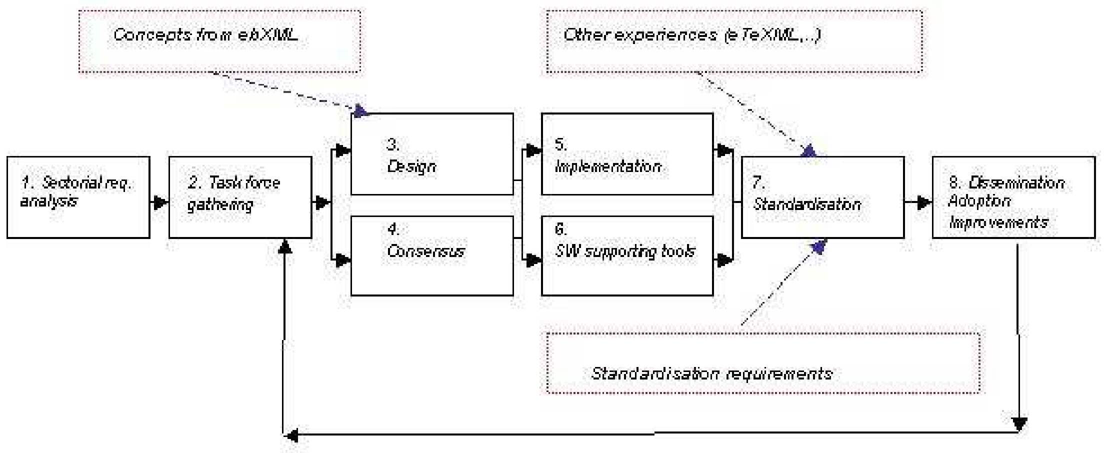

Concepts from ebXML
Other experiences (eTeXML,..)
1. Sectorial req. analysis
2. Task force gathering
3. Design
4. Consensus
5. Implementation
6. SW supporting tools
7. Standardisation
8. Dissemination Adoption Improvements
Standardisation requirements

Figure 7.11: Life Cycle of Moda-ML combined with the TexSpin standardisation initiative

The phases are:

1. The macro analysis phase: this phase regards the study of the main issues and requirements of the target sector (Textile/Clothing sector), to highlight key features. In this context existing solutions were criticized, cause of their poor usability. The main reference technologies have been identified and selected as the basis for the future development of the framework.

2. The organization phases, to set up a working team to carry out a project for the interoperability framework. In order to consider different working and usage perspectives and to exploit different abilities, the team was composed of:

- Industrial leading firms, with the knowledge of practical and pragmatic vision of the sectorial issues.
- Research institutes that contributed with the knowledge about proper technologies and solutions to adopt tackling interoperability problems.

- Technology suppliers, that should support the firms in the adoption /installation/customization of the B2B framework .

The more relevant aspect at this stage consisted in the bottom-up approach for standardisation: rather than starting from an official standardisation body, the initiative got on with well-established industrial districts, where each subject was familiar with the business scenario and was willing to work with others.

3. The Document structure design phase, where the documents and data formats have been defined. After a more specific analysis, conducted in cooperation with all of the teams, two basic results were reached:

- the definition (starting from the EDIFACT - EDITEXT experience) of the existing collaboration processes between the enterprises, and the set up of the document structure and the data model (see section 7.5.3, 7.8).
- the definition of the transport mechanism (see section 7.6).

4. The consensus creation activity: since the beginning of the analysis and the design tasks, a campaign to involve potential users and their trade associations started in parallel. This activity has comprised both the creation of a public Web site as well as an organisation of focus groups and technical events addressed to the partners. In this stage Moda-ML has defined awareness meetings (about 20, with hundreds of participants in the industry and in the area of consultancy and solution providers), targeted visits and any other types of possible promotion, including publication of scientific papers and participation to highly visible international Conferences dealing with the e-Business or with the Textile-Clothing industry.

After the phases 3 and 4, the activities to develop proper software for both the framework designers and the framework users start.

5. The software support design phase: this phase regards the definition of the document factory architecture to manage the interoperability framework. Instead of defining each business document as a single entity, we detected a large set of business components (near 300 terms) upon which the final business documents were built. In this manner we subdivided the construction of the documents. Besides the definition of the component set, we specified a set of document types built upon those components.

6. The implementing phase. Moda-ML provided a set of basic tools to ease the spreading of the framework and its usage for the target enterprises. These tools are:

- A demonstration software that implements a simple Message Service Handler (ebMS 1) to provide the users with a proper free software to exchange business documents.

- Managing tools: implementing tools to generate multilingual user's guides and UML diagrams.

7. The standardisation phase. During this stage the project enlarged its horizons at European level, joining and facing other actors, initiatives and standards. In TexSpin the results of Moda-ML and of eTexML were integrated and refined, and the feedback was collected. In this perspective, the standardisation path of CEN/ISSS required acceptable efforts to face, thanks to the simplified approach of the CEN/ISSS (addressed to the construction in a relatively short time of informal standards) and to the early adoption of a general interoperability framework (built on the past experiences of EDITEX and on the ebXML activities that were running at that time). The involvement in the CEN/ISSS TexSpin activity aimed to contribute in a standard prenormative definition for data exchange in the manufactory branch of the Textile/Clothing supply chain; the results were formalised in a CWA (CEN/Workshop Agreement 14948/2003 [25]). The three main public events were the plenary sessions in Belgium, Italy and France that viewed a strong participation of the industrial firms.

8. The dissemination and evolvement phase; this phase involves:

- Awareness set up on the developed specification.
- Integration with ERP products and developing of demo software.
- Comparison with other existing (not necessary standard) interoperability solutions adopted in the sector.
- Fostering the adoption of the project results.

Presently about one hundred firms have declared their support to the specifications.

9. The growth and extension phase. After the conclusion of the standardisation phase, the following activities aim to extend the potentiality and the scope of the developed interoperability framework. To this aim the evolution of the framework leaded to put in evidence further areas/relationships of the supply chain that require support; then, the activity of business analysis started again, producing a new set of specifications that has been prepared for a new standardisation activity.

The iteration of the cycle was completed on September 2003, in March 2005 the second iteration of the standardisation phase starts with the CEN/ISSS initiative TexWeave.

Within the life-cycles of the Moda-ML experience, phase 3 and the following ones can be considered as always "in progress", because of document and process structures need to be extended to cover new phases of the production process. The result is a repeated iteration of the phases 3-8 (iterative life cycle), with the aim to expand the coverage of the specifications from a core of high priority functionalities until the whole supply chain.

The evolvement of the project lead us to build a more and more dynamic structure, both in term of documents and of software.

Chapter 8

An ontology-based approach for B2B interoperability protocol management

8.1 B2B interoperability protocol management

8.1.1 What is a B2B protocol?

In order to allow two different systems to interoperate a language to exchange information between the systems must be defined. In the B2B scenario, this language is basically a B2B protocol that must be supported by both of the systems.

B2B protocols could be defined by private standards or proprietary initiatives, anyway they must be able to carry the appropriate information to allow the systems to communicate. The information are structured in well-defined document formats; moreover, the communication of the information between the systems must follow specific exchange sequences or rules. Then, from a practical point of view, the definition of a B2B protocol includes:

1. the definition of a set of documents used to exchange information between the involved partners.

2. the definition of a set of processes to support business information document exchange.

In some cases the second point could be ignored for the definition of a framework: the definition of the documents to exchange can be considered powerful enough to allow a basic level of interoperability between partners, and to avoid too complex protocols.

In other cases (f.i. RosettaNet), the defined processes are really very simple, and the definition of more complex and more useful business processes to implement are delegated to the enterprises, that use to this aim the customized composition of the predefined atomic processes. Finally, many complex languages can be adopted for the complete business process definition. In any case, the definition of exchange processes can be included in the definition of a B2B protocol, and can result necessary in many business relationships. In my proposal for a B2B protocol definition framework, my aim is focused on document building and management, and not on the processes, also considering the problem of their customization by the enterprises. The framework and the bounded B2B protocol management infrastructure represent a model for interoperability that can be exported in every business scenario, also when the complexity of the business processes to implement and the low technical skills and economic resources of the enterprises represent a considerable obstacle for system interoperability.

8.1.2 Structuring the B2B protocol

The aim of this chapter is not to define any specific B2B protocol, for a specific business domain. In any case, this would not be really feasible and would result in a useless effort for a single person: this task should be assigned to more authoritative subjects, like standardisation bodies or enterprise consortia, rather then to be performed by external entities like research centers or universities.

On the other hand, my aim is to outline an architecture to define and manage a B2B protocol. Nevertheless a relevant part of my work has concerned with an European initiative, devoted to the development of an interoperability framework for the Textile/Clothing sector (see chapter 7 for the details about the project); in this thesis I describe the development of this initiative linking this development with the proposed framework.

A vertical approach

One of the first decisions to take facing the definition of an interoperability framework is the approach to adopt (see section 5.1 about standards); to summarize briefly, there could be two main approaches, the horizontal one and the vertical one.

This thesis defines an architecture for the definition and management of a B2B protocol in a vertical perspective. The idea is that, especially in specific and heterogeneous business sectors with small and medium organisations, we need to provide documents tailored to match the requirements of the enterprises. Horizontal approaches, like UBL, need not only to provide too generic document templates, but also to prefigure customization mechanisms that might result finally too complex for the target users, that hardly know the underlying technologies used to implement the protocol (these customization mechanisms are even more complex if we consider the huge amount of business documents adopted in business transactions).

Document customization can in fact require (as for UBL) the knowledge of technical specifications of complex standards (like XML Schemas), and moreover imposes some limitations on the possibilities of the customization; the more a standard aims to be general (i.e. horizontal), the more it appears far from specific sector requirements, complicating customization mechanisms.

On one hand, this approach results to be tight and tangled for the enterprises, that should also find an agreement between themselves in order to develop a really wide-accepted B2B protocol; on the other hand, the vertical approach means that the framework includes some components that in their definition result to be strongly conditioned by the target production sector: a vertical approach foresees the definition of a vocabulary of specific terms, strictly related with the business sector.

Anyway, customization remains a relevant issue for interoperability frameworks[31]. Whatever is the approach adopted, in some cases (business sector with a large presence of SME) customization is needed to allow the widespread adoption of a B2B protocol. My approach prefigures the definition of a sector-specific standard designed within a standardisation process, but also foresees a set of customization tools that could aid the enterprises to easily modify, but not to upset, the defined standard within some restraints. We will talk about customization mechanisms in section 8.3.5.

8.2 The basic components of the framework

As I said, B2B protocols are built to provide interoperability. But what is ”interoperability”?. There are many definitions for this term. In the following I report two of them: the ”IEEE Standard Computer Dictionary” (the IEEE is the Institute of Electrical and Electronics Engineers) defines interoperability as.

the ability of two or more systems or components to exchange information and to use the information that has been exchanged [55]. This is a starting point, but for our purpose it is not enough. More precisely, B2B applications require to be interoperable in different perspectives: we have syntactical interoperability, semantic interoperability and model interoperability, as expressed in another definition [34]:

”Ability of different types of computers, networks, operating systems, and applications to work together effectively, without prior communication, in order to exchange information in a useful and meaningful manner. There are three aspects of interoperability: semantic, structural and syntactical.”

The last definition highlights what is required for our purpose, and, at the same time, it prompts the limitations of the proposed solutions up to now: the definition of a B2B protocol consists not only in the definition of the syntactic structure of the documents, but implies also the semantic definition of both the document set to exchange and the vocabulary; while syntactic interoperability, being fundamental, has been largely explored developing B2B protocol, what lacks in many suggested solutions is the definition of a semantic layer used to describe the conceptual model of framework, and thus the framework itself. In other words, an interoperability framework must foresee the necessary tools to define and manage a semantic layer for the framework. This semantic layer must be plugged into the architecture in order to improve its functionalities and the interoperability towards external systems, being them databases, ERP, or other kinds of systems. Up to now, semantic is not clearly described in any way and basically remains defined only implicitly; in this way it cannot be exploited to improve system interoperability: just in some cases, mapping mechanisms extract semantic models from data sources to defined mapping rules between heterogeneous data models.

The previous definition talks also about structural interoperability, that regards both the models adopted to exchange information and the formats used to present information;

Obviously, the three components of interoperability shouldn't be thought in an independent way, but are strictly related each other. Moreover, we have to bear in mind that computers are built for humans, so we have also to consider the idea that an interoperability framework should ease the human/computer interoperability. This basically means that we have to take in account of the human - machine interface (together with the machine - machine interface) for the definition and use of a B2B protocol.

In the abstract model of the interoperability framework of [11] (see chapter 3), while describes the role and the use both of the B2B protocol and the business logic management layer, there isn't a well defined "role" for the semantic description of the framework; among the basic concepts of an interoperability framework there are events, messages, processes, but there is no place for the idea of concepts, knowledge or semantic model for the information that underlies a communication protocol. Interoperability requires understandability, and a syntactical specification of a framework is not enough to target this aim. Also in other standards (see chapter 4 and 5), up to now the semantic is not considered as a integral part of the framework, but rather as an added support to integrate different visions (see 4.2).

My idea for an interoperability framework includes (even better, the interoperability framework starts from) a "semantic" layer in the framework, that could provide a vision of the concepts and knowledge managed by the framework. This layer can in some manner feed the definition and implementation of other parts of the framework, for example could be used to:

- define datatypes and document structures.
- enable back-end application integration with the protocol.
- support maintenance and development of the framework.
- develop user-friendly interfaces for protocol adoption.
- enrich the potentialities of the framework.

To coherently manage the needed information about the B2B protocol and considering also the definition of interoperability, the proposed framework is composed of three main abstract layers:

- The semantic layer, that comprises the definition of the knowledge domain that underlies the definition of the terms used to exchange information, and should specify as good as possible the semantic of the information. It must be specified (also considering the vertical approach adopted) that in general this layer couldn't aim to be the complete, universal and exhaustive knowledge representation, but should instead represent the semantic of a sectorial and specific domain knowledge.
- The syntax layer, that comprises the definition of datatype and document structures used to exchange information between partners. This component is mainly devoted to manage the syntactical aspect of the B2B protocol.
- The model layer, that outlines the business logic and the collaboration scenario of the production sector, modelling the exchange activities that drive enterprise collaborations.

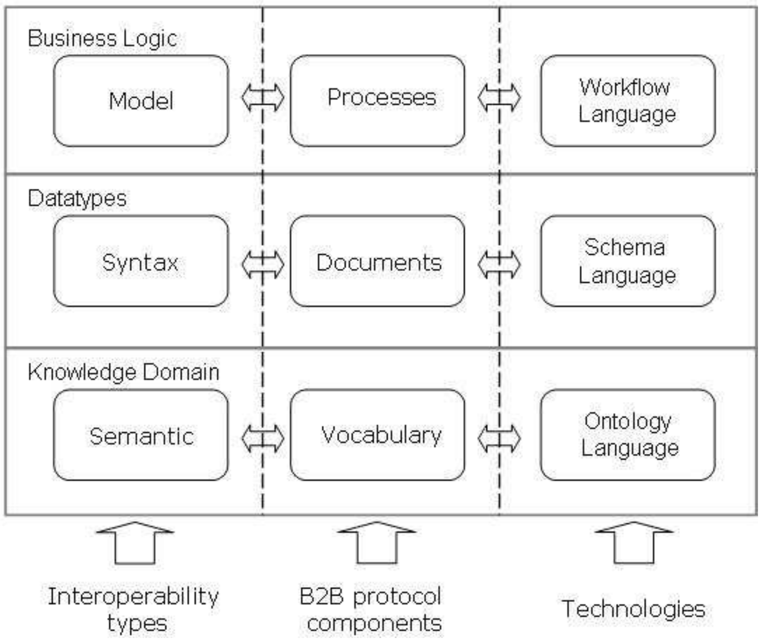


Figure 8.1: The three aspects of interoperability

As depicted in figure 8.1, each of these layers is defined to manage one of the three aspects of interoperability. Each of these layers will use different technologies (at the moment OWL represents one of the reference technologies for a semantic layer - other proposal are available - , XML Schema is a consolidated validation language to express syntax, whereas many different proposals deal with model definitions - see section 4.1.6 and chapter 5).

In respect of the architecture depicted in fig. 3.1, I introduce the explicit definition of the semantic layer upon which the connectivity and the business logic layer must be build. Basically, the documents are composed using terms, so the definition of a document layer is based on the definition of the vocabulary layer. The definition of a vocabulary of business terms represents the construction of the basic blocks upon which the B2B protocol will be built. This definition can be crucial in order to make the framework to meet the requirements expressed in chapter 6. In particular the structure of the vocabulary can influence the maintenance, usability, and scalability of the framework.

Following this idea, the process layer is based on the document layer, since the process needs to specify which documents are used, and how they are exchanged. In this perspective, this framework aims to prefigure a novel approach to build protocols and common standards: in fact nowadays the design of interoperability frameworks starts from the definition of documents in term of syntactic structures. Whereas a common vocabulary is provided (as in many ecommerce frameworks, or standards), there isn't a formal specification of the knowledge domain that underlies the vocabulary. In my vision the specification of the semantic of a knowledge domain represents the first step to which bound every other aspects of an interoperability protocol or standard.

Since the semantic layer is considered the basis for the design of the other layers in the B2B protocol its definition must be carefully analysed: the simplest solution consists in the definition of a vocabulary of business terms. Many different choices can be done to organize a vocabulary of terms [99]. In principle these choices depend from the kind and the number of relationships that the vocabulary holds: there are many types, of different complexity, of (controlled) vocabularies that distinguish upon the relationships that maintain between the terms. A second aspect to consider is how to implement the vocabulary, and which tools can be used to manage and to use it.

In order to model the complexity of the business processes or information, a taxonomy is not an enough expressive structure, since the model must describe hierarchies, equivalence relationships but also other kinds of associative relationships. A thesaurus is a controlled vocabulary that makes explicit the relationship among the concepts. It can manage both synonymic, hierarchical and associative relationships. There are some standards that cover the definition of a thesaurus: the ISO 2788 and the ANSI/NISO Z39.19. This guideline provides a generic framework and some rules to build thesauri, but are really very complex to use. The specified rules regard the grammatical form to adopt for the terms (nouns or verbs, use of plural or singular), selection of the preferred term, decision among ambiguous meaning.

One of the main reference for dictionary building is the ISO/IEC 11179[58] specification. It provides the basis for the structuring of MDR (Metadata Register). Basically, this specification introduces the idea of data elements, that represent the fundamental information units. A Data Element provides information about both the conceptual aspects and the representational (syntactical) aspects; thus in this vision they are defined at the same time. In particular a Data Element is composed of:

- Object class : a set of ideas or objects from real world, whose behaviors and properties follow the same rules;
- Property : a common feature of these ideas or objects;
- Representation : identifies the admissible values for the element.

As aforementioned, these three aspects of an element must be managed by a vocabulary in an interoperability framework. Anyway, this structure results to be too poor for an exhaustive description: the resulting model is strongly limited by the composition of a Data Element. ebXML adopts the concepts of Data Element described above for the definition of the vocabulary.

Nowadays, especially in the context of the semantic web, ”ontology” is the buzzword used to identify semantic models for interoperability; in this vision, ontologies represent the tools to provide the semantic description of specific business sectors and, consequently, of B2B protocols. They represent a step forward to enhance structures like taxonomies and thesauri.

8.3 The framework

In this section I will discuss the structure and the functionalities of the proposed framework, and the connection between them. First of all I will report a very simple, abstract definition of the framework. This description aims to be independent from the specific technologies adopted to implement and define the components whereas in the following sections, especially where talking about the presented use case, I will consider XML as the reference technology for the implementation. XML in fact allows to express data and models in a machine-readable format that can be easily identified and retrieved on the web, and it is supported by a wide set of tools for its management, whereas other specifications do not provide support for datatype identification and search.

8.3.1 A simple formal model

The proposed framework is a 5-tuple composed of:

- A *Domain Ontology* for the semantic modelling of the information of the business sector; it structures the knowledge that underlies the B2B protocol that must be defined and adopted in the sector. Part of the Domain Ontology should be defined as an extension of the KDMO (see the following point) for datatype definition.

- A *System Ontology* that defines both the model to adopt designing the domain ontology and the representation mechanisms adopted to represent the information modelled in the domain ontology. This ontology is then divided in two subparts: the *Knowledge Domain Model Ontology* (KDMO) and the *Datatype Ontology* (DO)and it is designed to solve the issues about the connection between the semantic and the syntactical components of the framework; basically, it then acts as a bridge between the semantic modelling and the syntactic description of the data for the framework. Since the KDMO and the DO are strictly interconnected, they are defined and described together.

- A *DataType Library* (Format Library) used to exchange data. Each type defined in the library must be unambiguously identifiable and accessible: this the basis for the definition of the syntax and to compose document templates used to exchange information. In specific contexts data types can be defined not only to model information, but also just to structure the information. To manage this situation, the Datatype Library is considered as a 2-tuple (A,B) where A is the subset of types that can carry some semantic information , whereas B is the subset of types that haven't semantic implication.

  With this distinction, the framework also requires that each datatype defined in the Datatype Library that carries semantic information is bounded with a concept modelled in the Domain Ontology. This means that the datatype represents the syntactical representation of the related concept.

  On the other hand, there is no need to connect the data types belonging to the B subset of the Datatype Library to the semantic model defined in the Domain Ontology.

- A *Document Set* that can be used to exchange data. This means that this documents can be adopted within some exchange process. Each document is naturally built composing the data types defined in the Datatype Library. In the XML context, the documents can be viewed as particular data types, and than they are considered also as part of the Datatype Library. Since documents exploit the definition of a common library and refer to a domain ontology, they cannot define own data types. The Document Set design should then follow a specific document programming style.

- A set of *Exchange Processes* that can involve two or more business partners. The information exchanged during the processes are carried by the documents defined in the Document Set.

These five components can be put in relation (see fig. 8.2) with the abstract architecture depicted in fig. 8.1.

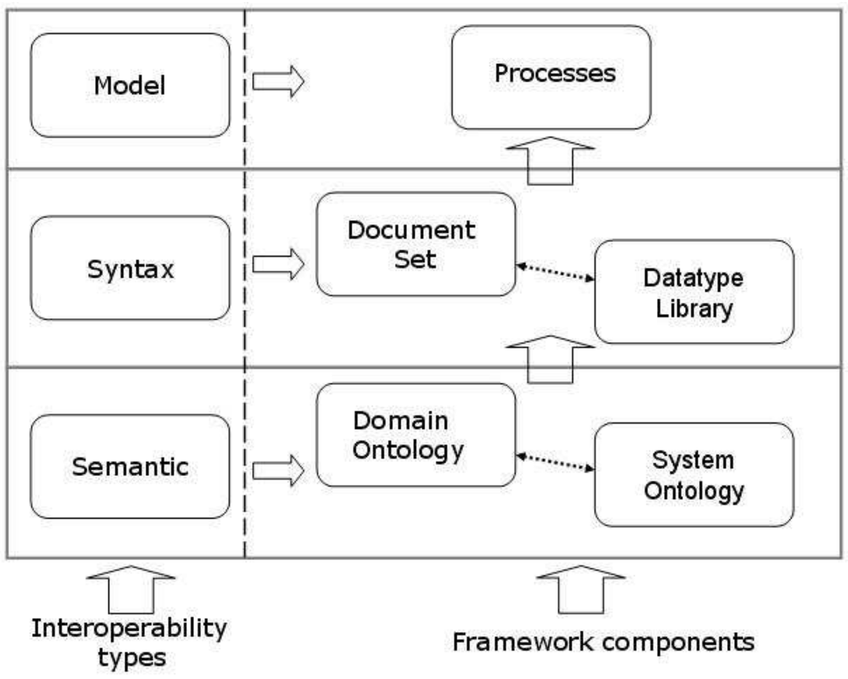


Figure 8.2: The five components of the architecture

The defined framework impacts both the life cycle and the software architecture to define B2B protocols.

In the following sections I will describe more in detail four of the five components of the framework just mentioned: in this chapter I will not face more Exchange Process Management. The intention is to provide a comprehensive vision about their roles and functionalities, and the connections that can be implemented between them. This explanation will be provided together with an example for the application of this approach to manage B2B protocol. My use case regards the Textile/Clothing sector (see chapter 6 for more details about the context of the work), where many different and complex types of data and documents are exchanged between the enterprises.

Note that in principle the Datatype Library should collect types defined following different standards and perspectives (for example types defined with the EDI or the XML format, that are substantially different), but in this thesis I will concentrate on XML data formats. XML technology is also used to implement the ontology and the stylesheet, although other languages can be used. In any case, differently from other standards, XML allows, using the namespace, to easily connect the components of the framework. Moreover, my use case aims to define a library of XML datatypes for the target business sector.

In the example, in particular I will consider the definition of a document to exchange information about the characteristics of a fabric, the TEXSheet document. This document may be considered as the Identity Card for a specific fabric, and the information included in the TexSheet document result naturally very relevant in commercial transactions; often enterprises have their own perspectives about the treatment of such information and the integration of such perspectives is needed. Moreover, in a real complex production sector like the Textile/Clothing one, both the amount and the complexity of the information available to describe a fabric could be very considerable. This example will be used throughout the chapter to show the functionalities of the proposed framework. The approach presented in the following could naturally apply to other kinds of documents (like Orders, Invoices, Catalogues and so on).

8.3.2 The domain ontology

The first step to provide a B2B protocol for a specific business sector is the ”specification of the conceptualisation of the knowledge domain”, in other words, the definition of an ontology (the Domain Ontology) for the sector. The Domain Ontology can also be considered as the first element that is plugged into the framework to generate all the other components.

The role of the Domain Ontology in the framework is then twofold: on one hand it must structure the knowledge of the domain that underlies the B2B protocol: this means to expose sector specific knowledge by defining concept meanings through semantic relationships and particular constraints; in this vision the ontology mainly concerns that part of a well-defined knowledge that tends to remain unchanged for long time or in any case rarely changes. On the other hand the domain ontology should represent the basis upon which a set of datatypes for the B2B protocol itself can be build .

Providing a semantic description for a protocol mainly carries three results:

- A better comprehensive understanding of the protocol itself (for enterprises and software houses but also for internal framework staff). This eases the support and maintenance operations of data models in the ICT context of the sector (B2B Framework maintenance and management);
- Ease the transition of the enterprise to the new formats by only defining semantic concept-to-concept mappings and doesn’t force the enterprise to a particular rigid structural model.
- Let the framework interoperate in an horizontal manner with the semantic layer of other sectorial frameworks (i.e. Semantic Mapping);.
- Provide a basis in Semantic Web context applications (Semantic Web Services, Horizontal B2B Portals, Semantic Annotation).

Though there is not yet a formal definition of ontology engineering process ([62]), I want here to fix some requirements that must be taken into account during ontology design in order to keep the final ontology the more flexible and multipurpose as possible, and, above all, suitable for the development of a usable B2B protocol:

- The ontology must be exhaustive about the description of the domain of interest. Therefore the ontology should be the more complete as possible and without lack of information. To this aim, the definition of the ontology must be done by domain experts, or, eventually, by technical employees that work strictly in contact with them.
- The ontology must be consistent; this means to have no contradictions between the defined concepts.
- The ontology must be as specialized and vertical as possible and it should not model more abstract and general concepts (i.e. product, business, artefact, etc.) that go over the target domain. These concepts should be defined separately by other bodies.
- The ontology must be both modular and extensible in order to allow future (inevitable) developments and an easy integration with other ontologies.

The ontology of the example is built up and filled starting from many information sources strictly related with the target domain. These information sources are represented by both document collections and human experts.

The Ontology Developer and Manager describes the Textile/Clothing knowledge especially related to the properties and characterizations of materials like fabrics. Basically, the ontology may be modelled using whatever available ontology editor and knowledge-base frameworks that can export the ontology in OWL format, that is the W3C standard to define ontologies, and can be easily integrated with other XML technologies. In this sense the tools used to model the Domain Ontology is a secondary issue. In my example, the ontology has been modelled and filled manually using Protégé [94], that provides a set of useful tools and code libraries for developing, editing and maintaining complex ontologies.

A further important aspect to remark about ontology based architectures is the capability to apply reasoners. Exploiting the Description Logic (DL), on which ontologies are mainly based, reasoners can infer (reason) from the knowledge formalised in the ontologies, additional information that are not directly held by the ontology. This represents an important research field that could be applied also in a semantic e-business scenario like collaboration frameworks and the Data Integration issues.

The main language used to represent ontologies in the XML technology scenario is the standard one defined by W3C called OWL (Web Ontology Language). This Description Logic based language is now becoming more and more popular and used in many contexts, so there is a great number of open-source and free tools and environments for supporting the ontology building and management. From OWL descends also a set of other languages and formalisms to express semantic rules and queries over OWL ontologies.

The OWL language is defined in three versions (OWL Lite, OWL DL, OWL Full) that differ in the expressivity they provide. In particular OWL Lite is the less expressive version, whereas OWL Full is the more expressive. They have been designed to match different requirements by the communities modelling knowledge domains.

In particular, OWL Full has the maximum expressiveness, but does not provide computational guarantees. This means basically that reasoning softwares will not be able to support complete reasoning on OWL Full ontology. In the proposed framework I want to avoid this limitation, in order to have ontologies that can also be exploited using reasoners. So the ontology must stay in OWL DL.

In order to also generate a set of XML datatypes for the information modelled in the domain ontology (and we want to generate these datatypes in a semi-automatic way), part of the domain ontology must be designed following the model defined in the KDMO ontology (see section 8.3.3).

Basically, the part of the domain ontology that reflects the model of the KDMO can be translated into a set of datatypes. On the other hand, that part that does not follow the KDMO model will not be mapped in datatypes, but still can be used to obtain additional semantic information to the datatypes and the documents defined for the B2Bprotocol. The domain ontology thus maintains at the same time both a part to generate the datatypes, and a part to add semantic information to the datatype

During the generation of the datatypes the framework maintains a connection between the identified concepts of the knowledge domain (that are the information exchanged) and their representations with datatypes.

Since the ontologies of the framework are defined using OWL DL, the consequence is a strong separation between the classes and instances. In particular the domain ontology must be designed in a way such that the information exchanged is modelled as more as possible as instances and not as classes. The use of instances allows to assign specific properties. If the information is modelled with classes, it is not possible to specify instances as value of an object properties.

In the proposed example I want to define a document (the TEXSheet document) to exchange information about the characteristics of a fabric so the first step is the design of an ontology that could model the needed information to exchange.

In this case, I have first of all modelled the textile products to be described, creating the *textileProduct* class; this class is the root of a taxonomy for the characterization of the different textile products. Then I have modelled the properties I need to provide information about my products;

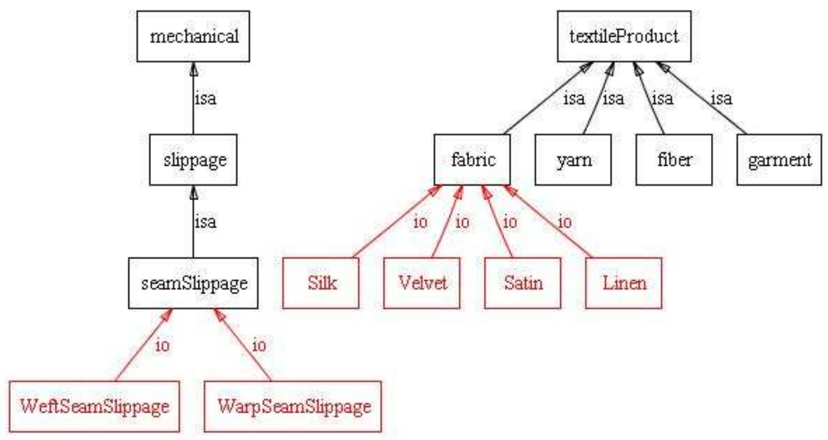


Figure 8.3: A graphical representation of a fragment of the domain ontology

therefore, I have created the *textileProperties* class; this class is the root of a taxonomy for the characterization of the different textile properties. One of the subclasses of the Textile Properties are the *mechanical* properties.

Fig. 8.3 depicts a graphical representation (using the Ontoviz plugin of Protègè [90]) of a very short fragment of the domain ontology defined. Because of the huge complexity of the whole ontology, it is not possible to visualize it completely; then, the representation lacks of the majority of the classes, instances and properties, and is proposed just to give an idea about the proposed approach.

For example, the instances of the class *Fabric* are nearly 40 (I show only the instances Velvet, Satin, Silk and Linen), and only few subclasses are reported for the textile properties. Basically, the ontology adopted should define taxonomies of the concepts we want to manage in the documents (like a taxonomy for textile products - like fabrics - and their properties). In case the concepts and the related properties are very heterogeneous, they should be modelled in different sub-ontologies, also to ease the reuse of them. Each concept that must be mapped in data format for information exchange within a protocol must be instantiated. In the example above the *WarpSeamSlippage* and *WeftSeamSlippage* are instances of the *seamSlippage* class, that represents a specific property of the fabrics [1].

In this phase the Ontology Manager does not define any structure for the documents, nor she defines any XML Schema types, that is delegated to a following stage. The Domain Ontology def-

[1]The slippage of yarns in the fabric along a seam when stress is applied. The result is that the yarns pull out but the thread and the stitch doesn't rupture. Seam slippage is usually caused by poor fabric design (too loose of a weave) or too narrow of a seam margin. Not using enough stitches per inch and a poor stitch balance can also contribute to seam slippage.

inition, performed by the Ontology Manager, is strictly related with the adoption of the KDMO, that I will describe in section 8.3.3 to complete the description of the ontology building procedure.

The definition of the Domain Ontology does not need to be definitely concluded before the document building, because the definition of the documents can naturally require some modification to the Domain Ontology. For example it is surely possible to add subclasses/instances in the ontology, but it is not possible to restructure a taxonomy tree; in other words, the Domain Ontology must not substantially change, it can grow up but its basic structure must be preserved. Since the definition of the other components of the framework strictly depend from the ontology, substantial modifications on the ontology itself could result in the upsetting of the related datatypes and documents.

Finally, in order to make the ontology flexible and exhaustive a set of tools for the ontology maintenance and browsing is needed.

8.3.3 The System Ontology

One of the main features the framework aims to provide is the connection between the semantic description of the Knowledge Domain and the syntactical representation and data format of the information modelled. Especially in the context of the XML technologies, another important purpose is to allow a semi-automatic generation of a library of datatypes to represent the information modelled in the ontology. To this aim the proposed framework includes a System Ontology that defines both a model for the modelling of the knowledge domain (represented by the Knowledge Domain Model Ontology), and a Datatype Ontology (that can be expanded) to structure the datatypes of the framework. These two sub-ontologies are strictly interconnected and thus they are presented together: they represent the bridge to connect semantic definition tailored for specific business domain with a common-sharable set of data types.

The connections are two-way, and are held in the Datatypes Ontology (versus the datatype definitions) and in the XML datatypes definition (versus the concept definitions). This idea is depicted in fig. 8.4.

Using the XML technologies in fact it is possible to link directly the XML Schema datatype formal definition with the concepts in the ontology, whereas, for other formats (i.e. graphical format) this link is not possible; I will explain better the linking mechanism in section 8.3.4. In these cases the DO maintains a proper description of these formats as instances of specific classes of datatypes.

The Knowledge Domain Model Ontology - KDMO

The KDMO is the model that must be adopted designing that part of the domain ontology that should lead to the generation of a datatype library. This model is based, like the RDF model,

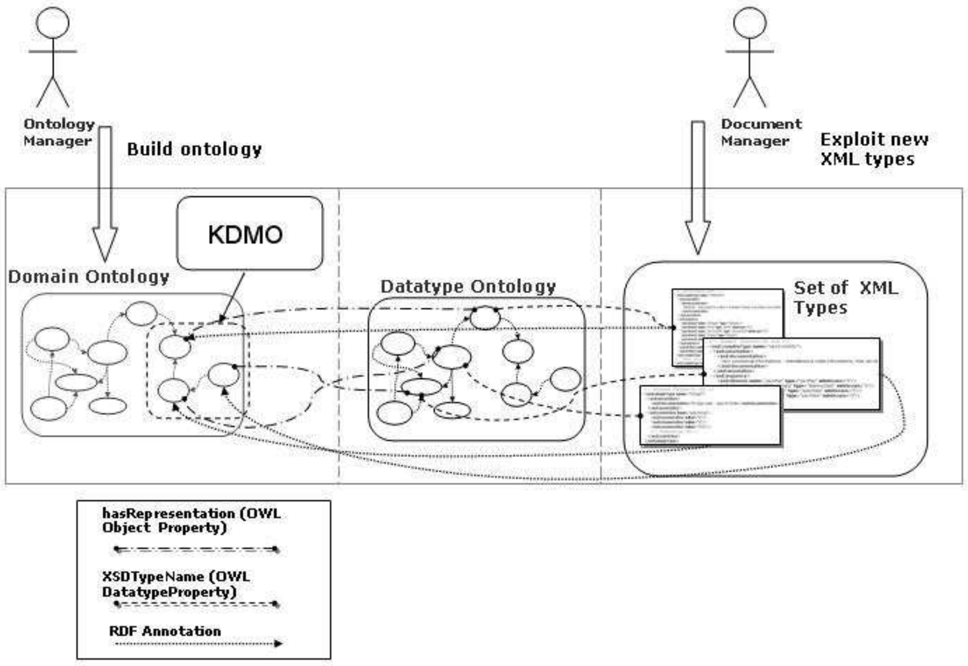


Figure 8.4: Connection between the semantic models and the datatypes

on the use of triples. Fig. 8.5 shows the basic structure of the model designed for the domain ontologies: it depicts the root classes of the KDMO and the main relationships between them. The picture does not depict subclassing relationships.

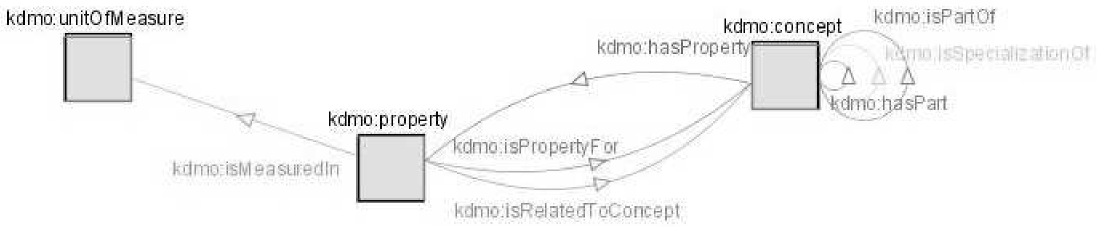


Figure 8.5: The KDMO Ontology

The model is really simple, and requires the identification of the information modelled in the Domain Ontology as subclasses of one of the two classes in the KDMO. The two classes that are defined to model and compose the information of a knowledge domain are:

- The class *concepts*: every information that is identified within the knowledge domain and need to be mapped in some representation format by the framework is a concept. Naturally, the information (and the related sub-concepts) can be organized in hierarchies and taxonomies, that can be wide and complex without any limitation. In a similar way of the RDF model, in our model, the concepts are not representable on their own, in any manner; in other words, the concepts cannot be merely described and then they cannot also have a value. A concept can be described only through its properties.

- The class *properties*: this class is a sub-class of the concept class (this relation is not shown in fig. 8.5) and includes those concepts used to describe other concepts. Every concept identified within the knowledge domain that is a property of another concept is in the *properties* class. In this model, also the properties of a concept are modelled as classes, and not using Object Properties (that are defined in the OWL Language). In the KDMO the class *concept* is connected with the class *property* with a simple Object Property *hasProperty* (and there is also its inverse Object Property, the *isPropertyFor* property. In this vision, modelling the structure of the knowledge domain, OWL restriction can be defined on the *hasProperty* property to link specific concepts with their specific properties (that are classes or instance). For example, if in the domain ontology a concept A (that is a class or instance) must be described by a concept B, A is defined as of type *concept*, B is defined as of type *Properties*, and a restriction is applied on the *hasProperty* property to specify the connection between A and B. As for the concepts, the properties can be organized in hierarchies and taxonomies, that can be wide and complex without any limitation. Both the concepts and the properties can be mapped to a datatype. To this aim the model defines an ObjectProperty named *hasRepresentation* that connects instances of the *Properties* and *Concepts* class with instances of the *Representation* class. In this way it is possible to specify a representation mechanism for a specific property or concept.

- the class *unitOfMisure*, that is a subclass of the property class. We observe that the description of the properties of a concept if often done specifying a unit of measure to correctly evaluate the values of a property. Therefore, once a textile property has been identified, it is possible to specify into the ontology which unit of measure is used to evaluate the property itself (for example, DaNewton could be the unit of measure for the traction resistance of a fabric). This part of the ontology impacts the generation of the datatypes and allows to distinguish among numeric types and text types.

So this class allows to connect the properties with their specific units of measure. I want also to remark as the representation and the unitOfMeasure are substantially different and both of them are needed to properly describe a property.

In the context of XML technologies, the KDMO is the model to adopt by an Ontology Manager that wants to develop a Domain Ontology to build a B2B protocol for a specific sector. Once the Domain Ontology is plugged into the framework, it will be put in relation with the KDMO model.

In this task the ontology manager should therefore:

- identify the basic information of the Domain Ontology that must be treated and described within the B2B protocol. This information will be classified as subclasses or instances of the class *concept*. After this task, the Domain Ontology concepts inherit the *hasProperty* property, that is used to link the concepts with their properties.
- identify the parameters/dimensions/properties to describe the identified concepts. In this way the defined properties will inherit the ObjectProperties *hasRepresentation* that will allow to specify a representation mechanism included into the Datatype Ontology. The representation are not managed in this stage, but will be defined later.
- fix some restrictions on the OWL ObjectProperty *hasProperty* between the concepts and the properties of the defined ontology. It could also mean to transform some ObjectProperties in the origin Domain Ontology in restriction on the *hasProperty* property.

In fig. 8.6 I show how the model has been applied to the fragment of the Domain Ontology:

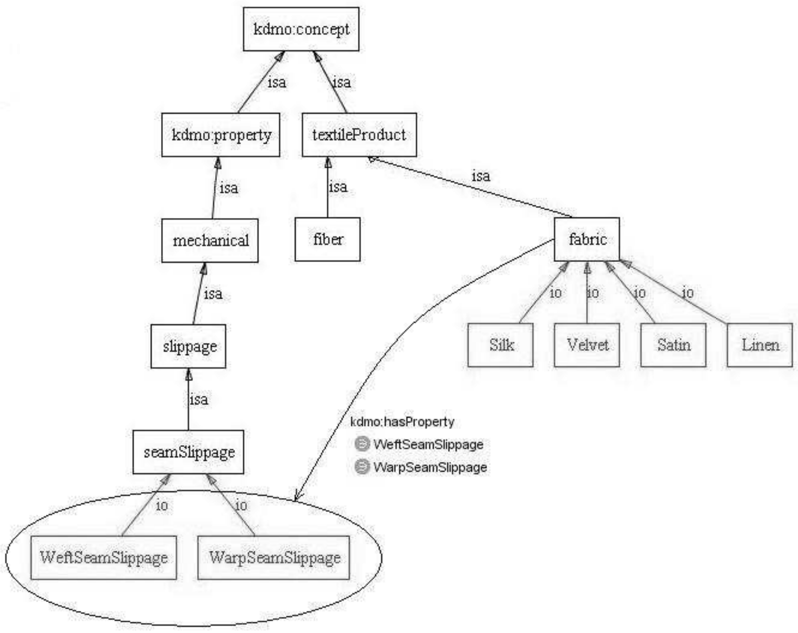


Figure 8.6: The application of the KDMO to the Domain Ontology

In this case the class textileProduct is the root concept that should be described in the protocol we want to build (in other words, it represents the information we want to communicate), and than it becomes a subclass of the class *concept*. A textileProduct is described with its properties (here is shown only the class *mechanical*) that are then modelled as subclasses of the class *property*.

Finally, an *hasValue* restriction has been put on the *hasProperty* property to specify which are the specific properties used to describe a fabric. As said before, this is only a fragment for the explanation of the approach followed.

The Datatype Ontology - DO

The *Datatype Ontology* is the second relevant part of the System Ontology. While the KDMO shows the model to follow building a specific domain ontology to generate datatypes, the Datatype Ontology is used to structure the data formats, to identify data type specifications and to add some other information to manage data type interoperability.

The basic structure of the Datatype Ontology is depicted in fig. 8.7, where the main classes and their relationships are depicted. The picture does not depict subclassing relationships.

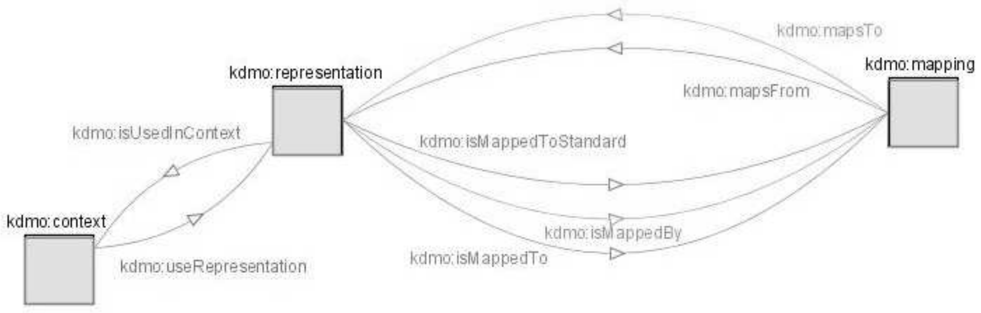


Figure 8.7: The basic structure of the Datatype Ontology

The three main classes defined are:

- The class *representation*: each object (both instances or classes) that identifies a representation mechanism used to represent a *property* or a *concept* is a *representation*. This class is the root of a taxonomy defined to structure the datatypes and formats formalised to represent and to manage the different kinds of information identified in the domain ontology. The idea is that each data type or format used to represent an information must be identified by an instance of a class that is a sub -class of *representation*. In our view, this ontology represents a basis that can evolve to comprise many different, novel and heterogeneous types of data that are defined in time. Moreover, this ontology does not want to be the reference ontology about datatype, nor claims to be the right one. So in its definition it is just a starting point to show the functionalities of the proposed framework and needs the contribution of document/data type developers to be improved. It is important to remark that the aim of this ontology is not to provide the full specification of a specific datatype (there are other more proper tools and ways to obtain this task), but just to identify these datatypes and to provide a link to them in order to allow users to check data and to perform validation with them. Since the framework generates in a semi-automatic manner a set of XML datatypes from the domain ontology, the related instances of sub-classes of the representation class are managed automatically. Otherwise, if the representation format is not XML, the instances must be defined manually, together with their characteristics. The ontology defines also the *hasStandardRepresentation* property to identify those representation mechanisms that are part of an approved standard.

- The *context* class: many datatypes are defined, adopted and used only in specific working context. This class is the root of a sub-ontology for the specification and identification of such use context.

- The *mapping* class: the different data types and formats defined can be translated each other to obtain interoperability between heterogeneous models or systems. The idea is to provide a support to data interchange identifying the mapping mechanisms available to the different data formats, and to provide links to them. An instance of the mapping class represents a mapping mechanisms. The *isMappedBy* property allows to specify for a specific starting instance of a representation (that represents the format we want to translate) which are the mapping mechanisms defined for the representation. The *mapsTo* property allows finally to retrieve the target representation formats managed by each identified mapping mechanism. Also the inverse properties have been defined for the two-way connection among the starting and the final formats. Mapping mechanisms can be different, starting from XSLT templates to web services online, to ASP services, web applications and so on. As in the previous case, the mapping class can be considered as a starting point for the definition of a sub-ontology that could structure the scenario of the mapping mechanisms.

The Fig. 8.8 shows the overall structure of the main classes of the System ontology.

Developing the Datatype Ontology, it is surely possible not only to create a taxonomy of datatypes and formats, but, in order to well manage each specificity of the formats, it is possible also to define proper Datatypes Properties. In our use case, we treat basically XML datatypes. So, we have created a subclass named *XMLRepresentation* that is subclass of the *StructuredRepresentation* class. We have then added to this class those Datatype properties needed to manage the characteristics of XML Datatypes (like *XSDTypeName* and *XSDTypeNamespace* that are used to specify the type name and its namespace).

Fig. 8.9 shows the link between the information modelled in the ontology and its representation: the instance *WarpSeamSlippage* of the Domain Ontology is linked with the instance *MPseamSlipWarp*, that represents a specific XML type, in the Datatype Ontology using the *isRepresentedBy*

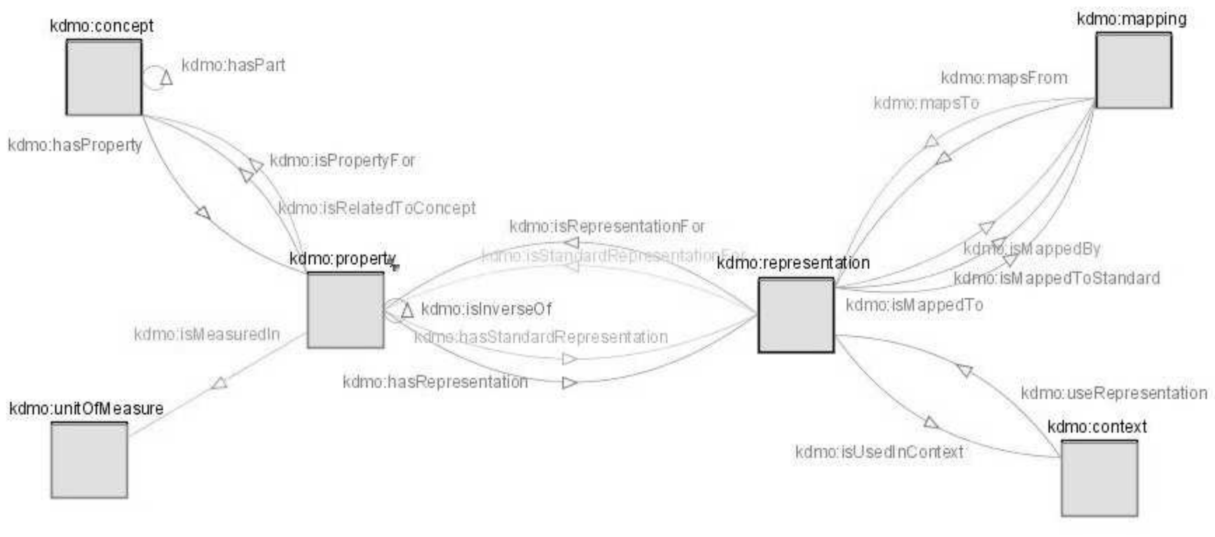


Figure 8.8: The infrastructure of the System Ontology

property. This instance allows the finding of the corresponding type definition according to the value of the *XSDTypeNamespace* and *XSDTypeName* properties.

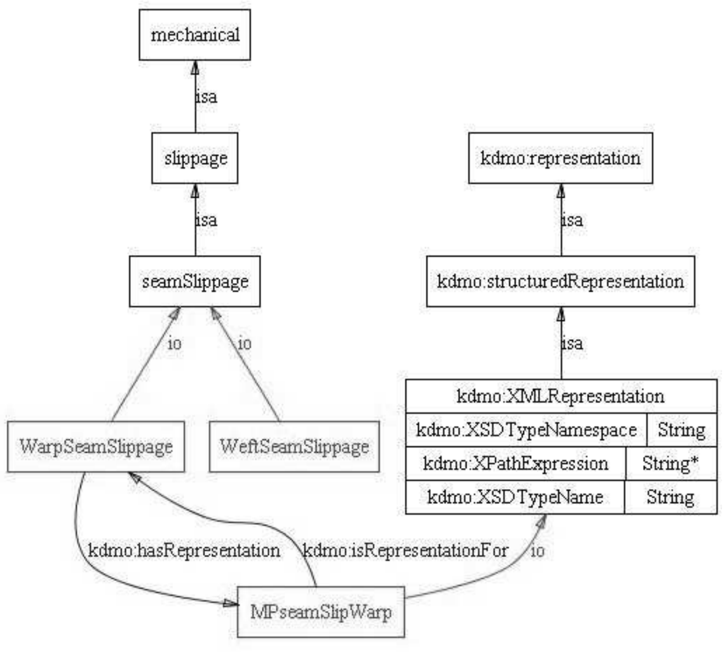


Figure 8.9: Link between a concept and its representation instance

8.3.4 The DataType Library

In a framework for the definition of a B2B protocol, obviously the definition of a domain ontology is just a first step to develop the protocol. The protocol to exchange information is basically constituted by a set of data types that must be adopted in building business documents and messages.

If the ontology should be associated with a set of non XML types, this types are represented only using instances of the Datatype Ontology that should exploit specific defined Datatype Properties to describe the adopted formalism . In this case, the framework does not provide any automatic mechanism to generate and manage the type definition and the link with the ontology. The framework just allows the finding of the definition of such types in the Datatype Ontology.

On the other hand, the proposed framework prefigures a mechanism to generate a set of XML types defined using XML Schema to exchange the information previously modelled: once the knowledge domain has been described in the ontology by the Ontology Manager, it is possible to associate a set of types to the identified concepts and properties. In the following discussion I will consider only XML types.

The aim is then to create in a semi-automatic manner a library of datatypes that can be used to build documents and protocols. There are several aspects to consider defining the datatype library: in the following sub-section I describe each of them.

Datatype sub-sets definition

In some cases it is possible to identify in a library different kinds of data types. For example there could be primitive or basic datatypes, together with aggregated/complex datatypes. In this case, since the definition of the datatypes derives directly from the semantic vision of a specific knowledge domain, it is not possible to define an ”a priori” classification or categorization of the datatypes, (like in UBL), nor is possible to define specific dependencies. These dependencies are implicit into the ontology definition, and are then made clear with the generation of the related datatype library. Moreover, taking a vertical approach it is useless to write a predefined set of types, and this approach would add too much complexity to the library structure, complicating the understanding, adoption, and usage of the protocol itself.

Datatypes can also be collected on the use context of the protocol, but this information is maintained in the domain ontology exploiting the *hasContext* property.

On the other hand, I want to distinguish among other two different ”categories” of types.

- Types that carry semantic information strictly related with the knowledge domain. I will call these types ”semantic” types.

- Types that do not carry semantic information, and are only defined for formatting purpose or as containers of other types. For example their could be a type ”Header” that is just used inside a document to collect information in a specific part of the document, but that does not represent a specific categorization of the contained data. I will call these types ”formatting” types.

In my vision, the datatype library should be composed of only semantic types. Datatype generation must avoid the specification of formatting datatypes that are used only for formatting purpose. To this aim the framework foresees other more proper tools that manage formatting/presentation issues for data.

Then, the building of datatypes within the framework will not lead to the definition of ”formatting” datatypes that, if required, must be defined and managed separately. Naturally, in any case only the ”semantic” datatypes will maintain links with the domain ontology.

Programming style

In order to build a library of datatypes, and considering in our case to adopt the XML technology for its implementation, it is relevant to analyse and decide the programming style for the target XML Schema that will come out from the ontology.

XML Schema provides several mechanisms to define types and elements for the documents. Basically, the differences among these approaches are in the possibility for the programmers and document managers to reuse, to modify and to extend pre-defined types and elements into other schemas that want to import the data defined in the library. The four approaches available are (I will summarize briefly their characteristics):

- Russian Dolls: both element and type are defined as anonymous; this means basically that they both can’t be reused in external schemas
- Salami Slices: element definitions are global, but types are declared anonymously. In this way, the types are not reusable, but it is possible to adopt the defined elements (but elements cannot be extended in the external schemas).
- Venetian Blind: all the types are named and declared globally, but only the root element (in a document definition) is declared as global. The other elements are declared locally. In this way it is possible to reuse (and extend) the types, but not to reuse the elements.
- Garden of Eden: both the elements are declared globally and the types are named and there aren’t local declaration. In this way they can be both reused and types can be extended.

The basic motivation to build a datatype library is in the possibility of the reuse of the defined types in different documents, and also in the possibility to easily extend them. In this perspective it appears clearly as the first two approaches (the ”Russian Dolls” and ”Salami Slices” ones) are incompatible with the requirements of the framework. The choice is then between the ”Venetian Blind” and the ”Garden of Eden” styles. Whereas the ”Garden of Eden” seems to be more powerful in respect to ”Venetian Blind” (since it allows element reuse), it add a too complex and cumbersome management structure for the definition of the elements; also the readability of the schemas will be reduced. The idea is that the framework can provide a set of datatypes with a well-defined semantic, but don’t want to impose too strong conventions on element usage. Element definition is then partially entrusted to document builders. On the other hands, element mapping could be performed on the basis of the common datatypes used and their references to the ontology. Then the adopted programming style is the ”Venetian Blind”.

Datatype building mechanism

The building of the datatypes is performed in two stages:

- An automatic stage, where there is a first formalisation of the datatypes. In this phase an automatic tool exploits as more as possible the definition of the domain ontology to deduce the more proper syntactical basic structure for the datatypes. In this phase pre-defined patterns and rules are applied to translate the structure of the ontology in the structure of the datatypes; moreover, the specification of the unit of measure can help to infer which datatype should be adopted; for example, if some properties is measured in meter, the corresponding datatype will be a numeric one. On the other hand, if no information of this kind is provided about a concept it will be mapped to a text type. Nevertheless, we don’t want to mix the semantic definition with the syntactical one, but only to provide a strong connection between them; so the idea is still to avoid to explicitly specify syntactical information, whatever they could be, into the domain ontology modelled by human. This basically leads to the impossibility to produce a complete syntactical definition about the datatypes. For example, if a concept should be represented using a string 10 characters long, this information should not reside in the ontology, nor is possible to deduce this information in some way. The tool is entrusted to mark these ”syntactical” holes (with a specific syntax) for their following definition. The tool is also entrusted to build the connections between the domain ontology and the datatype library (see also *Link with the semantic definition*). The result of this stage is a temporary, incomplete definition of the datatypes. These datatypes are not ready to use.

- A manual-guided stage, where a human can fix the so called ”syntactical” holes that are marked in the automatic stage. A tool analyses the datatype library and presents the problems to fix to a human personnel than can complete the definition of the datatype exploiting a user friendly interface that allows to manage specific feature of the target datatype specification language (in this case the XML Schema Language). For example, the tool can recognize the need to specify a string length or pattern and than requires this information from the human personnel.

All the syntactical information that are manually added during this stage to complete the definition of the datatype library must be stored to be eventually reused, after following modifications of the Domain Ontology, to re-build the datatype library itself in following versions, without the reinsertion of such information. In this way it is possible to simplify ulterior datatype generation exploiting the stored information. This information is automatically included into the Representation Ontology, and its management is entrusted completely to the tool that manages the information provided by the human personnel to fix the ”syntactical” holes in the ontology. According to the principle of separating in the framework the semantic and the syntactical perspectives, this information must impact only the Representation Ontology, but should not be managed by the Domain Ontology. The second relevant activity in this phase is the identification of the enumerated datatypes with the related values (see section ”Link with the semantic definition”).

During the automatic stage some rules and patterns are applied to transform those definitions of the origin ontology that are compliant with the KDMO in the datatypes library. Basically, the main rules are:

- Each class of the ontology that is also a *concept* (in other words, that is a subclass of the class concepts of the KDMO) and that has some restrictions on the property *hasProperty* is mapped to a complex type. In similar way, each instance of the class *concepts* of the KDMO that has a value for the *hasProperty* property, is mapped to a complex type. The content model of the generated type is defined by the set of properties assigned to the starting concept. The corresponding properties are defined as simple types, but are structured in the complex type following the taxonomy of the properties themselves.
- If a starting concept is the root of a taxonomy, its complex type includes a special element *type* that has an enumerated types. The values of this type are given by the children of the starting class, regardless if they are classes or instances.
- Each datatype property of the ontology concerned with a concept or a property is mapped to an attribute for the corresponding complex/simple type.

The rules listed above are the fundamental ones. The generation of the library must take into account more complex situations to manage.

Starting from the simple model shown in Fig. 8.6, the automatic tool will first generate:

- 1 complex type ”fabric”

- 2 simple types ”WeftSeamSlippage” and ”WarpSeamSlippage” to describe the properties of the fabric

The 2 simple types generated are then structured following their taxonomy, and then other three complex types are generated and nested following the taxonomy structure: ”mechanical”, ”slippage” and ”seamSlippage”.

Fig. 8.10 depicts a graphical representation (built using XML Spy) of the structure of the generated set of types, starting from the simple model in fig.8.6.

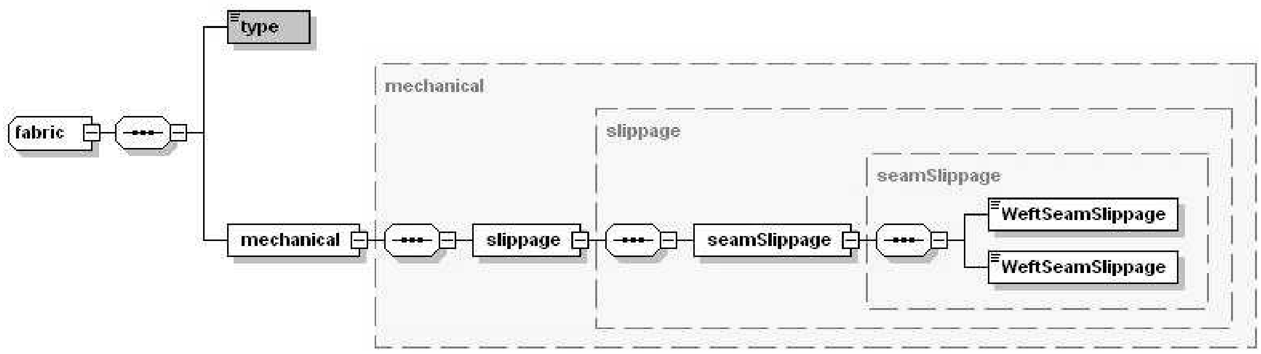


Figure 8.10: A graphical representation of the structure of the generated types

The corresponding XML Schema code is:

```
<xsd:complexType name="fabric">
  <xsd:sequence>
    <xsd:element name="type" type="fabricType"/>
    <xsd:element name="mechanical" type="mechanical"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:simpleType name="fabricType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="Silk"/>
    <xsd:enumeration value="Velvet"/>
    <xsd:enumeration value="Satin"/>
    <xsd:enumeration value="Linen"/>
  </xsd:restriction>
</xsd:simpleType>
```

```
<xsd:complexType name="mechanical">
  <xsd:sequence>
    <xsd:element name="slippage" type="slippage"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="slippage">
  <xsd:sequence>
    <xsd:element name="seamSlippage" type="seamSlippage"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="seamSlippage">
  <xsd:sequence>
    <xsd:element name="WeftSeamSlippage" type="WeftSeamSlippage"/>
    <xsd:element name="WeftSeamSlippage" type="WeftSeamSlippage"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:simpleType name="WeftSeamSlippage">
  <xsd:restriction base="xsd:decimal"/>
</xsd:simpleType>

<xsd:simpleType name="WarpSeamSlippage">
  <xsd:restriction base="xsd:decimal"/>
</xsd:simpleType>
```

In the example above, the specification of the xsd types for the simpleType (i.e. string , that is used for the WarpSeamSlippage) are delegated to the manual phase.

Link with the semantic definition

In order to improve the connection between the semantic modelling of the knowledge domain and the definition of a syntax for the B2B protocol, the framework foresees a mechanism to link these two components. On the other hand, the management of such links depends on the technologies adopted for the datatypes. In particular,

1. If XML datatypes are used, it is possible to link directly the definition of the datatypes with their semantic modelling, and to exploit automatic tools to support this connection

2. If other non-XML types (like image format) should be defined to represent the information of the ontology, these datatypes cannot be directly connected with the formal definition of the types, but must be described by instances in the ontology and managed manually.

Again, I will consider in the following only XML types. During the generation of the library, for each new type the automatic tool:

- creates an instance of the XML representation in the Datatype ontology specifying in the proper Datatype Properties the name and the namespace of the new type (as depicted in fig. 8.8).
- annotate the generated XML Schema code, adding RDF assertions to express that the new type is a resource and it is representation for a concept in the Domain Ontology. Semantic annotations are reported within the definition of the xsd types, into the *annotation/appinfo* schema component. For example the following code:

```
<xsd:simpleType name="WarpSeamSlippage">
  <xsd:annotation>
    <xsd:appinfo>
      <rdf:Description rdf:about="http://.../TexSheet.xsd#WarpSeamSlippage">
        <ont:isRepresentationFor
          rdf:resource="http://.../Ontology.owl#WarpSeamSlippage"/>
      </rdf:Description>
    </xsd:appinfo>
  </xsd:annotation>
  ...
</xsd:simpleType>
```

  expresses that the WarpSeamSlippage simple type is a representation of the concept WarpSeam-Slippage modelled into the ”Ontology.owl”. In this way each XML Schema contains also the links to the semantic modellisation of its XML types.

In the context of datatype management a specific approach is taken to manage enumerated datatypes. In fact, in order to provide a complete semantic description of the datatype library it is relevant to model the values of an enumerated type. Let see the simple following example:

```
<xsd:simpleType name="country">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Afghanistan "/>
      <xsd:enumeration value="Albania "/>
      <xsd:enumeration value="Algeria "/>
      ...
    </xsd:restriction>
</xsd:simpleType>
```

In this case it is useful to organize the values of the enumeration into an ontology about countries (for example for mapping purpose). To do this, it must be possible in the manual phase to specify as *enumerated* the datatype and to link it with an existing concept in an ontology. In the example, the idea is to connect the simpleType *country* with the concept *country* defined in an ontology. The enumerated datatype is finally built and each single value is connected using semantic annotation with a specific subclass or instance of the concept *country* in the related ontology.

8.3.5 The Documents

The generation of the type library is just the first step in order to build a set of documents to exchange business information. Classic examples of business documents of a B2B protocol are the Purchase Order, Catalogues, Invoices, DeliveryNotification. Each of these kinds of documents are used for a well defined purpose, and could sound to be generic, but they also must carry specific information for the sector and reflect the needs of the firms.

After the ontology definition, the document definition is performed on a semantic basis : in order to define a document template to adopt within a business process, a Document Manager identifies and selects the information that the document should carry. This operation is performed exploiting the ontology definition and browsing the ontology itself with a software application.

Practically, the document building is not any more performed with the syntactical definition of the document templates, but with the semantic identification of the information treated in the document.

Once the target information has been identified into the ontology by a human personnel, a software tool can automatically compose the related datatypes defined in the library to build the final document template. In this way the XML Schema code definition is completely entrusted to an automatic application. Clearly, the resulting XML Schema template contains all the references supported in the ontology.

The definition of a new document type requires the definition of a root for the new document and a new XML Schema complex type for the root. In this sense the definition of a new document corresponds to the definition of a new complex type; in other words, a document is basically a special "*complex*" datatype, and it is therefore inserted into the Datatype Library. Once the document has been defined it is ready to be used within a business process. The set of complex types for these roots represents the implemented documents in the framework.

The implementation on the *semantic document builder* could be done as a plugin of an ontology editor, for example a plugin for Protègè. In this way it could be possible to navigate the ontology, and select those part of the ontology that should be described in the document.

Element customization

Standardisation of documents, processes, and protocols is fundamental for interoperability, but specific business requirements and characteristics of the target sectors and of the enterprises can require to customize the business documents also in a vertical perspective. The semantic approach can ease the customization process for the enterprises. Starting from a "standard" document, a user can modify the structure of the document upon a semantic basis, adding or deleting information.

In case of deletion it is possible to generate in an automatic manner a set of instructions (expressed with a XSLT template) to transform a standard document into a customized one (see picture 8.11). On the other hand, in case a user adds some information to the content of the customized document, the ontology will provide a semantic modellisation of the added information to allow the understanding of the lacking information and to support mapping mechanisms from the different formats.

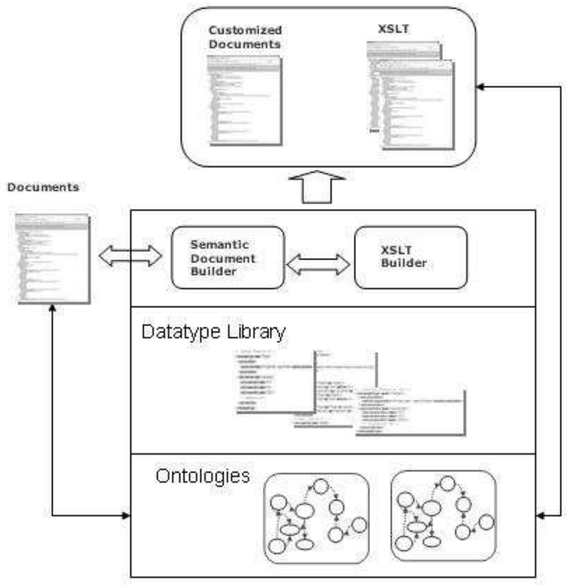


Figure 8.11: Document customization

The information about the generated XSLT templates and the relative mapping mechanisms to transform the document format each other are maintained into the *mapping* and *context* section of the System Ontology. In this way, once generated, mapping mechanisms will be available for data exchange.

8.3.6 The architecture

Fig. 8.12 depicts the overall structure of the architecture prefigured to design, implement and improve a B2B protocol.

The image highlights first of all that there are two basic layers: the semantic layer and the syntactic layer. These layers are directly related with those presented in fig. 8.2 .

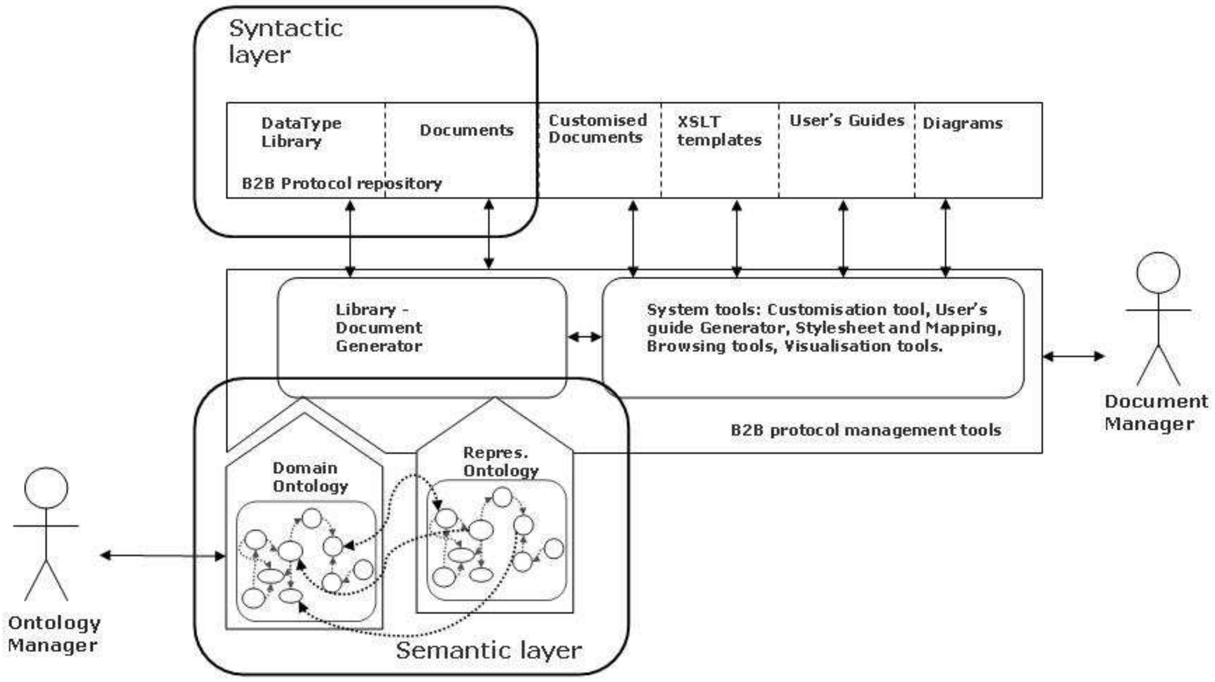


Figure 8.12: The overall architecture

The framework aims to represent an improved ”standard factory”, where the domain ontology represents the variable parameter starting from which it is possible to generate each other component. In respect with the experience matured within the development of the Moda-ML framework, and the structure of the document factory architecture depicted in fig. 7.4, the proposed architecture prefigures the improvement of the standard design and implementation with the adoption of the semantic web technologies to implement the vocabulary, that is the core of the architecture. The vocabulary definition is no more performed using a database, but with the more powerful mechanisms of OWL ontologies (and following a specific simple model - represented by the KDMO).

Following the idea of the document factory, the architecture still prefigures the development of a set of tools around the vocabulary, but using the ontologies these tools will result more powerful and useful for standard development. The main software tools comprised in the framework are:

- A datatype library generator.
- A semantic document builder.

Other tools form the improved ”document factory”: these tools allow

- to customise the documents.
- to define XSLT style sheets for presentation purpose of the documents.
- to automatically generate User Guides for the documentation of the framework.
- to browse and to provide a graphical visualisation of both the ontology and the business documents.

For example, a specific navigation tool should provide a mechanism to browse into the developed specification. In particular the navigation tool provides different perspectives to consult the specification and connect them: the semantic view, that exploits the browsing of the ontology is connected with the formal view of the datatype (the definition of the XML Schema)and with a textual description, provided by the user's guide; browsing the repository it is possible to switch from one perspective to another one.

All the resulting documents generated by the tools that form the specification of the defined B2B protocol are then collected in a repository that is accessible to the users via web.

As said before, the proposed framework aims basically to target three results, strictly related with the world of standards:

1. to ease the definition and maintenance of a B2B protocol for a specific business domain.
2. to ease the understanding and the dissemination of the defined B2B protocol.
3. to ease the set up of integration mechanisms that can allow the enterprises to adopt and to exploit the standard for actual commercial transactions.

Especially the third point is fundamental, where both the development and the adoption of business standards suffer of many limitations. The integration mechanisms should be used not only to extract information for different heterogeneous data sources, but they should interface the back-end application systems of the enterprises with different data models in order to efficiently import and export data. Nowadays many research initiatives are going on to exploit semantic description of data models to achieve system interoperability.

In the context of data integration among enterprises, the defined domain ontology is exploited

- to represent a common vision about the domain knowledge. This common vision (for example, adopting the MOMIS-SEWASIE architecture [5] [4]) can be used to perform queries on the different data sources regardless the specific data structure adopted by each sources.
- to map each single data source, described using the extraction of semantic information of the data source, with the common ontology that represents the standard. As in the model proposed by Harmonise [82], the mapping is performed resolving the clashes between the involved ontology. The mapping stage involves the writing of a set of rules that allow to associate elements of the data source with elements of the domain ontology. These mapping rules are then be exploited for data transformation between the local resources and the ontology.

8.3.7 Life-cycle of the framework

The definition of an ontology-based architecture for the development of B2B protocols and standards impacts in the life-cycle of standard definition. This vision reinforces the separation between the modelling phase and the implementation phase of B2B protocols and business documents: the ”new” life-cycle presents different complementary perspectives during the B2B protocol development.

In particular, considering the life-cycle presented in section 7.9, and in fig. 7.11 the adoption of the ontology regards:

- phase 3, design. The design phase is performed using ontologies to model knowledge domain, and it is strictly correlated with the model proposed in the KDMO. Clearly, this phase is crucial for the development of the following components of the B2B protocol. This task is entrusted to a team of domain and technology experts that should co-operate in order to well-define the Domain Ontology. In this phase no more the documents and data formats are designed, instead the focus is on the semantic aspect of the information that should be exchanged. The target in this phase is the modellisation of the basic knowledge that underlies the future definition of the protocol.
- phase 5, implementation. The implementation phase is strongly distinct from the design phase, and the implementation of fundamental components - like documents and data types - that are part of the B2B protocol are built exploiting semi-automatic mechanisms. The business document building can be performed on a semantic base, exploiting the connection between the Domain Ontology and the Datatype Library.
- phase 8, dissemination, adoption and improvements. Standard dissemination and adoption is a crucial step for the interconnection of different systems. Domain Ontologies represent

the basis to better perform all the stages prefigured in this phase (see section 7.9, phase 8). In particular, data integration among the enterprise systems and the proposed B2B protocol, and dissemination of the results could strongly benefit from the modellisation in ontologies of the knowledge domain. Moreover, in this phase simple demo software could exploit the definition of Domain Ontology: for example ontology browsing could be used to ease the spreading and the comprehensibility of the defined business documents.

- phase 9, growth and extension phase. The evolution of the framework is associated with the evolution of the domain ontology: in fact the ontology is not only the starting point for the definition of the B2B protocol and for its dissemination, but it is also a model that can highlight limits of the developed protocol and can be compared with existing solutions and requirements to put in evidence further functionalities that must be supported. In this sense, the improvement of the protocol passes by the extension of the related domain ontology.

Chapter 9

Conclusions

This work is focused on the development of interoperability mechanisms to improve business collaboration among industries, enterprises and firms. The contribution of the thesis is basically composed of two parts: the analysis, study and the implementation of a framework and a set of software tools for the Textile-Clothing (T/C) sector, and the definition of an ontology based approach to improve the definition and adoption of both B2B protocol and standards.

Starting from the analysis of the requirements of the T/C business sector, the wide plethora of diverse standards that have been defined up to now, I contribute in the Moda-ML project for the definition of a common platform that can be easily adopted by enterprises to improve interoperability among themselves. The platform has adopted the ebXML standard specification and in particular has resulted in the definition of a vocabulary of ebusiness terms, compliant with the ISO 11179 specification, specific for the target sector. The implementation of the components of the framework has been centered on the vocabulary that represents the core of the whole architecture. Recognizing the relevance of the world of the standards to support real business collaborations, the activities of the Moda-ML project has been joined with two standardisation initiatives (Texspin and TexWeave) to encourage the birth of an european standard.

On the other hand, standard definition and adoption could be really difficult and cumbersome to achieve. While the enterprises really need a common language to model and formalize business information and a set of shared protocols to exchange such information, they also present many difficulties in their adoption. The study of the peculiarities of the different business sectors shows us that these difficulties can vary a lot from one business scenario to another one. In particular the nature of the supply chain (if it is characterised by the presence of big industries or by a lot of Small and Medium Enterprises, the heterogeneity of the business partners and the length of the supply chain itself) and the nature of the standard itself (if it is more bounded with technological aspects or it is more related with specific business characteristics) can strongly impacts both the development and the adoption of the protocol. Considering the experience maturated into the development of the Moda-ML framework for the textile-clothing industries, and the emerging ICT technologies, especially in the scenario of the web and the Semantic Web, a relevant part of the thesis concerns the proposal for a novel approach for the development of B2B protocol, exploiting the adoption of the emerging semantic web technologies; the ontologies can represent the core of an interoperability architecture, representing an ”improved” ebusiness vocabulary. The aim is to enhance the efficiency of the standard life-cycle impacting on three important activities: protocol development, dissemination and adoption.

In order to overcome not only the issues related with the definition of B2B protocols and standards but also those problems (perhaps more complex) related with their dissemination and adoption, the thesis proposes a ontology-based interoperability framework where I consider the semantic definition of the standards and of the protocols as the first step for their development. The background of this proposal has been both the study of the methodologies, the characteristics and the procedures that are adopted by other relevant standards, and the existing initiatives that exploit semantic web technologies to improve system integration. In this perspective an important consideration is that ontologies are fundamentally used for describe meta data or for mapping purposes, but are not used as modelling tools up to which to generate and to connect data formats.

The approach starts from an abstract definition of interoperability, that includes three main components:

- The vocabulary component, that comprises the definition of the terms used to exchange information, and should specify as good as possible the semantic of the information.
- The document component, that comprises the document structures used to exchange information between partners.
- The process component, that outlines the business logic and the collaboration scenario of the production sector, modelling the exchange activities that drive the enterprise collaborations.

In order to manage these components, the proposed approach consists of two fundamental part:

- the definition of a model for the developing of Domain Ontologies. These ontologies will represent the basis for the development of B2B protocols for specific business domain and for the birth of the related vertical standard (that is also semantically described).
- the outline of a software architecture that can exploit the semantic definition and the domain ontologies to build practically B2B protocols and to foster their adoption.

In the phase of business standard definition, the semantic view allows to better involve domain experts and to ease the modallisation of the knowledge domain. Moreover the use of Semantic Web technologies, and in particular of the OWL language, allows to easily interconnect the data format, expressed using XML Schema, with the semantic representation of the documents. As shown in the proposed example, in this way B2B protocol definition consists only in the modellisation of the knowledge domain for the specific business sector. Although this is not necessary an easy task, this approach raises the developers from the difficulties related with the technical definition of all those aspects related with data format(and for example with the widespreaded validation languages).

The dissemination of the B2B protocol (and the related standard) benefits from the definition of the ontology, because semantic description can exploit user-friendly interface to simplify the comprehension of the protocol: in order to understand the content and the functionalities of a specific data format it will not more be required technical competencies. In this way it will be more easy also the customization of the released specification to face enterprises peculiarities.

The interest on the ontology development justifies the adoption of ontology to model knowledge domain, to build data formats and to introduce ontology-based modellisation in the standardisation processes. The new semantic web technologies - based on the use of ontologies - for system integration can be successfully exploited for the adoption of business standards allowing the enterprises both to improve their business processes (using shared formats for digital document exchange) and to save money in the configuration of their information systems.

More in detail, considering the requirements outlined in chapter 6, the proposed approach for the development of B2B protocols positively impacts on the following points:

- Usability. The adoption of ontologies to model domain knowledge related with B2B protocols will increase the easiness using the protocol itself: both the installation and the integration with the back-end application systems could exploit ontology-based tools to reconfigurate the importing and exporting of data between different systems. From a human perspective, each document, message and component of the procotol, being connected with a semantic definition, will be more understandable. The content of a business message will be easy to browse exploiting the connection with the related ontology. This can also ease the customization of the standard and the business message. In this case specific tools to manage document customization exploing semantic definitions should be provided.

- Scalability. The use of a widespread standard in general eases the scalability of a specific framework. With this preamble, a common formal description of a B2B protocol increases the usability of the framework (see previous point), and then improves scalability. So a more easy and usable standard means also a greater scalability.

- Looseely coupling among the partners. The easeness in the definition of importing/exporting mechanisms towards back-end application systems results also in a more loosed coupling among the partners. In fact, in this case complex and ad-hoc integration applications are no more needed for business data exchange.

- Easy to interface with other interoperability framework. In many case, for a specific business sector many different standards and interoperability frameworks are available. The interoperability among these different frameworks and standards are surely improved if each of them will expose a semantic definition of its data formats and documents. In this way, the effort for an enterprise to adopt or to integrate different views is reduced.

Security aspects are not related with the stage of B2B protocol definition, but are mainly charged to the applications used to exchange data. In this sense they are not interesting in this context.

Particular attention must be paid to the aspects of maintenance and upgrade. In fact, from a certain perspective, the semantic definition of the protocol surely eases the maintenace of the protocol itself. On the other hand, the upgrade of a semantic model could hide several difficults and traps. This is strictly related with a limit of this approach: in fact up to now there is no a clear, unique guideline or methodology, like in database systems, for the definition of semantic models. The ontology definition process could result to be an hard and complex effort.

References

[1] E. Agirre, O. Ansa, E.Hovy, D.Martinez. Enriching very large ontologies using www. *In Proceedings of the Ontology Learning Workshop ECAI*, Berlin, Allemagne (2000).

[2] D.L. Anderson, F.F. Britt, D.J. Favre. *The Seven Principles of Supply Chain Management*, from the Spring 1997 issue of Supply Chain Management Review.

[3] N. Anicic, N. Ivezic, A. Jones. An Architecture for Semantic Enterprise Application Integration Standards. *In proceedings of INTEROP-ESA 05, First conference of Interoperability of Enterprise Software and Applications*, February 23-25, 2005, Geneva, Switzerland .

[4] D. Beneventano, S. Bergamaschi. The MOMIS Methodology for Integrating Heterogeneous Data Sources. *IFIP World Computer Congress*, Toulouse France, 22-27 August 2004.

[5] D. Beneventano, S. Bergamaschi, F. Guerra, M. Vincini. Building an integrated Ontology within SEWASIE system. In *Proceedings of the First International Workshop on Semantic Web and Databases (SWDB)*, Co-located with VLDB 2003 Berlin, Germany, September 7-8, 2003.

[6] J. Bermudez. *Supply Chain Management: More Than Just Technology*, from the March/April 2002 issue of Supply Chain Management Review.

[7] Business Process Management Initiative (BPMI). http://www.bpmi.org/.

[8] Business Process Modeling Language (BPML). http://www.bpmi.org/BPML.htm.

[9] M. Bichler, A. Segev, J. Zhao. *Component-based E-Commerce: Assessment of Current Practices and Future Directions*, SIGMOD Record Vol. 27, No. 4, 1998, 7-14.

[10] C. Bussler. *The Role of SemanticWeb Technology in Enterprise Application Integration*, Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2003.

[11] C. Bussler. *B2B Integration - Concepts and architecture*, Springer - Verlag ISBN 3-540-43487-9.

[12] F. Cargill. Consortia and the evolution of Information technology Standardization. *Proceedings of the 1st Conference on Standardisation and Innovation in Information technology (SIIT 1999)*, Aachen, Germany, September 15-17, 1999, pp.37-42.

[13] J. L. Cavinato. *What's Your Supply Chain Type?*. Supply Chain Management Review - May 1, 2002.

[14] P.G. Censoni, P. De Sabbata, G. Cucchiara, F. Vitali, L. Mainetti, T.Imolesi. MODA-ML, a vertical framework for the Textile-Clothing sector based on XML and SOAP. In *Challenges and achievements in e e-business and e-work*, Prague 15-18 October 2002, ISBN IOS Press 58603 284 4/ISBN Ohmsha 4 274 90541 1 C3055.

[15] European Committee for Standardization - Information Society Standardization System. http://www.cenorm.be/isss.

[16] European Committee for Standardization (CEN). http://www.cenorm.be/.

[17] O. Corcho, A. Gomez-Perez. *Evaluating Knowledge Representation and Reasoning Capabilities of Ontology Specification Language*. In Proceedings of ECAI-00 Workshop on Applications of Ontologies and Problem-Solving Methods (Berlin, Germany, 2000).

[18] Z. Cui, D. Jones, P. O'Brien. *Semantic B2B Integration: issues in Ontology-based Approaches*, SIGMOD Record 31 (1) Vol. 31, Number 1, March 2002: 43-48.

[19] J. Chen, P. Kacandes, D. Manning, B. Meltzer, T. Rhodes. eCo Architecture for Electronic Commerce Interoperability, CommerceNet, 1999, http://www.commerce.net/docs/ecoframework.pdf .

[20] E. Cimpian, A. Mocan. D13.7 v0.2 Process Mediation in WSMX, July 2005.

[21] CommerceNet. http://www.commerce.net/.

[22] R. Costello, Mitre Corporation and members of the xml-dev list group. XML Schema: Best Practices, 2001, www.xfront.com/BestPracticesHomepage.html.

[23] OASIS ebXML Collaboration Protocol Profile and Agreement Technical Committee. Collaboration-Protocol Profile and Agreement Specification Version 2.0. OASIS 2002. http://www.oasis-open.org/committees/ebxml-cppa/documents/ebcpp-2.0.pdf

[24] F. Curbera et al.. *Unraveling the Web Services: An Introduction to SOAP, WSDL, and UDDI*. IEEE Internet Computing, vol. 6, no. 2, Mar./Apr. 2002, pp. 86-93.

[25] CWA 14948/2003. Guidelines for XML/EDI messages in the Textile/clothing sector. CEN/ISSS Workshop Agreement, CEN/ISSS, Bruxelles, 2003.

[26] Commerce XML (cXML). http://www.cxml.org/.

[27] S. Damodaran. B2B Integration over the Internet with XML - RosettaNet Successes and Challenges. *WWW2004*, May 17-22, 2004, New york, New York, USA.

[28] DAML-S Coalition. DAML-S: Web Services Description for the Semantic Web. In I. Horrocks and J. Hendler, editors, *Proc. First International Semantic Web Conference ISWC 2002*, LNCS 2342, pages 279–291. Springer-Verlag, 2002.

[29] A. Dogac, I. Cingil. A survey and comparison of business-to-business e-commerce frameworks. *ACM SIGecom Exchanges*, Vol. 2, Issue 2 Spring, pp. 16 - 27, 2001

[30] P. De Sabbata, N. Gessa, G. Cucchiara, T. Imolesi, F. Vitali. Supporting eBusiness with a dictionary designed in a vertical standardisation perspective. In *Proceedings of IEEE-CEC Conference*, Munich, Germany, July 19-22 2005, pp. 160-167, ISBN 0-7695-2277-7.

[31] P. De Sabbata, N. Gessa, C. Novelli, A.Frascella, F. Vitali. B2B: Standardisation or Customistation?. In *"Innovation and the Knowledge Economy Issues, Application, Case Studies", e-Challenges 2005 conference*, Ljubljiana, October 19-21 2005, edited by Paul Cunningham and Miriam Cunningham, pp 1556-1566, IIMC International Information Management Corporation LTD, Dublin, Ireland, IOS PRESS, ISBN 1-58603-563-0.

[32] U. Dayal, M. Hsu, R. Ladin. Business Process Coordination: State of Art, Trends, and Open Issues. *Proceedings of the 27th VLDB Conference*, Morgan Kaufmann Publishers Inc., Roma, Italy, September, 11-14, 2001, pp 3-13.

[33] Dean, M. ed. 2004. OWL-S: Semantic Markup for Web Services. http://www.daml.org/services/owl-s/1.0/owl-s.pdf.

[34] Dublin Core Metadata Glossary. http://library.csun.edu/mwoodley/dublincoreglossary.html.

[35] electronic Business in the Textile Clothing and Footwear industries. e-Business Watch, sector report n. 01-II, European Commission Enterprise Pubblications, Brussels, August 2004.

[36] ebusinessWatch. http://www.ebusiness-watch.org/resources/charttool.htm.

[37] European Commission. The European e-Business Report - A portrait of e-business in 10 sectors of the EU economy, 11-2005, http://www.ebusiness-watch.org/resources/documents/eBusiness-Report-2005.pdf, ISBN 92-894-5117-3.

[38] Electronic Business using eXtensible Markup Language (ebXML). http://www.ebXML.org/.

[39] electronic Business in the Textile Clothing and Footwear industries. e-Business Watch, sector report n. 01-II, European Commission Enterprise Pubblications, Brussels, August 2004.

[40] T. M. Egyedi, A. Dahanayake. Difficulties implementing standards. In *3rd Conference on Standardisation and Innovation in Information technology (SIIT 2003)*, Delft, the Netherlands, October 22-24 2003, pp.75-84.

[41] Electronic Data Interchange For Administration, Commerce and Transport (EDIFACT). http://www.unece.org/cefact/.

[42] EDITEX. TEDIS project, Trade EDI Systems Programme. Interim report, 1992, Office for official pubblications of the European Community, ISBN-92-826-5658-6.

[43] European e-business Showcase. European Communities, 2003. ISBN 92-894-5057-6.

[44] Fensel, Horrocks et al.. OIL in a nutshell, *In Proc. Of EKAW-2000*, LNAI, 2000.

[45] Foundation for Intelligent Physical Agents (FIPA) http://www.fipa.org/.

[46] N. Gessa, M. Busanelli, P. De Sabbata, F. Vitali. Extracting a semantic view from an ebusiness vocabulary. In *proceedings of IEEE International Conference on E-Commerce Technology*, San Francisco, California, June 26-29, 2006, pp 398-401, ISBN 0-7695-2511-3.

[47] N. Gessa, G. Cucchiara, P. De Sabbata, A. Brutti. A bottom-up approach to build a B2B sectorial standard: the case of Moda-ML/TexSpin. In *Proceedings of INTEROP ESA 05-NOREST Workshop*, Geneve February 22nd 2005, publication by Hermès Science Publications.

[48] N. Gessa, De Sabbata, M. Fraulini, T. Imolesi, L. Mainetti, M. Marzocchi, F. Vitali. MODA-ML, an interoperability framework for the textile-clothing sector. Proceedings of *IADIS Conference*, 5-8/11/2003, Carvoeiro, Portugal.

[49] N. Gessa, P. De Sabbata, M. Fraulini, T. Imolesi, L. Mainetti, M. Marzocchi e F. Vitali. Moda-ML, an experience of promotion of a sectorial interoperability framework. In *Building Knowledge Economy: Issues, Applications, Case Studies*, Bologna October 23-24 2003, pag 350-357, ISBN IOS Press 1 58603 379 4 ISBN Ohmsha 2 274 90623 X C3055.

[50] N. Gessa, C. Novelli, M. Busuoli, F. Vitali. Use and extension of ebXML business profiles for Textile/Clothing firms. In Proceedings of *E-Commerce and Web Technologies*, the 5th international conference EC-Web 2004, Zaragoza, Spain, September 2004, LNCS 3182 Springer, pp. 186-195

[51] J. Hendler, *Agents and the Semantic Web*. IEEE Intelligent Systems. March/April 2001 (Vol. 16, 2).

[52] Hollingsworh. The Workflow Reference Model: 10 Years On, http://www.wfmc.org/standards/docs/Ref_Model_10_years_on_Hollingsworth.pdf.

[53] Ian Horrock. DAML+OIL: a description logic for the semantic web. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2001.

[54] W. Hasselbring, H. Weigand. Languages for electronic business communication: state of art. Industrial Management & Data Systems 101/5, 2001 pp. 217-226.

[55] Institute of Electrical and Electronics Engineers. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York, NY: 1990. http://www.sei.cmu.edu/str/indexes/glossary/interoperability.html.

[56] Internet Engineering Task Force (IETF). http://www.ietf.org/.

[57] International Organization for Standardization (ISO). http://www.iso.org/.

[58] ISO/IEC 11179-3, Information Technology – Metadata Registries (MDR), Geneve, February 2003, http://metadata-standards.org/11179/.

[59] ISO 15000-5. ebXML Core Components Technical Specification.

[60] EDIFACT Application level syntax rules - ISO 9735 - 1987.

[61] K. Jakobs. Standardisation and SME Users Mutually Exclusive?. *In Proc. Multi-Conference on Business Information Systems*, Cuvillier Verlag, 2004.

[62] D. Jones, T. Bench-Capon, P. Visser. Methodologies for Ontology Development. In *IT & KNOWS - Informations Technology And Knowledge Systems*, IFIP World Computer Congress in Vienna and Budapest, 31 August - 4 September, 1998, José Cuena.

[63] J. Euzenat. Research Challenges and Perspectives of the Semantic Web. Report of the EU-NSF strategic workshop held at Sophia-Antipolis, France, October 3rd-5th, 2001.

[64] S. J. Kahl, *What's the "Value" of Supply Chain Software?*. Supply Chain Management Review - January 1, 1999.

[65] G. Larsen. Component-based enterprise frameworks. *Communications of the ACM*, 43(10):24–26, October 2000.

[66] H. L. Lee, *Creating Value Through Supply Chain Integration*. From the September/October 2000 issue of Supply Chain Management Review.

[67] C. B. Lee. Demand Chain Optimization: Pitfalls and Key Principles. Evant Inc., January 2003.

[68] P. Lehti, P. Fankhauser. XML Data Integration with OWL: Experiences & Challenges. In *Proceedings of SAINT 2004*: 160-170.

[69] L. Li, I. Horroks. A Software Framework For Matchmaking Based on Semantic Web Technology. *In Proceedings of WWW2003*, May 20-24, 2003, Budapest, Hungary. ACM 1-58113-680-3/03/0005.

[70] C. Legner, K. Wende. Towards an Excellence Framework for Business Interoperability. In *Proceedings of 19th Bled eConference eValues*, Bled, Slovenia, June 5 - 7, 2006.

[71] M. Li, M. W. Küster, B. Gatti. CEN/ISSS report and recommendations on key eBusiness standards issues 2003-2005. CEN, Bruxelles, 2003.

[72] Leymann, Roller, Schmidt. *Web services and business process management*. IBM Systems Journal, Vol. 42, n° 2, July 2002.

[73] J. Lubell. XML representation of Process Description. May 2002, http://ats.nist.gov/psl/xml/process-descriptions.html.

[74] H. L. Lee, S. Whang. E-Business and Supply Chain Integration. November 2001, http://www.stanford.edu/group/scforum/Welcome/EB_SCI.pdf.

[75] B. Medjahed, B. Benatallah, A. Bouguettaya, A. H. H. Ngu, and A. Elmagarmid. *Business-to-Business Interactions: Issues and Enabling Technologies*. The VLDB Journal, Springer, Vol. 12 (1), 2003.

[76] D. McGuinness. Ontologies for Electronic Commerce. *AAAI '99 Artificial Intelligence for Electronic Commerce Workshop*, Orlando, Florida, July, 1999.

[77] B. Meadows. Universal Business language (UBL) 1.0. cd-UBL-1.0, OASIS, September 15 2004, docs.oasis-open.org/ubl/cd-UBL-1.0/

[78] Moda-ML Public Final Report. doc MG11-029, Bologna, september 2003, www.moda-ml.org/moda-ml/download/mg11-029-moda-ml_final_report.doc.

[79] S. A. McIlraith, D. Mandell. Comparison of DAML-S and BPEL4WS. 2002, URL:http://www.ksl.stanford.edu/projects/DAML/ Webser-vices/DAMLS-BPEL.html

[80] A. Maedche, S. Staab. Semi-automatic engineering of ontologies from texts. In *Proceedings of the 12th International Conference on Software Engineering and Knowledge Engineering(SEKE2000)*, Chicago, IL, USA, pages 231–239, July 2000.

[81] OASIS, ebXML Message Service Specifications. http://www.oasis-open.org/committees/ebxml-msg/documents/ebMS_v2_0.pdf.

[82] M. Missikoff, F. Taglino. *An Ontology-based Platform for Semantic Interoperability*, Handbook on Ontologies, 2004, p. 617-634.

[83] Stephen J. New. *The scope of supply chain management research*, Supply Chain Management Review ,May 1, Vol. 2, Number 1, 1997, pp. 15-22.

[84] I. Niles, A. Pease. Towards a Standard Upper Ontology. *2nd International Conference on Formal Ontology in Information Systems (FOIS)*, Chris Welty and Barry Smith, eds, Ogunquit, Maine, October 17-19, 2001.

[85] I. Niles, A. Terry. The MILO: A General-purpose, Mid-level Ontology. In *Proceedings of the International Conference on Information and Knowledge Engineering, IKE'04*, June, 2004, Las Vegas, Nevada, USA.

[86] N. Noy, M. Musen. An Algorithm for Merging and Aligning Ontologies: Automation and Tool Support. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI99)* , Workshop on Ontology Management, Orlando, FL, 1999.

[87] Organization for the Advancement of Structured Information Standards (OASIS) http://www.oasis-open.org/.

[88] B. Omelayenko, D. Fensel. *Scalable document integration for B2B electronic commerce*, Special Issue of Electronic Commerce Research Journal on B2B Research, 2001.

[89] Object Management Group (OMG). http://www.omg.org/.

[90] Ontoviz. 5-2006, http://protege.cim3.net/cgi-bin/wiki.pl?OntoViz.

[91] OWL-S. http://www.daml.org/services/owl-s/1.0/.

[92] C. Peltz. *Web Service Orchestration and Choreography*, Web Services Journal, July, 2003.

[93] J. Park, S. Ram. *Information Systems Interoperability, What Lies Beneath?*. ACM Transaction on Information Systems, vol. 22, No. 4, October 2004, pp. 595-632.

[94] Protègè. 8-2006, http://protege.stanford.edu/.

[95] J. Pyke, R. Whitehead. Does better Math Lead to Better Business Processes?. November 2003, http://www.wfmc.org/standards/docs/better_maths_better_processes.pdf.

[96] J. Qin, S. Paling. *Converting a controlled vocabulary into an ontology: the case of GEM*. In Information Research, Vol. 6 No. 2, January 2001. http://informationr.net/ir/6-2/paper94.html (11/2005).

[97] N. Radjou. *Deconstruction of the Supply Chain*. Supply Chain Management Review - November 1, 2000.

[98] J. M. Reeve. *The Financial Advantages of the Lean Supply Chain*. From the March/April 2002 issue of Supply Chain Management Review.

[99] L. Rosenfeld, P. Morville. *Information Architecture for the World Wide Web.* 1998, Sebastapol, CA: O'Reilly Media. (ISBN 1-56592-282-4).

[100] Rosetta Net. http://www.rosettanet.org/.

[101] Supply Chain Management Center. http://www.cio.com/research/scm/edit/012202_scm.html.

[102] Z. Shan, D.K.W. Chi, Q. Li. Systematic Interaction Management in a Workflow View Based Business-to-business Process Engine. In *Proceedings of the 38th Hawaii International Conference on System Sciences*, Hawaii, USA, 2005.

[103] Smith, Fingar. Workflow is just a pi-process. BPTrends, November 2003.

[104] M.H. Sherif. *When is standardisation slow?*. International Journal of IT Standards & Standardistion Research, vol. 1, N. 1, Jan.-June 2003.

[105] E. Soderstrom. Formulating a General Standards Life Cycle. In *Proceedings of 16th International Conference of Advanced Information Systems Engineering - CaiSE 2004*, LNCS 3084 Springer, Riga, Latvia, June 2004, pp 263-275.

[106] S. Shim, V. Pendyala, M. Sundaram, J. Gao. *Business-to-business e-commerce frameworks*. Computer, Vol.33 N.10, IEEE Computer Society, 2000, pp. 40-47.

[107] Definitions of standard on the Web: http://www.google.it/search?q=define:standard& hl=it&lr=&c2coff=1&defl=en.

[108] Standard list. http://www.oasis-open.org/cover/sgml-xml.html.

[109] STandard for the Exchange of Product (STEP). http://www.tc184-sc4.org/SC4_Open/SC4 Legacy Products (2001-08)/STEP_(10303)/.

[110] THE ISO Standards Glossary. http://www.standardsglossary.com/.

[111] H. Sneed, C. Verhoef. Reengineering the Corporation - A Manifesto for IT Evolution. http://citeseer.nj.nec.com/446988.html.

[112] Society for Worldwide Interbank Financial Telecommunication (SWIFT). http://www.swift.com/.

[113] T. Berners-Lee, J. Hendler, O. Lassila. *The semantic web*. Scientific America, May 17, 2001.

[114] Trastour, Bartolini, Preist. Semantic Web Support for the Business-to-Business E-Commerce Lifecycle. *WWW2002*, May 7-11, 2002, Honolulu, Hawaii, USA.ACM 1-58113-449-5/02/0005.

[115] TEXSPIN - e-business for the textile/clothing sector. http://www.cenorm.be/cenorm/businessdomains/businessdomains/isss/cwa/textilecwa.asp.

[116] P. Timmers. *Business Models for Electronic Markets*. Journal on Electronic Markets, 8 (2), 1998, pp. 3-8.

[117] B. Meadows. Universal Business language (UBL) 1.0. cd-UBL-1.0, OASIS, September 15 2004, docs.oasis-open.org/ubl/cd-UBL-1.0/.

[118] Universal Description, Discovery and Integration (UDDI). http://www.uddi.org.

[119] What is UDDI. http://www.webopedia.com/TERM/U/UDDI.html.

[120] Unified Enterprise Modelling Language (UEML project). http://www.ueml.org.

[121] United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT). http://www.unece.org/cefact/.

[122] K. Vassilopoulou, A. Pouloudi, A Patronidou, A. Poulymenakou. E-business models, a proposed framework. In *Proceedings of the e-Business and e-Work Annual Conference*, Prague, Czech Republic 16-18 October (2002), pp 1003-1009, 2002.

[123] World Wide Web Consortium (W3C). http://www.w3.org/.

[124] Workflow Management Concilium (WfMC). http://www.wfmc.org.

[125] E-business, Wikipedia. http://en.wikipedia.org/wiki/E-business.

[126] Web Service Architecture Requirements (WSAR). http://www.w3.org/TR/2004/NOTE-wsa-reqs-20040211/.

[127] Web Services Business Process Execution Language (WS-BPEL). http://www.oasis-open.org/committees/download.php/10347/wsbpel-specification-draft-120204.htm.

[128] Web Services Choreography Description Language (WSC-DL). http://www.w3.org/TR/ws-cdl-10/.

[129] XML Common Business Library (xCBL). http://www.xcbl.org/.

[130] Extensible Markup Language (XML). http://www.w3.org/XML/.

[131] XML for Electronic Data Interchange (XMLEDI). http://www.eccnet.com/xmledi/guidelines-styled.xml.

[132] J. Yang, M. Papazoglou. *Interoperation support for electronic Business*. Communication of the ACM, Vol. 43, No. 6, June 2000, pp 39-47.

[133] J.A. Zackman. Concepts of the Framework for Enterprise Architecture. Los Angels, CA; Zackam International, 1996.

[134] J. Leon Zhao, H. Lu, V. B. Misic. Applications and Architectures of Electronic Commerce Management Systems(ECMS): A Domain Oriented Approach. http://citeseer.ist.psu.edu/407711.html.

[135] Zhao, Sandahl. Potential advantages of semantic web for internet commerce. In *Proceedings of International Conference on Enterprise Information Systems (ICEIS)*, Vol 4, pp151-158, Angers, France, April 23-26, 2003.