Interoperability of technical enterprise applications

Nicolas Figay

N° d'ordre Année 2009

THESE DE L'UNIVERSITE DE LYON

Délivrée par

L'UNIVERSITE CLAUDE BERNARD LYON 1

ECOLE DOCTORALE

DIPLOME DE DOCTORAT

(arrêté du 7 août 2006)

soutenue publiquement le 27 Novembre 2009

par

Mr. FIGAY Nicolas

TITRE : Interopérabilité des applications d'entreprises dans le domaine technique
"Interoperability of technical enterprise applications"

Directeur de thèse : Mme GHODOUS Parisa

JURY:

- Mr. Aris. M. OUKSEL, Professeur des Universités à l'université de l'Illinois à Chicago, USA
- Mr. AIT-AMEUR Yamine, Professeur des Universités, Ecole Nationale Supérieure de Mécanique et d'Aérotechnique, France
- Mr. FENG Shaw, chercheur au NIST, USA
- Mr. JARDIM-GONCALVEZ Ricardo, chercheur senior de la nouvelle université de Lisbonne, directeur de UNINOVA-GRIS, Portugal
- Mr. GARDAN Yvon, professeur des universités à l'université de Reims-Champagne-Ardennes, groupe MONGE, France
- Mr. MONDON Jean-Yves, responsable du projet d'harmonisation PLM du Groupe EADS (Phenix), EADS, France
- Mme. GHODOUS Parisa, directrice de thèse, Professeur des universités, LIRIS, Université Lyon 1, France

Acknowledgments

I would like to thank especially Parisa Ghodous, Professor at the University Lyon 1, who has kindly mentor my thesis. She had the difficult task of guiding a researcher from the industrial world, an expert in his field, with all the bad habits to correct when achieving a more academic work, with its conventions and codes.

I also thank Mr. Aris M.OUKSEL and Mr. Amine AIT-AMEUR for the honor to be the reporters for my thesis, and for their comments and advice on this document.

I also warmly thank Mr. Shaw FENG, Mr. Ricardo JARDIM-GONCALVEZ, Mr. Yvon GARDAN and Mr. Jean-Yves MONDON for agreeing to be the examinators of this thesis.

I would also like to thank Mr. Eric DUCEAU and Me. Martine CALLOT, my managers at the research centre of EADS, who helped me in my activities and my mission to work towards the achievement of this thesis.

I will not forget to thank my colleagues from EADS and the different experts and researchers I met during various international or French projects, both research and industrial, for discussions and the contribution to the ideas that follows. In particular, thanks to Mr. Tony BAGNALL who took the time to read the manuscript and provided me valuable feedback. Thanks also to Mr. Michel DUREIGNE. Our numerous exchanges contributed to enrich the thinking and the conceptual approach. Many thanks to Mr. Jean-Yves DELAUNAY, who oriented many of the research projects and studies I contributed to for industrial usage.

Finally I would like to thank my family members, who have given me the strength and the energy to undertake this thesis. Thanks to their patience and their support, I have been able to handle this work.

To Françoise, Romain and Charlotte

Table of content

Table of Figures

List of tables

Abbreviation and acronyms

AFNOR	Assocation Française de NORmalisation
AIF	ATHENA Interoperability Framework
AP	Application Protocol (STEP)
AP203	Application Protocol 203 (“Configuration controlled 3D designs of Mechanical parts and assemblies”)
AP214	Application Protocol 214 (“Core data for automotive mechanical design processes”)
AP233	Application Protocol 233 (“Systems Engineering data representation”)
AP239	Application Protocol 239 (“Product life cycle Support”)
ArchiMate	An integrated architectural approach for the description and visualization of different business domains – a project and a language standardized by Open Group
ATHENA	Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Applications (FP6-IST 507849)
ATL	Action Transformation Language
ASD Stan	Aerospace and defense industries association of Europe
B2B	Business to Business
BLOB	Binary Large Object
BMM	Business Motivation Model
BPEL	Business Process Execution Language
BPM	Business Process Modeling
BPMM	Business Process Maturity Model
BPMN	Business Process Modeling Notation
CAD	Computer Aided Design
CAE	Computer Aided Engineering
CAM	Computer Aided Manufacturing
CAx	Computer Aided application x
CCM	CORBA Component Model
CHiSEL	Computer Human Interaction & Software Engineering Lab
CIM	Computer Independent Model (MDA, OMG)
CORBA	Common Object Request Broker Architecture
COTS	Commercial Off-The-Shelves
CPU	Computer Processing Units
CRAN	Research Centre for Automatic Control
DBE	Digital Business Ecosystem
DBMS	Data Base Management System
DEX	ata EXchange set (PLCS)
DI	Diagram Interchange (OMG, MDA)
DL	Descriptive Logic
DMU	Digital Mock-Up
DOD	Department Of Defense
DRM	Digital Rights Management
DTD	Document Type Definition
EADS	European Aerospace Defense and Space Company
EAI	Enterprise Application Integration

ebXML	electronic Business XML
EJB	Enterprise Java Bean
EPISTLE	European Process Industries STEP Technical Liaison Executive
EU	European Union
FTP	File Transfer Protocol
FUNSIEC	Feasibility study for a UNified Semantic Infrastructure in the European Construction
GIFAS	Groupement des Industries Françaises Aéronautiques et Spatiales
GME	Generic Modeling Environment
HTTP	Hyper Text Protocol
ICT	Information and Communication Technologies
IDEAS	InteroperabilityDevelopmentfor Enterprise Application and Software – Roadmaps (IST-2001-37368)
IDEF	Integration Definition for Function Modeling
IDL	Interface Definition Language (CORBA, OMG)
IEEE	Institute of electrical and electronics engineers
IETF	Internet Engineering Task Force
IMM	Interoperability Maturity Model (NETHA)
IMS LAPS	Intégration du Matériau au Système -Automatique, Productique et Signal (laboratoires)
INCOSE	International Council on Systems Engineering
INRIA	Institut National de Recherche en Informatique et Automatique
INTEROP	Interoperability Research for Networked Enterprise Applications and Software
IOR	Interoperable Object Reference
ISO	International Organization for Standardization
ISO TC184 SC4	ISO Technical Committee 184 (Automation systems and integration) Sub Committee 4 (Industrial data)
J2EE	Java for Enterprise Edition
JDBC	Java DataBase Connectivity
JVM	Java Virtual Machine
LCI	Layer of Coalition Interoperability
LCIM	Levels of Conceptual Interoperability
LIRIS	Laboratoire d'InfoRmatique en Image et Systèmes d'information
LISI	Levels of Information System Interoperability
MANTIS	MANufacturing Technology and Industrial Systems task force
MDA	Model Driven Architecture (OMG)
MDD	Model Driven Development
MDE	Model Driven Engineering
MEL	Manufacturing Engineering Laboratory
MMEI	Maturity Model for Enterprise Interoperability
MOF	Meta Object Facilities (OMG, MDA)
NATO	North Atlantic Treaty Organization
NC3TA	NATO C3 Technical Architecture
NCPD	Networked Collaborative Product Development
NETHA	National E-Health Transition Authority
NIST	National Institute of Standards and Technology
OASIS	Organization for the Advancement of Structured Information Standards
OASIS PLCS	OASIS Product Life Cycle Support Technical Committee

TC	
ODBC	Open DataBase Connectivity
ODP	Open Distributed Processing
OEM	Original Equipment Manufacturer
OIM	Organizational Level of Interoperability
OMG	Object Management Group
OpenDevFactory	Sub project of SYSTEM@TIC PARIS REGION French project called "Usine Logicielle"
OMA	Object Management Architecture*
OpenGL	Open Graphic Libraries
ORB	Object Request Broker (CORBA)
OWL	Web Ontology Language
PDM	Product Data Management
PDMS	Product Data Management System
PERL	Practical Extraction and Report Language
PHENIX	PLM Harmonization for ENhanced Integration & eXchange
PHP	PHP:Hypertext Preprocessor
PIM	Platform Independent Model
PLCS	Product Lifecycle Customer Support
PLM	Product Lifecycle Management
POEM	Process Oriented Enterprise Modeling
POSC CAESAR	Petrotechnical Open Software Corporation Caesar (project)
PSL	Process Specification Language (ISO 18629)
PSM	Platform Specific Model (MDA, OMG)
RDB	Relational Database
RDF	Resource Description Framework (W3C)
RDFS	Resource Description Framework Schema (W3C)
RDL	Reference Data Libraries
RFC	Request For Comments (IETF)
RISESTEP	enteRprise wIde Standard accEss to STEP distributed databases (Esprit project 20459)
RMI/IIOP	Remote Method Invocation over Internet Inter ORB protocol
RM-ODP	Reference Model - Open Distributed Processing
SADT	Structured Analysis and Design Technic
SAVE	Step in A Virtual Enterprise (Bright Euram project 97-5073)
SBVBR	Semantics of Business Vocabulary and Business Rules (OMG)
SDAI	Standardized Data Access Interface (ISO STEP)
SE	System Engineering
SEINE	Standards pour l'Entreprise Innovante Numerique Etendue
SINTEF	The Foundation for Scientific and Industrial Research
SLM	Simulation Lifecycle Management
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SOMF	Service Oriented Modeling Framework
SOSI	System of systems Interoperability
SparQL	Simple Protocol and RDF Query Language
SPEM	Software Process Engineering Meta model
SQL	Structured Query Language
S-TEN	Intelligent Self-describing Technical and Environmental Networks (FP6-

	IST 2005 project 027683)
STEP	Standard for the Exchange of Product model data
SysML	System Modeling Language
TOGAF	The Open Group Architecture Framework
U3D	Universal 3 Dimensions
UBL	Universal Business Language
UEML	Unified Enterprise Modeling Language
UML	Unified Modeling Language
UNINOVA	Institute for the Development of New Tecnologies
UP	Unified Process
URI	Universal Resource Identifier
URL	Universal Resource Locator
URN	Universal Resource Name
UUID	Universal Unique Identifier
VDA	Verband Der Automobilindustrie
VIVACE	Value Improvement through a Virtual Aeronautical Collaborative Enterprise (AIP-CT-2003-502917)
VRML	Virtual Reality Markup Language (W3C)
W3C	World Wide Web Consortium
Wfmc	Workflow Management Coalition
WSDL	WEB Service Definition Language
XMI	ML Model Interchange
XML	Extensible Markup Language
XSD	XML Schema Definition

Genesis 11:1-9[1]

"And the whole earth was of one language, and of one speech. And it came to pass, as they journeyed from the east, that they found a plain in the land of Shinar; and they dwelt there. And they said one to another, Go to, let us make brick, and burn them thoroughly. And they had brick for stone, and slime had they for mortar. And they said, Go to, let us build us a city and a tower, whose top may reach unto heaven; and let us make us a name, lest we be scattered abroad upon the face of the whole earth. And the Lord came down to see the city and the tower, which the children built. And the Lord said, Behold, the people is one, and they have all one language; and this they begin to do; and now nothing will be restrained from them, which they have imagined to do. Go to, let us go down, and there confound their language, that they may not understand one another's speech. So the Lord scattered them abroad from thence upon the face of all the earth: and they left off to build the city. Therefore is the name of it called Babel; because the Lord did there confound the language of all the earth: and from thence did the Lord scatter them abroad upon the face of all the earth."

Figure 1: Engraving The Confusion of Tongues by Gustave Doré (1865).

[1] King James Version

General Introduction

The objective of the research undertaken through this thesis is to propose an innovative approach to obtain interoperability of enterprise technical applications supporting collaborative processes within extended and virtual enterprises.

Based on the federation of existing mature open standards, information and communication technologies and engineering disciplines, this approach aims enabling efficient and fast digital based collaboration between enterprises which belong to a given community. It means that the right product information, being documents or computable models, should be available for the right actor, in the right place, at the right time and in the right format, despite heterogeneity of the involved applications and associated operational environments.

Continuous interoperability should be achieve without undue efforts on the part of humans and involved stakeholders, including after system upgrades.

1 - Thesis context

The thesis manuscript formalizes the research work carried out during the thesis in the last three years at LIRIS, as a student at the University Lyon 1. It also aims to synthesize and reflect results of Aerospace industry research projects I contributed to since 1997 within the Aerospatiale joint research centre, now EADS (European Aerospace Defense and Space) innovation works, within a department dealing with engineering systems and industrial sharing.

The aim of my work in the research center is to help reduce cycles and development costs of an industrial product in the Aerospace & Defense domain, by the means of Information and Communication technologies.

The central theme of my work is the exchange, sharing and preservation of data describing an industrial product, for which definitions are exchanged and shared between different organizations and different disciplines, using various software tools.

The numerous projects and studies where I was involved led me to ask the question of interoperability of enterprise applications based on the use of software products, supporting inter-disciplines collaborations within networked organization, and involving the exchange and sharing of digital informational and computational resources. European research project RISESTEP (Esprit project 20459) aimed to demonstrate that it was possible to share elements of digital mock-up units distributed within heterogeneous systems, on the basis of standards. SAVE (STEP in A Virtual Enterprise, Bright Euram project 97-5073) project aimed to support collaboration through Internet between several industrial partners involved in a given industrial project. They were followed by European research roadmap for interoperability of enterprise applications within virtual enterprise, IDEAS (Interoperability Development for Enterprise Application and Software –Roadmaps, IST-2001-37368), which led participating to ATHENA program (Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Applications, FP6-IST 507849) for the IDEAS roadmap implementation. OpenDevFactory French research project (SYSTEM@TIC PARIS REGION – Usine Logicielle) was the opportunity to assess and experiment deeply the maturity of

model driven approach and associated modeling platforms based on open source. Results were exploited within EADS Phenix (PLM harmonization) industrial project and BoostAerospace industrial project.

Figure 2 summarizes the research and standardization context of my thesis. Research projects and industrial projects, which I contributed to, are positioned on the timescale. The Product Lifecycle Management (PLM) standards are also positioned on the timescale. They are also positioned according the underlying standardized information and communication technologies they are based on and specified with[2]: STEP technology for data exchange (STEP), STEP technology for data sharing (SDAI standing for STEP Data Access Interface), Common Object Request Broker Architecture (CORBA) and Object Management Architecture (OMA), CORBA Interface Definition Language (IDL), Simple Object Access Protocol (SOAP), W3C's web services and Service Oriented Architecture (SOA), Model Driven Architecture (MDA) and semantic WEB technologies (Web Ontology Language) for Reference Data Libraries (RDL).

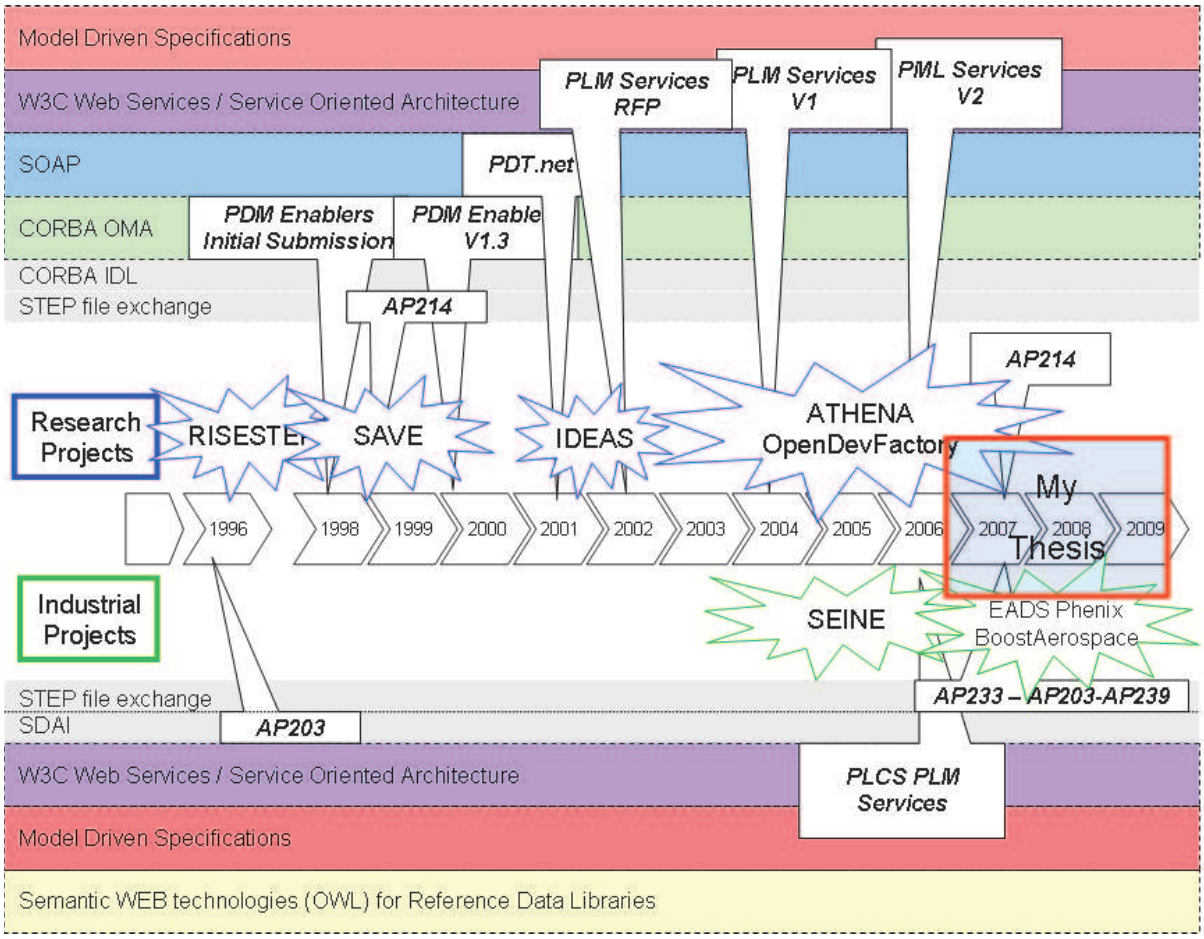


Figure 2: Thesis context map

Two families of Product Lifecycle Management (PLM) exist, which were used and studied in order to support Industrial collaboration. They are produced by two different communities: European automotive industry and Aerospace & Defense industry involved with NATO. European automotive standards are placed on top of the timescale while Aerospace & Defense standards are placed below the timescale. It is important to point out that European Aerospace

[2] Most of these standards will be explained further within Part 1 of the thesis manuscript, when relevant. In addition, links are provided on specifications at the end of the thesis manuscripts.

was not a key player for PLM standards and had to assess the different available standards according their needs. In addition, evolution of standards and of used technologies was constrained by the environment and progress of PLM communities and Information and Communication Technologies (ICT) community. Figure 2 illustrates fast and continuous evolution of Information and Communication Technologies used by PLM standards. STEP applications protocols (AP203, AP214, AP239[3]) provides business semantic, while technologies are continuously evolving, covering information exchange (STEP technologies – with STEP standing for Standard for the Exchange of Product model data - , Web technologies – with SOA standing for Simple Object Access Protocol - and Semantic Web technologies – with OWL standing for Web Ontology Language and RDL standing for Reference Data Libraries) and sharing (STEP technologies – with SDAI standing for Standardized Data Access Interface - , Object Management Group middleware technologies – with IDL standing for Interface Definition Language - and WEB service oriented technologies).

Both automotive community and aerospace community are developing holistic approaches leading to develop their own sets of standards for product data exchange and sharing, considering not only information (Application Protocol 203 for aerospace, Application Protocol 214 for automotive), services (PLCS PLM services for aerospace, OMG PDM enablers, PLM services and PDT.net for automotive) and processes (Data Exchange sets-DEXes - for aerospace, VDA[4] processes for automotive), but also model driven approach and BPM (Business Process Modeling).

In order to understand the problem addressed in the thesis, it is important to understand how these different projects, their objectives and their results contributed to the way I formalize the problem addressed by the thesis and to the proposed innovative approach.

Sharing Digital Mock-up units between heterogeneous environments: RISESTEP

I started in 1997 as leader of European research project RISESTEP, which stands for "Enterprise wide standard access to STEP distributed databases" (Esprit project 20459). RISESTEP aimed to demonstrate that it was possible to share elements of digital mock-up units distributed within heterogeneous systems, on the basis of standards. Firstly, the ISO 10303 standard (Standard for the Exchange of Product model data or STEP) Part 203 (Configuration Controlled 3D Design application protocol or AP203) was used for the definition of the semantics of the data product. Secondly, the standard CORBA (Common Object Request Broker) was used for communication between heterogeneous systems, encompassing the machine types, operating systems and development languages.
The feasibility of sharing digital mock-up unit on heterogeneous was demonstrated, through the development of a high level interface using the Interface Definition Language (IDL) of CORBA and AP203. This demonstration shows that such a standardized higher level interface was needed. The standardization was realized within the SAVE project.

[3] AP203 stands for Application Protocol 203 ("Configuration controlled 3D design"), AP214 stands for Application Protocol 214 ("Core data for automotive mechanical design processes") and AP239 stands for Application Protocol 239 ("Product life cycle Support")
[4] VDA stands for Verband Der Automabilindustrie

Data exchange and sharing within the Virtual Enterprise: SAVE

RISESTEP highlighted need of a standardized higher level interface for interconnection of PDM systems. Standardization of such an interface was done in 1999 through the European Project SAVE (STEP in A Virtual Enterprise, Bright Euram project 97-5073).
SAVE project aimed to support collaboration on Internet between several industrial partners involved in a given industrial project (Original Equipment Manufacturers[5], Equipment providers, risk sharing partners, sub-contractors). Contractual aspects and categorization of partners was an important aspect for accurate coverage of business scenarios.
Used application protocol for product data was not AP203 ("Configuration Controlled 3D Design") but a similar one, AP214 ("Core Data for Automotive Mechanical Design" application protocol). The reason was identification of better accuracy of AP214 for the needs of Aeronautic partner of the projects, in order to support aircraft customization for different airline companies. Interface specifications were submitted to the Object Management Group, the "PDM enabler". "PDM" stands for Product Data Management. It was partially derived from AP214.
I contributed to this project, in particular by providing business cases and through evaluation of the proposed innovation. Some demonstrations of partial implementations of these services were performed, with an insufficient functional coverage when willing to cover a complete industrial case. I identified here some issues related to resource consumption when willing to deal with interoperability, implying commercial software product of the shelves. At the end of the project, we had only a demonstration, not an operational implemented solution. Following the OMG's process, it is expected from solution providers delivering a specification, to produce a commercial product implementing the specification the following year. Consequently, we had to wait for availability of the commercial implementation to validate maturity and robustness of "PDM enablers" specifications and implementations.

Post analysis of RISESTEP and SAVE

In order to evaluate maturity of "PDM Enablers" specifications and available implementations, I then launched and realized several internal studies the year following the end of SAVE project.
The first study was evaluation of implementation of PDM Enablers within a software product (PDM System), provided one year latter by a partner of the SAVE project.
The study demonstrated that the quality of the implementation was not sufficient for industrial usage. It also pointed out that usage of PDM enablers and STEP interface for a same information exchange scenario, were leading to incoherent results. In fact, these two kinds of interfaces are addressing two different viewpoints, informational and engineering. It is absolutely required to ensure coherency between them. But the coherency between these two viewpoints was ensured neither by internal architecture of software product nor by the specifications themselves.
In parallel, I asked some students who were working for me to develop a PDM server from scratch, based on OMG specifications, in order to evaluate completeness of the PDM Enablers specifications if they were willing to define precisely and formally a Product Data Management System. It was a failure because of complex interdependencies between PDM Enablers Modules and lower CORBA services defined within the set of Common Services (c.f. 4.6 - Object Management Group Framework). Because of such interdependencies, the important level of expertise is mandatory, concerning CORBA and Object Management

[5] An **original equipment manufacturer** (OEM) is typically a company that uses a component made by a second company in its own product, or sells the product of the second company under its own brand.

Architecture. This second study explained encountered difficulties for the software product provider when willing to provide industrial implementation of the specification. Firstly the inheritance mechanism brings up over-complexity when the vendor is willing to provide compliant implementations, in particular if wrapping existing software. Secondly, no coherent approach was proposed by standardization projects to insure coherency between the viewpoints addressed by STEP application protocol, the Informational viewpoint, and by PDM Enablers, the Engineering viewpoint.
The conclusion was that interoperability can't be insured considering only a standardized formal business information model and a standardized business interface which is domain specific, developed through isolated and independent projects. It is the reason why I decided to investigate state of the practice related to enterprise application integration and eBusiness collaboration within running projects of my enterprise.

Further investigation for Enterprise Application Integration and eBusiness collaboration

In order to check if encountered issues where they have not already resolved in practice and within operational context, I then analyzed research projects results and eBusiness applications integration best practices within the Aerospatiale Group. This work pointed out high costs in implementing interfaces based on standardized neutral information model if considering only two applications. Considering a larger number of interconnected applications is required in order to make savings[6]. Consequently interconnection and integration of several enterprise applications cannot be though out of the frame of a global integration strategy aiming to support business processes. In addition, as interoperability is the quality expected for operational application systems exchanging and sharing informational resources all along the lifecycle of the industrial product (lifetime is about 50 years for an aircraft product), it must be insured through a support structure (continuity on time) and not through a project (limited on time). My conclusion was that interoperability of technical application allowing management, sharing and exchange of « product » data must be addressed at enterprise level, with strategic choices on standards to use and to develop in order to be able to insure it.

I then launched and realized a new study about tools allowing governance of enterprise information systems, in relation with enterprise strategy, in order to identify how choices of standards and their usage can be formalized through maps and business rules of emerging approach called "controlled urbanism of the enterprise information system"[7]. This study allowed, through formalization of maps, to illustrate how important a governance structure is at the scale of the enterprise, when willing to address interoperability of enterprise applications supporting industrial programs going beyond the frontier of the enterprise, implying numerous companies working on the same product. When willing efficient exchange and sharing of digital information describing a product and managed by local enterprise applications, it is mandatory to consider federation of enterprise applications and to consider urbanism within the context of eBusiness collaboration, with associated frames and infrastructures.

In parallel, I also launched and realized a study on the semantic web and associated ontology language, in order to compare advantages and drawbacks of using such or such formalism and paradigm for description of the Product data. The study also aimed to assess usage of

[6] It is commonly considered that important savings is possible with more of five interconnected systems.
[7] It is also called "city planning".

emerging sets of standards, defined by several standardization bodies, and using different formalisms for different purpose. Examples are schema definition language for product data exchange, Unified Modeling Language for design modeling and Web Ontology Language for the semantic web.
Efforts for the definition of business use cases allowed to point out importance of involvement of all the concerned stakeholders for concerned given disciplines, without being limited only by enterprise or project boundaries. It also identified importance of being able to change formalism without loosing semantic due to translation. Finally, the study allowed pointing out that interoperability has to be managed using a framework based on business concepts, shared and agreed (consensus) by a community of interest, which are software solution, enterprise, project and formalism independent.

At last, I launched and realized study in order to address impact of infrastructure constraints on interoperability. I firstly considered usage of interconnected computers belonging to enterprise internal networks (intranet) or enterprise external networks (extranet). I then considered usage of distributed, integrated or federated software systems, in particular within the area of product data management, and considering standards associated to Product data exchange and sharing, coming from Object Management Group (community working on interoperable software components market), World Wide Web Consortium (community working on Web standards), OASIS (community working on eBusiness development) and ISO TC184 SC4 (community working on product data exchange and sharing). It highlighted existence of numerous, overlapping and heterogeneous standards allowing to address interoperability at information and communication technologies levels, with technological silos due to usage of different incompatible paradigms, formalization languages and conceptual frameworks. It was the reason motivating launching new European research projects in order to work on interoperability of enterprise applications within networked organizations.

Interoperability of Enterprise applications: IDEAS and ATHENA

The results of RISESTEP project, SAVE project and following studies pushed me to associate the EADS research Centre to the launching of European research roadmap for interoperability of enterprise applications, IDEAS (Interoperability Development for Enterprise Application and Software –Roadmaps), in order to allow the community to address the identified issues when willing to take the full benefits of a approach based on usage of PLM standards. If feasibility was demonstrated, ensuring interoperability within industrial context was still too much difficult and costly.

Research programs such as ATHENA (Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Applications) and research network such as INTEROP (Interoperability Research for Networked Enterprise Applications and Software) were launched in order to implement this roadmap.

Due to limited resources, I was only able to contribute to ATHENA. My contribution was related to establishment of dynamic requirement engineering method for interoperability (as scientific leader), to elaboration of business scenarios and pilots, to contribution of the ATHENA interoperability framework (AIF) [Berre et. Al. 2006] and to the integration of results through establishment of a Collaborative Product Development platform for networked enterprises.

This platform was based on concepts and approaches developed by ATHENA partners: joint usage of enterprise modeling, model driven engineering allowing generation of business components to be deployed on service oriented execution platforms, semantic mediation allowing resolution of semantic conflicts using ontology, and collaborative process modeling as aggregation of public views of private executable processes.
Integration was particularly difficult, as project delivered solutions were only demonstrators, not prototypes. I had to develop some missing components, in particular in order to reuse legacy application protocols coming from STEP as ontological models to describe the Product.
I adopted an iterative approach for integration of components, with a strategy based on identification of alternative solutions based on open source and open standards, in order to cover full expected functional coverage of the demonstrations, but also to assess and measure real innovation comparing ATHENA solutions to the state of the practice.

Finally, I had to formalize a coherent and innovative approach and to apply it for setting up of a collaborative platform dedicated to Product development by a networked organization, integrating results of previous research activities and completing it in order to address identified interoperability issues.

My thesis was initiated during ATHENA, aiming to formalize the adopted approach and to improve ATHENA principles, by considering numerous issues which were not considered by the ATHENA partners. In particular, manufacturing standards were not considered, and being able to reuse STEP application protocols was an absolute prerequisite within a model driven context, in order taking advantage of service oriented execution platforms and ontology. I consequently initiated the thesis by publishing a first paper describing how to enable interoperability of STEP application protocols at meta-data and knowledge level (R. Jardim-Goncalvez and N.Figay, 2007), and producing a first demonstrator called the STEP Mapper.

Model Driven Approach assessment and extended hypermodel initialization

During my thesis, I also participated to the project OpenDevFactory, in order to apply principles developed during my thesis to Simulation Lifecycle Management, in particular for Eclipse based design platforms coupled with Simulation Lifecycle Management and Product Lifecycle Management tools. OpenDevFactory was the opportunity to assess and to experiment deeply maturity of model driven approach and associated modeling platform based on open source. It was also the opportunity to point out required alignment of graphical representation dedicated to human (diagrams) and formal representation dedicated to automates (models), leading to develop the concept of “extended hypermodel”. Finally, it allows identifying issues related to semantic preservation when trying to use together models formalized using different languages and based on different paradigms.

Product lifecycle Management, System Engineering, Standardization and training

Working on the thesis, I was in parallel involved, as an expert on Product Lifecycle Management (PLM) standards, in industrial projects. These projects aimed to harmonize PLM approaches within EADS group (Phenix, which stands for PLM Harmonization for ENhanced Integration & eXchange), and to address existing digital break within the extended enterprise when working on an aerospace and defense program (SEINE, or “Standards pour l’Entreprise Innovante Numérique Etendue”).

I was also acting on standardization as an expert at AFNOR and ASD STAN (Aerospace and defense industries association of Europe). It was an opportunity for applying and presenting some of the principles developed during my thesis and to evaluate results within an industrial context.
These activities allowed me identifying some important issues related to interoperability, such as maturity level of involved communities, governance or information interchange and sharing between internal private management systems of different organization implied within a collaboration process.

2– The addressed problem

All my past activities allowed me identifying importance but also limitations of the current legacy interoperability standards and associated frameworks. If not associated an appropriate way, they are not ensuring operational interoperability of technical enterprise applications within virtual enterprise. Heterogeneous relevant legacy solutions, being standards or frameworks, are to be federated, aggregated, combined, configured, completed and improved. The aim is to address interoperability a holistic way, implying relevant disciplines and stakeholders, preserving previous investments and according accurate processes and methods.

The concerned disciplines are those associated to domains involved in the lifecycle of enterprise applications and their infrastructures, such as knowledge representation by ontology, software engineering, enterprise modeling and finally product data exchange, sharing and long term preservation.

As most of the standards and frameworks for interoperability are partial solutions and were not designed in order to be combined, their federation is quite challenging and required new innovative approach allowing to produce accurate frameworks for enterprise technical applications interoperability within the virtual enterprise.

Within the globalization context, such approach should support emerging communities and ecosystems to share common objectives, infrastructures, issues, principles and processes in order to efficiently govern the set of relevant interoperability standards and coordinate the different actors and stakeholders to involve in order reaching continuous interoperability.

Today no holistic approach exist in order to establish interoperability frameworks for collaboration space dedicated to a given eBusiness multi-disciplinary community, federating efficiently heterogeneous legacy solutions, emerging paradigms and emerging technologies.

3 – Personal contribution

My contribution consists in proposing a holistic approach aiming to establish interoperability framework for collaboration space dedicated to a given eBusiness multi-disciplinary community, federating efficiently heterogeneous legacy solutions and emerging paradigms and technologies.
I firstly characterized the industrial trends, emerging interoperability needs for networked organizations and drawbacks of legacy solutions and approaches. This is not only based on analysis of the state of the art, but also on practical experimentations and lessons learnt from industrial projects I contributed to or I analyzed during the fifteen last years.
Using models of reference allowing describing interoperability levels of the different systems implied, I characterized the ideal collaborative system which will support enterprise

application interoperability within networked organizations. It is expected to achieve seamless exchange of information between distributed information systems and collaborative organizations, by mean of semantic integration, loosely coupled technologies and asynchronous communications. In order to achieve adaptive and continued interoperability, interoperability should be programmatic, constructive and operational, involving evolutionary decision, information and physical systems. Finally matured selected architectures of reference and domain standards should be used and governed.
Having clearly established industrial context and characterized expected interoperability between the different systems constituting a collaboration space, I iteratively assessed several combinations of open standards and technologies, and continuously adapted and extended the approach instrumented and promoted by research community within ATHENA: model driven engineering allowing to project business logic on an execution service oriented platform, taking advantage of more advanced features coming from enterprise modeling, model driven software engineering, ontology and service oriented technologies.
The originality of the approach is to be eBusiness community centric, and to consider that several kinds of standards and associated frameworks are to be aggregated a posteriori, considering them as components on the shelves used in order to build a federated composite framework dedicated to support growing interoperability maturity of the considered community.
Using such an approach within eBusiness context, important issues exist, in particular semantic preservation and data lost between the different artifacts of a model driven approach and between the different applicative components which have to be interconnected in order to support digital collaboration within the virtual enterprise.
It is the reason why I develop a new concept, the extended hypermodels for interoperability, closely integrated in the approach, as a response to the identified issues.

In order to apply and validate the proposed approach, I produced numerous pilot demonstrators, components, language mappings and technology assessments through different research programs and industrial projects. They were produced iteratively, most of the time focusing on some aspects of the proposed approach. I also iteratively defined, used and validated a set of principles for interoperability and a set of interoperability issues and brakes. In parallel, I performed assessment of technologies and assessment of available free and open tools on the web implementing standards, in order to contribute to specification and architecture of enterprise applications enabling interoperability. I finally used the capture "know how" and experience in order to contribute to standardization policies within EADS, Interoperability framework for governance of standards with International standardization Group for the Aerospace community and for contribution to standardization communities.
All these experiences were the opportunity to test and to validate the proposed approaches and principles according several viewpoints: communities, enterprises, system engineering process, application users, application support, software product providers, etc.

The formalization of extended hypermodel concepts was coming late within the process, as it was resulting from analysis and assessment of the proposed approach, leading to a very late proof of concept. However it brings a lot of indirect results. In particular, it can be considered as a tool allowing to analyze current standards drawbacks when dealing with interoperability, and to be able to produce recommendations for their improvement and their evolution in order to face new emerging requirements for better interoperability.

It was not possible to describe all the realizations performed during the three years of the thesis, and consequently it was required to make a choice in order to focus on some particular

points. I fully designed and implemented the Networked Collaborative Development Platform pilot demonstrator during the last year of ATHENA. It constituted my first iteration in order to define the proposed approach, with a special focus on targeted execution platform based on Service Oriented execution platforms. OpenDevFactory provides the opportunity to evaluate emerging OMG standards and their implementation on top of Eclipse, as a suitable open environment allowing to design applications and to generate applications. SEINE allowed assessment of the approach developed in ATHENA, within PLM Aerospace community, and demonstrated important gain in cost and time to produce and validate requirements related to usage of standards for data exchange between numerous partners through a hub. Finally extended hypermodel concept can be applied not only to software engineering, but also to multi-disciplinary design of an industrial product. The concept will be applied in conjunction with some assessed technologies for preliminary design phase of a product implying simulation capabilities.

4 - Organization of the thesis manuscript

This thesis manuscript is composed of twelve chapters, distributed between three parts.

Part 1-State of the art on Technical Enterprise Application Interoperability

Chapter 1 provides general context and defines technical enterprise application interoperability.

Chapter 2 presents models of reference allowing characterizing interoperability

Chapter 3 presents the different standards aiming to address interoperability

Chapter 4 presents the different interoperability frameworks

Chapter 5 presents a set of formalized interoperability issues and brakes which are not currently solved looking at the state of the art, and which are to resolve if willing to obtain in operation interoperability of technical enterprise applications

Part2 – Establishment of pragmatic framework for interoperability of enterprise application

Chapter 6 describes a set of identified, classified, assessed and validated principles of construction of an interoperability framework for the federation of enterprise applications within the virtual enterprise, which are allowing to solve individually or through combination most of the identified interoperability issues.

Chapter 7 defines the concept of extended hyper model, aiming to solve issues not solved within the state of the art, in particular semantic preservation and semantic mismatch between artifacts and applicative components concerned by the proposed approach.

Chapter 8 presents a concrete example with set of relevant standards and frameworks identified within the first part, with identification of impact on governance of standards.

Chapter 9 formalizes a simple generic process with can be used as a template for establishing a methodology shared by an eBusiness community governing his standards for evolutionary and continuous interoperability.

Part3 – Concrete experimentations and implementations

Chapter 10 described the demonstration platform that I developed within the ATHENA, related to establishment of a Networked Collaborative Product Development platforms for the PLM domain. Focus is on standards based infrastructure.

Chapter 11 presents demonstrator developed within OpenDevfactory, with focus on Model Driven Engineering and assessment of Eclipse as MDA ground.

Chapter 12 presents the demonstration environment developed for the project SEINE, in order to assess the use of STEP application protocols for exchange of data between specific private PDM applications and a generic server based on the neutral application protocol 214 (meta - PDM data). It illustrates and validates the proposed approach within an industrial environment.

Conclusion

The conclusion firstly summarizes the innovative approach defined within the thesis. It secondly provides a discussion about the encountered difficulties and the choices I made to build the thesis manuscript. It thirdly explains how to measure the added value and usability of the proposed approach, in particular on a comparison with the results of the research projects I was involved in before the thesis.
It finally presents the numerous perspectives for potential exploitation of the results, theoretical extensions of the proposed approach and finally applications of the developed concepts to other field and domain of research.

Part I – State of the art on Technical Enterprise Application Interoperability

In this part, I introduce the need for innovative approach and principles in order to define a framework allowing achieving pragmatic interoperability of enterprise application within networked organizations.

I firstly propose a set of definitions stating clearly what interoperability of enterprise applications is. Then I propose a panel of relevant models of reference, standards and frameworks addressing interoperability.

Then I provide a state of the art on global holistic approaches addressing interoperability through the usage of legacy standards and associated technologies.

I finally characterize identified emerging needs for innovative solutions in order to cover interoperability needs, in particular through the definition of a set of interoperability brakes and issues which are to be solved.

Chapter 1 – General context for technical enterprise applications interoperability

1.1 - Defining interoperability

Very numerous definitions of interoperability are provided by the literature and by the different standardization organizations. (Ford et Al. 2008) points out that, according their survey, thirty-four definitions of interoperability where created since 1977 which are used within the literature, providing history of creation and number of time it is used, in particular within Department Of Defense documents. One of the more usually used is the 1977 definition with is repeated under convenience and often rephrased:

> INTEROPERABILITY: The ability of systems, units, or forces to provide services to and accept services from other systems, units, or forces and to use the services so exchanged to enable them to operate effectively together.

According to performed analysis, this definition is probably often used due to the fact it is generic enough to fit with different kinds of interoperability.
Within the information system field, several suitable definitions are proposed:

> INTEROPERABILITY:
> (a) Ability of information systems to communicate with each other and exchange information.
> (b) Conditions, achieved in varying levels, when information systems and/or their components can exchange information directly and satisfactorily among them.
> (c) The ability to operate software and exchange information in an heterogeneous network (i.e., one large network made up of several different local area networks).
> (d) Systems or programs capable of exchanging information and operating together effectively.

According to (Ford et All. 2008), sixty-four types of interoperability are mentioned in research papers, demonstrating richness of the interoperability field.
Is it needed to agree on a single and precise one? As stated by (Morris et Al. 2004), "We may never have any agreement on a precise definition due to differing expectations that are constantly changing. New capabilities and functions … continue to offer new opportunities for interactions between systems."

In order to provide precise definition, it is consequently important to make a point about the different expectations and considered interacting systems. The expectations, as interacting organizational systems, are defined by the industrial context and emerging interoperability needs. The interacting systems and concerned capabilities are defined through available models of reference for interoperability and the concerned standards and frameworks.

1.2 - Industrial context and new interoperability needs

Within industry, current trends and organizations are reinforcing needs for extensive collaboration between enterprises and communities, requiring interoperability of enterprise applications supporting the business processes**.**

In the following, I explained current industrial practices and trends which impact the needs in term of interoperability: the concurrent engineering, the virtualization of the product, the virtualization of the enterprise, the product lifecycle management, the system engineering, the nomadic and virtual working environments and finally the systematic usage of Commercial product Off-The-Shelves (COTS). The combination of these trends and practices creates new interoperability needs and challenges.

1.2.1 - Concurrent engineering and virtualization of the Product

Competition today is leading enterprises to short the time-to-market. It is particularly important to gain parts of a continuously changing and evolving market, but also for shareholders for who shorter industrial projects should allow better visibility and faster benefits.

As production is optimized since several years, the phase where important reduction can be expected is the design phase. The idea was to develop new ways of working, based on parallelization of engineering tasks, also called Concurrent Engineering, and to provide computer aided design tools to the engineers for the multiple implied disciplines.

Design offices are consequently organizing their work through production of multidisciplinary electronic models, which will be the inputs of next phases (production planning, test elaboration, production, tests, support, exploitation…). **Authoring tools** are used for aiding engineering, usually defined as Computer Aided Design software, while **Product Data and Process management tools** support collaboration and product data sharing and the information flow between different concerned actors, taking the advantage of emerging Information and Communication Technologies.

During the last ten years, Paper based models (static 2D) have been replaced by electronic dynamic models, which will be used for certification of future aerospace products.
Due to the "digital" nature of these models, that can be consider as engineering artifacts but also as code to be processed by a machine, numerous challenges exist for product data exchange, sharing, retention and trust.

In addition, the tooling for such models is highly complex, and it is no more possible for enterprises to create their own in house software applications as software engineering is out of their core business activities. Such trends consequently imply usage of software products, creating a software editors dependency.

1.2.2 - Virtualization of the enterprise

Competition today is also leading enterprises to be focused **on core business and high value activities.** Enterprises have to identify sub-contractors, risk sharing partners or finally equipment suppliers (e.g. engine, landing gear…) that are the best in class for complementary

activities or required components. For such group of enterprises, only one enterprise is the interface with the clients, who can have the impression that they are working with only one enterprise: the virtual enterprise.

The underlying model is also leading to outsourcing in order to reduce cost of the product. When applied on software development, it pushes **usage of Commercial Off-The-Shelves (COTS)** that is preferred compare to in-house software development.

It is important to point out that several drawbacks exist for virtualization of the enterprise. Pushed to the extreme, it leads the enterprises to be only empty shells as important part of the knowledge and know-how is out of the enterprise. Knowledge management and high level of expertise for collaborative methods is consequently more and more important. It also creates **complex interdependency** between actors and information systems. Today, the consequence of the globalization, with in particular heterogeneity of manpower cost, impact of the price of the dollar or contracts implying to share the work with other countries, is the **evolution of Partnership/Subcontracting networks**: sub-contracted activities will increase while the number of sub-contractors of range one will decrease. The way to share the knowledge should evolve to support these new organizations and constraints, leading to complex digital business ecosystems[8].

Within the virtual enterprise, there is a looser coupling than within the integrated and extended enterprise that implies process, methods and software products heterogeneity. Well managed collaboration and federation of enterprise applications are of a particular importance in such a context.

1.2.3 - Product life cycle management

In order to have more competitive products, other approaches are focusing on reducing number of changes when developing a product and to have the changes at the earlier stage of product development: the latter a change is required, the more expensive it will be.

The other idea is to have **early involvement of downstream activities**, in order to design right at the first time, and to consequently reduce number of potential future change requests in manufacturing, exploitation or support context. These approaches are also developed for a **more competitive product,** which will be easier to exploit, maintain, support and recycle than those of the competitors.

The selling price of the product is no more considered alone. The price of associated services for exploitation and support are also considered. It is why what is called PLM or Product Lifecycle Management is today considered as strategic within industrial enterprises.

As PLM implies a holistic view of the product and concerned actors/stakeholders all along the lifecycle of the product (that can have duration of 50 years), management of Product data and metadata in configuration is of prime importance all along the lifecycle, with growing

[8] The "digital ecosystem" concept has emerged around 2002 in Europe, as an innovative support for the adoption and development of information technology and communication to support the audacious goals of the Lisbon Council: growth, more and better quality jobs, with a stronger social inclusion, taking into account that the financial development of Europe is mainly based on a diffuse network of small and medium-sized enterprises and local innovation. This concept has reached its maturity and was included in the European Research Framework Program 7. Various tools are from an experimental phase. It is considered that this paradigm is applicable to a virtual company including a nebula of subcontractors of level 1, 2, etc.

importance of obtaining coherency of product data which are widespread between heterogeneous information systems of implied companies.

1.2.4 - System engineering

System engineering is an interdisciplinary field of engineering, dealing with the design and the management of large and complex engineering projects. It deals with work processes covering all the lifecycle of the product and tools to handle such projects, taking into account issues such as logistics and coordination of different teams. It overlaps with both technical and human-centered disciplines such as project management and control management. System engineering is considered both as a discipline in engineering and as a holistic and interdisciplinary approach which have to deal with complexity.
Models play important and diverse roles in system engineering, such as answering specific questions about the real world, representing the real world process or structure, or assisting a decision makers.

As **system engineering is becoming a best practice within the industry**. System engineering is closely related to fields such as product lifecycle management, requirement engineering, product data management and configuration management. As software tools are more and more used to support all the activities related to engineering, it is also closely related to product data management, computer aided design, and computer aided manufacturing.

1.2.5 - Nomadic and virtual working environments

Another important viewpoint to consider is the working environment of people, who are today working in a very different context. First users of applications are becoming nomads: because of geographical distribution of the enterprises, they have to travel but should still be able to access a transparent and secure way to their working environment with accurate quality of services. They have to deal with distributed informational resources, using different heterogeneous devices such as phones, laptops, static computers, etc. In addition, they have to collaborate with other people from other domains and from other enterprises, and to rely on expertise and know how they don't have. Finally, they depend on several independent organizations and belong to several communities.

From hierarchies, new organizations are becoming matrixes or networks. Enterprise applications have then to capture the different contexts a worker belongs to. It firstly aims to optimize efficiency of individuals, who are working more and more in several distinct organizations and need accessing several heterogeneous resources and software tools a secure and simple way. It is what is so called "virtualization" of the working space.

1.2.6 - Software industry and Commercial Off-The-Shelves

A special case of the virtualization of the companies applies to the field of the software engineering. Companies forsake more and more internal software in-house development and prefer to buy off-the-shelf products. It is the case in particular for the enterprise applications, i.e. those supporting of the processes of companies which are general and shared enough to create a market. This is the case for a large number of functions of the company which is non specific to a business domain: the scheduling of the resources, management of product data, management of the supply chain, human resources management, etc.

These software products, supporting enterprise processes, are based on generic models of business and disciplines. In order to support deployment within specific operational environments, organizational and technical, they usually can be specialized by parameterization, extension or customization.

These products are themselves composed of other Commercial Off-The-Shelves, which are more and more based on Information Technologies standards, and which can either be commercial products or commodities on the Web. I define "commodities on the WEB" as "free and open tools of quality industrial and based on standards". Examples are relational databases, Web servers, Web navigator, etc. In addition to the products supporting a business function or process, many software items aim at supporting the policies of integration of the enterprise applications. It includes systems such as enterprise application integration systems, enterprise service bus, application servers, enterprise portals, etc. There still one notices the emergence of commodities on the WEB. In addition to the operational tools, the modeling tools, the software design tools (conventional or supporting model driven development) and the development tools know the same trend, with the appearance of open source design and development platforms based on standards, as for example Eclipse, for which components supporting open standards are provided for coverage of software design, software development and software deployment.

In such a context, the interoperability of the applications should be reconsidered because of impact of the miscellaneous players and their role on the enterprise applications interoperability. Each player of the channel has a responsibility on the interoperability, but can insure alone neither data interchangeability and consistency nor interoperability of the applications in operation (consisted of application components, software items and parameterization settings, being business or technical). The importance of eBusiness framework for standardized specifications, controlled by business communities and "enterprises ecosystems[9] " is primordial.

Interoperability must also be rethought because of the impact of available commodities on the WEB. The first impact is that the commodities are open source software components, i.e. they are not black boxes. This aspect is crucial to solve interoperability problems in operation. This is illustrated by the fact that the only legal case where you can decompile commercial software is when you must solve problems of interoperability. The right to decompile or not does not arise with the Open Source. But changing code implies taking the responsibility concerning the modification of a software product, which is no longer guaranteed by the supplier, and may pose problems to upgrade to future versions of the product. The second impact is that commodities on the Web are based on formal definitions, i.e. formal managed and controlled interfaces definitions. You can from there consider applications engineering as composition of models based on reusable bricks, bricks being provided for design (model components), development (code libraries) or operations (distributed software components). Finally, these formal specifications are standard for large communities, often as part of a definition of interoperable business solutions, within a company or eBusiness community.

9

1.2.7 - New interoperability needs and challenges

In such a context, combining virtualization of the product, virtualization of the enterprise, virtualization of the applications and multi-disciplines collaboration implying distributed and nomad workers, some challenges are emerging for next generation of enterprise technical applications in term of interoperability. Firstly, it is required to ensure efficient global configuration management, and coherency of product information/data distributed between the different Product Data Management systems of the partners involved in collaboration. Secondly, it is more and more important to ensure efficient product data exchange, sharing and long term retention supporting in order to support engineering and collaboration processes. The concerned process are in particular, for product lifecycle management strategy, exploitation, maintenance and support processes, but also change and configuration management, and finally traceability for legal aspects. Thirdly, it is important to ensure seamless collaboration of actors despite the fact they are belonging to different organizations or communities. Interconnection of the applications they are working with must be ensured at a reasonable cost, in order to share and exchange digital information or services. The interconnected systems should support as well agile task flows (Lillehagen 2005) and structured workflow models flexible enactment within a nomadic environment. Finally, it is important to take into account emerging information and communication technologies, as well as trends within software industry and e-Business collaboration, in order to establish new collaboration processes with emerging complex applicative platforms. It must be done smoothly, without any disruption on operations and with the legacy systems.

Considering these challenges, it clearly appears that new expectations for interoperability are emerging, and that several kinds of interacting systems are to be considered when willing to deal with enterprise technical applications: organizations, applications, software product components and product. In order to address these new expectations, it is required to provide a clear definition of what technical enterprise application interoperability is. The definition I propose follows.

1.3 - Technical enterprise applications interoperability

Within the previously described context, and in order to be able to clearly state what the expectations are in term of interoperability of technical enterprise applications, I defined "Technical Enterprise application" the following way:

> TECHNICAL ENTERPRISE APPLICATION: any application supporting a business process related to product development, and used by several actors within the enterprise (by opposition to personal application). For example, product data and process management systems, all along the lifecycle of the product, are considered like technical enterprise applications, while individual tools such as office program or Computer Aided Authoring tools are not considered as technical enterprise applications.

From the experience gained during previous research projects, previous interoperability definitions can be considered as relevant for technical enterprise applications interoperability given some adaptations:

> INTEROPERABILITY OF TECHNICAL ENTERPRISE APPLICATIONS:
>
> (a) Ability of technical enterprise information systems to communicate with each other and exchange information.
>
> (b) Conditions, achieved in varying levels, when technical enterprise information systems and/or their components can exchange information directly and satisfactorily among them.
>
> (c) The ability to operate software and exchange information in a heterogeneous technical, knowledge and organization networks (i.e., one large network made up of several different local area networks).
>
> (d) Applicative enterprise technical Systems allowing several organizations and domains exchanging and sharing digital information, models and services in orders to collaborate together effectively.

In order to ensure whished interoperability, it is important considering within the state of the art the advantages and the drawbacks of relevant models of reference allowing to characterized interoperability (Chapter 2 - Models of reference for interoperability), of standards addressing interoperability (Chapter 3 – Standards for Interoperability) and of the different interoperability frameworks (Chapter 4 – Interoperability Frameworks).

Chapter 2 – Models of reference for interoperability

As previously stated, interoperability field is very rich and active. Different Interoperability models of reference have been produced these last years. Defining interoperability was done according different levels and following evolutions of paradigms used by communities and domains using information and communication technologies for complex systems.

2.1 - NATO C3 Technical Architecture (NC3TA)

NATO developed a reference model for interoperability, within the frame of his workgroup on technical architectures (NC3TA).
This model focuses on information exchange and addresses technical interoperability. It establishes interoperability degrees and sub-degrees for data exchange. The four degrees of "data" interoperability definitions are unstructured data exchange, structured data exchange, seamless sharing of data and finally the seamless sharing of information. They are detailed within Table 1.

Degree	Description
1. Unstructured Data Exchange	Exchange of human-interpretable unstructured data such as the text found in operational estimates, analyses and papers.
2. Structured Data Exchange	Exchange of human-interpretable structured data intended for manual and/or automated handling, but requires manual compilation, receipt and/or message dispatch.
3. Seamless Sharing of Data	Automated sharing of data amongst systems based on a common exchange model.
4. Seamless Sharing of Information	Universal interpretation of information through data processing based on cooperating applications.

Table 1: Degree of data interoperability

The degrees were intended to categorize how effective operational data exchange could be. This model does not characterize the information systems. It is the purpose of the Level of Information System Interoperability System (LISI) model described below. In December 2003, the NC3TA model was updated in order to closely reflect the Level of Information System Interoperability model.

2.2 - Levels of Information System Interoperability

The **LISI (Levels of Information Systems Interoperability)** model was developed as one of the C4ISR (Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance) Universal Reference Resources to define interoperability between information systems. It provides a mechanism to define the maturity of information systems and a way to proceed from one level to another.

LISI defined interoperability for information systems on the basis of a matrix crossing five levels (isolated, connected, distributed, integrated and universal systems) affecting four characteristic of information systems (processes, applications, infrastructures and data). The levels are detailed on Table 2.

Level	Description
0- Isolated Systems	No connection exists between applications, exchanges are made manually
1- Connected Systems	Applications are connected by electronic means, but are fully separated as well as their data. Some exchange of semi-structured data is possible.
2- Distributed Systems	Applications have some common functionality, but are separated as well as their data. Heterogeneous data sharing is possible.
3- Integrated Systems	Applications are separated but are sharing common data (databases). They can collaborate a sophisticated way.
4- Universal systems	Applications and their data are shared within the enterprise or between several enterprises. Collaboration processes between applications are developed.

Table 2: Levels of Information System Interoperability

These five levels impacts four characteristics (PAID):

- **Processes**, which represent practices, architectures and standards allowing to exchange information;
- **Applications,** which represent software applications allowing exchange, treatment and processing of information ;
- **Infrastructures** (or Architectures), which represent technical environment (material, network, system) allowing interaction between applications ;
- **Data** represent format and protocols allowing exchange of information using a common semantic.

Interoperability of enterprise applications required to consider, at the minimum, applications classified at the level one of this typology, taking into consideration collaboration procedures, applications, infrastructures and data.

LISI focuses on technical interoperability, with a strong focus on information exchange and sharing. The model considers neither environmental nor organizational issues contributing to establishment, construction and maintenance of interoperability all along the time. It led to definition of the Organization Interoperability Model.

2.3 - Organization Interoperability Model

Acknowledging some limitations of the LISI model, (Clark and Jones 1999) proposed an approach, related to organization of enterprise, in order to enhance interoperability of applications. It is based on an interoperability model for organizations with five levels, described within Table 3.

Level	Description
0- Independent organizations	Organizations are independent and no exchange is ready and possible. These organizations can collaborate only through human interaction.
1- Ad- Hoc organizations	At this interoperability level, organizations don't have a common framework but some isolated actions could exist.
2- Collaborative organizations	At this interoperability level, organization actors must have put in place standardized frameworks for understanding (ontology) but still being distinct.
3-Integrated organizations	At this interoperability level, knowledge sharing is required between all the actors of the enterprise.
4- Unified organizations	Organizations own objectives, system of values, decisional structure and core knowledge shared by all the actors.

Table 3: Interoperability levels of organizations

The focus of Organization Interoperability Model is on the human activities and usage aspects of operations that are to be undertaken. (Clark and Jones 1999) provides a mapping between Organization Interoperability Model and Level of Information System Interoperability, illustrated by Figure 3 extracted from (Clark and Jones 1999).

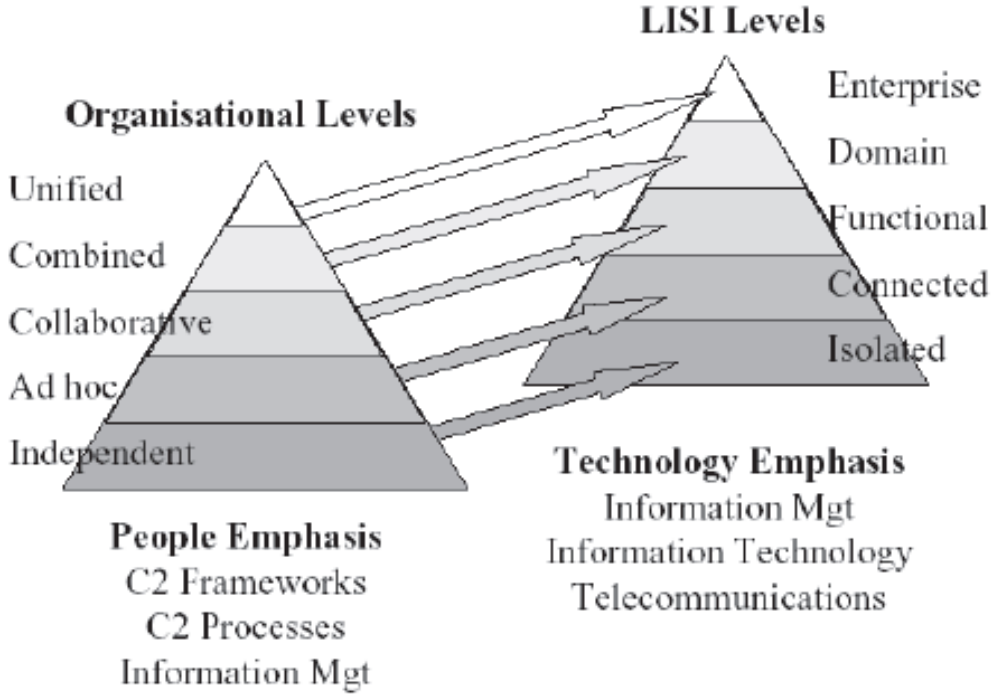


Figure 3: Alignment between Organizational and Information System Interoperability levels

This model of reference is important, as it leads to consider operations which are enabled by interoperability, i.e. organizational environment. Neither the Organization Interoperability Model nor the Levels of Information System Interoperability model characterizes the information systems. It is the purpose of the Level of Conceptual Interoperability Model (LCIM) model.

2.4 - Level of Conceptual Interoperability Model

LCIM (Levels of Conceptual Interoperability Model) was proposed by NATO and adapted in (Tolk and Muguira 2003), in order to be able to address some aspects of organizational interoperability of applications, which were not addressed by previous models. A conceptual framework with five levels was proposed. Each level characterizes how data are exchanged between applications and available interface documentation. The layers of the LCIM model are detailed within the Table 4.

Enhanced LCIM model was derived from the LCIM model. Subsequent to the publication of the original LCIM model, (Tolk 2003) and (Tolk 2004) developed further an enhanced model, influenced by (Zeigler 2003) and (Hofmann 2003). The enhanced LCIM model incorporates changes to address interoperability issues other than those related to data interchange, and also adds a pragmatic level, which establishes that the receiver of information not only knows what it means, but also understands how it should be used. Levels of enhanced LCIM are detailed in Table 5.

Level	Description
0- System Specific Data	Black box components with no interoperability or shared data.
1- Documented Data	Shared protocols between systems with data accessible via interfaces.
2- Aligned Static Data	Common reference model with the meaning of data unambiguously described. Systems are black boxes with standard interfaces. However, even with a common reference model, the same data can be interpreted differently in different systems.
3- Aligned Dynamic Data	Use of data is defined using software engineering methods like Unified Modeling Language. This allows visibility into how data is managed in the system. But even systems with the same interfaces and data can have different assumptions and expectations about the data.
4- Harmonized data	Non-obvious semantic connections are made apparent via a documented conceptual model underlying components. This goes beyond Level 3 because the assumptions concerning the data are made apparent.

Table 4: LCIM levels of application interoperability

Level 0- system specific data	No connection is established at all. Stand-alone systems with no interoperability
Level 1- technical	At technical level, a communication protocol exists, allowing data exchange between participating systems. Physical connectivity is established allowing bits and bytes to be exchanged. Underlying networks and protocols are unambiguously defined.
Level 2- Syntactical	At syntactical level, data can be exchanged in standardized formats, i.e., the same protocols and formats are supported. This layer defines structure.
Level 3 – Semantic	At semantic level, not only data but also its contexts, i.e. information, can be exchanged. The unambiguous meaning of data is defined by common reference models. This layer defines meaning.
Level 4- Pragmatic	At pragmatic level, interoperating systems are aware of the methods and procedures employed by each system. Use of date, i.e. context of application, is understood. This layer defines context.
Level 5- Dynamic	At dynamic level, systems operate on data over the time, despite changes of the systems. They even can take advantages of the change. This layer focuses on change.
Level 6- Conceptual	At conceptual level, a common view of the world is established, i.e. an epistemology. This level not only comprises the implemented knowledge, but also the interrelations between these elements. It implies conceptual models are documented based on engineering methods, enabling their interpretation and evaluation by other engineers. It requires a fully specified, but implementation independent formal model (and not just text describing the conceptual idea).

Table 5: Enhanced LCIM interoperability levels

In order to reach level six within eBusiness context, models and standards are required that provide structure beyond consistency of data and understood interfaces. This can't be achieved without considering organizational aspects related to harmonized strategy. It is the goal of Layers of Coalition Interoperability (LCI) model.

2.5 - Layers of Coalition Interoperability (LCI)

(Tolk 2003) establishes a reference model for coalition interoperability the LCI (Layers of Coalition Interoperability) model, described by Figure 4. This model is intended to facilitate discussion on technical and organizational (political and military) support required for interoperable solutions. It is not intended to be a substitute for other models. The four lower levels of the model deal with technical interoperability. The knowledge/awareness level provides a transition between technical interoperability and organizational interoperability, which is represented by the top four levels. With enhanced Layer of Conceptual Interoperability model and Layer of Coalition Interoperability model, it is not clearly defined

if we are considering physical systems, information systems or organizational systems, but it appears clearly that organizations are playing an important role when willing to obtain the best level of interoperability.

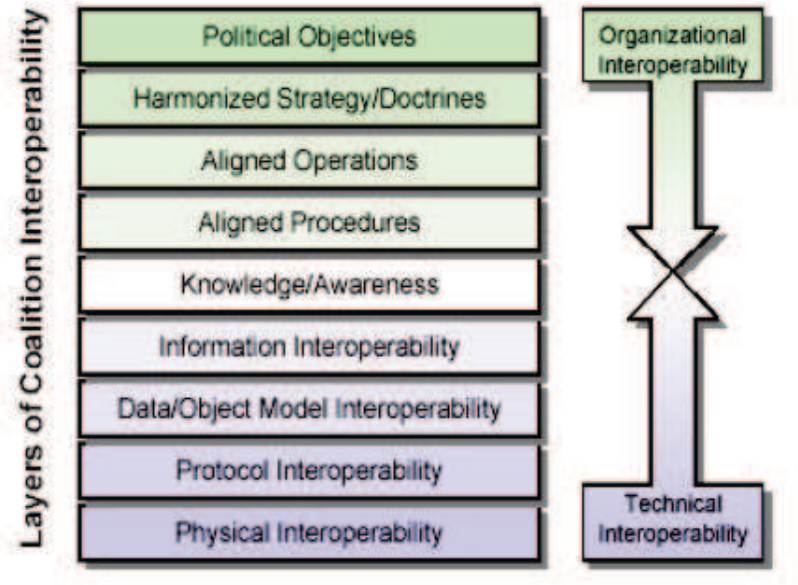


Figure 4: the layers of Coalition Interoperability (from Tolk 2003)

All these models are considering neither the technologies, nor the processes nor the quality. It is more the purpose of the following models.

2.6 – Interoperability scale reference model

(Obrst 2009) proposed technological framework for interoperability, considering that for eBusiness and eCommunities, the need for loosely coupling technologies in conjunction with asynchronous communication and higher semantic expressivity is mandatory if willing to achieve the semantic interoperability level. Data exchange technologies are classified according their semantic explicitness, while software engineering technologies are classified according to looseness of coupling. Definition of seven levels is provided, including data, object, component, application, system, enterprise and community. Three types of integration are considered: syntactic, structural and semantic. All the types of integrations are to be considered for all the levels, but they are not balanced the same way concerning need of semantic, structural or syntactic integration. The higher the level is, the higher the semantic integration part is important, while syntactic and structural integration parts decrease, as described by Figure 5.

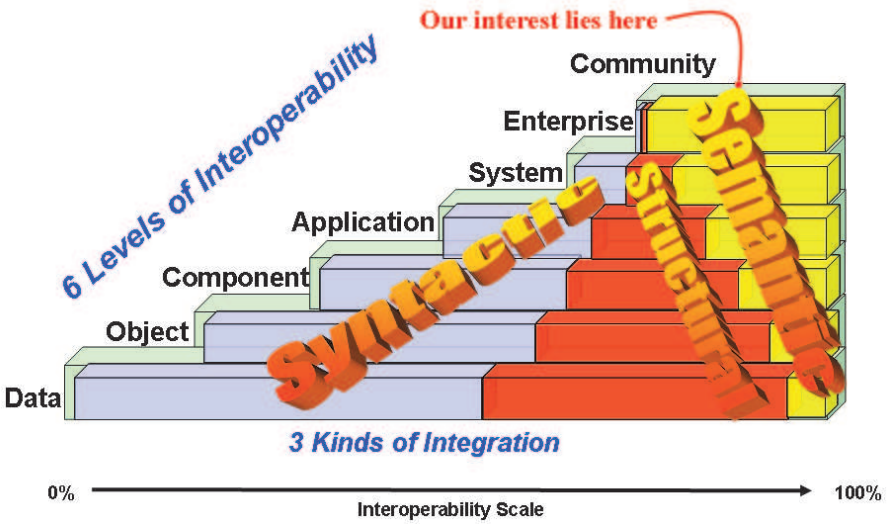


Figure 5: Dimension of interoperability and integration (Obst 2008)

Syntactic integration concerns mainly modeling languages, structural integration concerns mainly model transformation and semantic integration requires formal logic description. In association, Figure 6 shows an ontology spectrum, positioning the different technologies in term of looseness of coupling and semantic explicitness. For eBusiness and eCommunities, it highlights the need for loosely coupling technologies in conjunction with higher semantic expressivity and asynchronous communication.

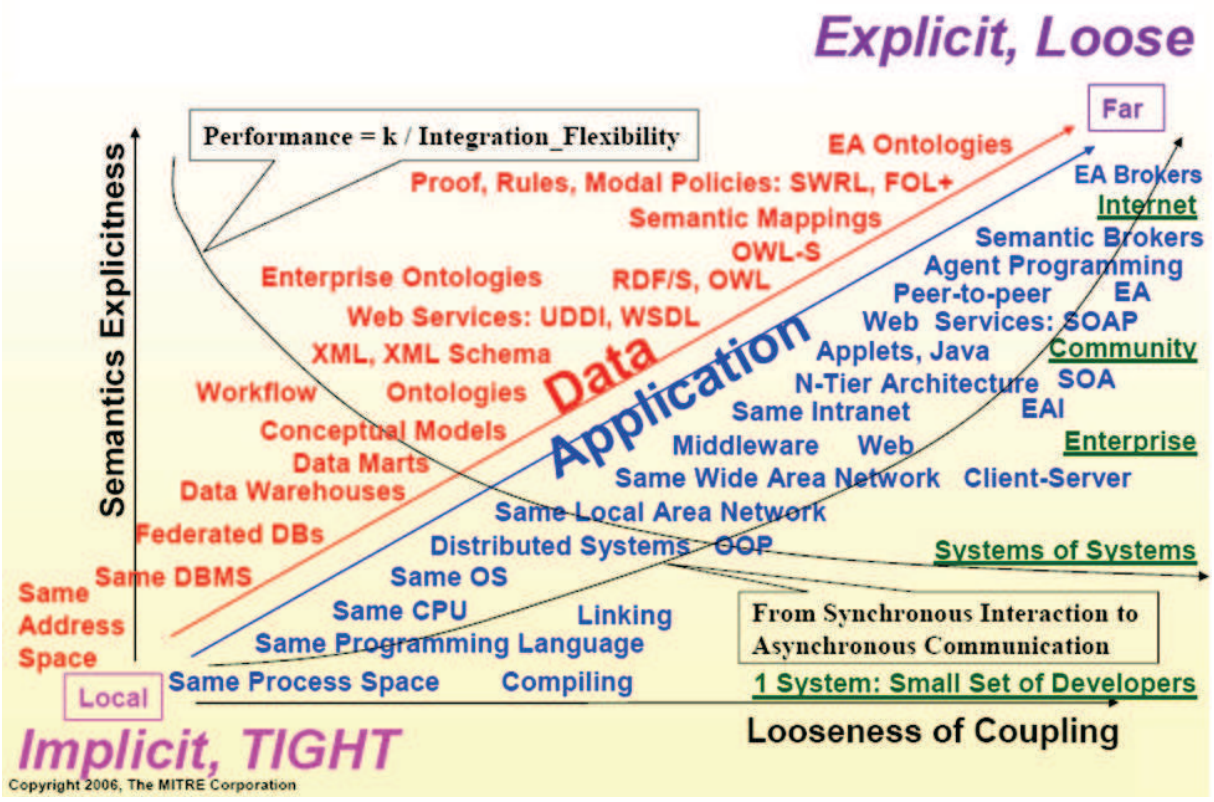


Figure 6: Technology and language spectrum of eBusiness Interoperability (Obst 2009)

Such a model of reference is very helpful for a community having to defined frameworks and standards to use. Choice should be made according interoperability to achieve is local, enterprise or community. Local interoperability required lower semantic explicitness than community and eBusiness interoperability. In addition, the higher semantic explicitness is, the higher looseness of coupling is.

As it does not focus on organizational and informational systems, it completes previous model of reference. Dynamic aspects related to systems development, eCommunity governance, maturity and software engineering are also to be considered. The System Of Systems Interoperability (SOSI) model addresses these points and is described in the next section.

2.7 - System of systems Interoperability model (SOSI)

The SOSI (System of Systems Interoperability) model, defined in (Morris et Al. 2004), is based on the observation that interoperability in operations can't be achieved without:

- activities related to acquisition of a system (program management);
- activities creating and sustaining interoperability with focus on architecture, standards and Commercial Off-The-Shelves;
- activities related to operation, based on interactions between systems and with users, including operational support.

Figure 7 shows the three sorts of activities that must be further aligned for effective interoperability. The SOSI model defines three type of interoperability: programmatic, constructive, and operational.

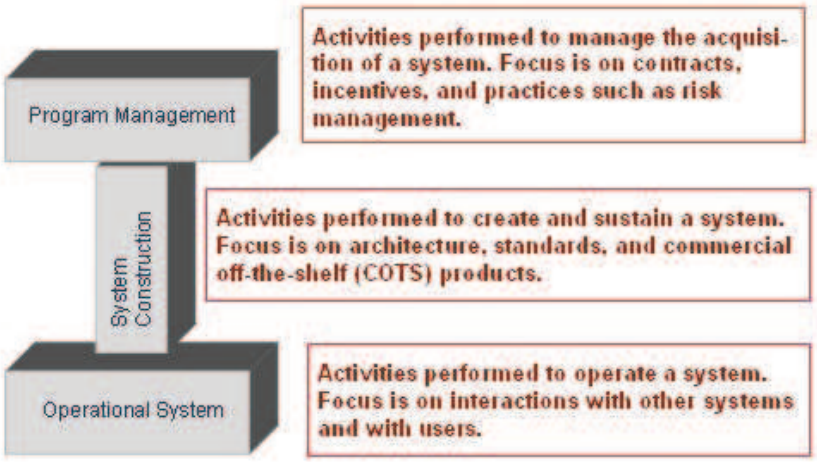


Figure 7: SOSI model from (Morris .et .al. 2004)

Programmatic Interoperability is related to the required cooperation between programs or projects building interoperating systems. Reaching some agreement on requirements regarding the interoperation, the formats of information to be exchanged or shared, and the quality if service expectations and the control of versions of the systems are different ways to achieve it. It is important to share information between people in programs in order to achieve mutually beneficial results and good agreements. The focus is on strategy (why).

Constructive Interoperability is related to engineering processes, standards, and specific technical agreements to put in place in order to achieve interoperability. Thus, constructive interoperability is aligned with how are built interoperable systems. Current available sets of standards are nevertheless today not sufficient, as they are not covering all the required aspects, as for example quality. Recommended practices and certification for conformance are most of the time to be defined.

Operational Interoperability is related to the ability of interoperating systems to contribute to achievement of a greater human goal in the operational environment. It implies delivering the right information at the right time and in the right format, but also consistently delivering it without undue effort on the part of humans, including after system upgrades.

Operational interoperability is thus always the ultimate goal, and involves (for example) common methodologies and operational procedures as well as collaborating systems. My thesis was motivated because too many breaks and important issues are today identified to achieve it.

As other previously presented models, the SOSI model makes the distinction technical aspects of interoperability and those aspects of interoperability that are required between organizations. SOSI draws a further distinction between programmatic interoperability involving organizations responsible for system development, and operational interoperability, involving the organizations and individuals using the systems.
Nevertheless SOSI does not address different issues such as maturity of involved organizations. Establishment of technical framework for application engineering taking advantage of legacy interoperable platforms and emerging technologies is also not considered. Following models addresses these issues.

2.8 - Interoperability Maturity Models

Several initiatives put emphasis on Maturity models, focusing on communities, technologies or information systems. Several Interoperability Maturity Models are being produced.

The National eHealth Transition Authority Interoperability Maturity Model (NEHTA 2007) allows the analysis of both work products and organizational practices supporting interoperability. It provides maturity of practices levels (aligned with Enhanced LCIM), set of interoperability goals and assessment framework. They distinguish several domains: local domains, enterprise domains and finally the community domains which are our target for eBusiness interoperability within a networked organization.

Researchers from LAPS (N. Daclin et Al., 2006) are proposing a conceptual framework proposing maturity levels (empirical, unexploited, interoperable and evolutionary) for different interoperable systems (decision, information and physical) and characterized by interoperability types (difficult, punctual, adaptive and continued).

Let's have more details on these two models of reference.

National eHealth Transition Authority(NEHTA) Interoperability Maturity Model

Authority (National eHealth Transition Authority) proposes an Interoperability Maturity Model (IMM) allowing the analysis of both work products and organizational practices supporting interoperability and provides tools for identifying programs to improve interoperability maturity.
The Interoperability Maturity Model consists on several components, maturity of practices levels, set of interoperability goals and assessment framework. It is dedicated to a precise eBusiness community considering governance aspects.

The five maturity of practice levels are inspired by the widely used Capability Maturity Model Integration (CMMI[10]) framework, considering interoperability goals, interoperability practices and interoperability work product. The proposed IMM maturity levels are detailed by Figure 8.

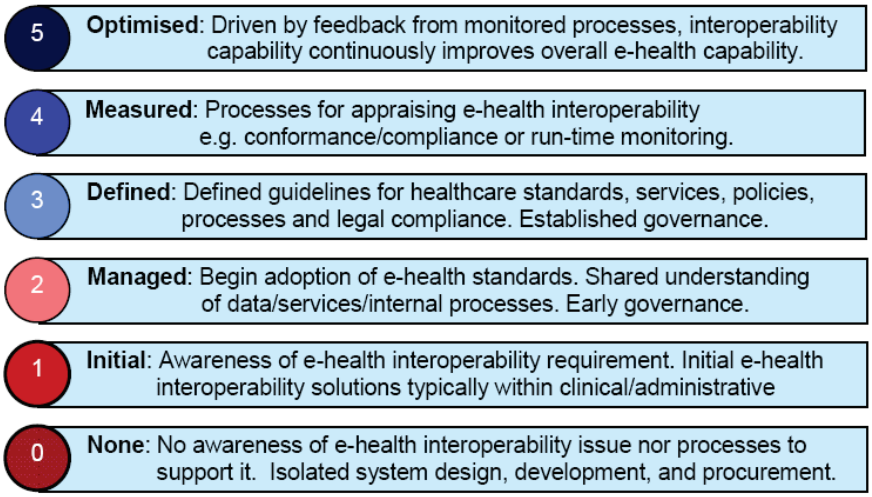


Figure 8: Interoperability maturity levels (NEHTA 2007)

Despite specific context (Australia and Health), the model is very generic and could be apply to any country or community. Different organizational granularities are considered for boundaries enclosing a set of co-located constraints for which NEHTA uses the term "interoperability domain". The considered domains are local, enterprise and (eHealth) community. The different levels of maturity are defined for each as follows:

	Local	Enterprise	e-Health Community
5		Continuous interoperability improvement Enable organisational goals Inward and outward: EA as a binding process	Continuous interoperability innovations Enables Community/Social goals Emergence, dynamics, adaptation
4		Impact of EA/SOA on organisational goals Identify interoperability weak points	Established Certification program Community SLA monitoring
3	Local standards governance Early architecture principles	Enterprise-wide standards/governance Generic EA principles augmented with SOA Early organisational interoperability	'Community' architecture Interoperability governance defined Governed use of open standards
2	Shared early interoperability experience ICT standards adopted	Some defined EA processes Business and IT committed to EA	Interoperability frameworks Community/National collaboration Governance in development
1	Technical integration efforts Ad-hoc solution architecture	Individual champions for technical and information interoperability Efforts underway for executive EA buy-in	Community interoperability vision Policy makers delivering social benefits Ad-hoc use of standards
0	Isolated design/development Siloed procurement	No interoperability awareness No processes to support EA	No processes to support cross-organisational interoperability

Figure 9: maturity for local, enterprise or community interoperability (NEHTA 2007)

[10] Capability Maturity Model Integration: trademarked process improvement approach that provides organizations with the essential elements for effective process improvement. It can be found on Software Engineering Institute web site at http://www.sei.cmu.edu/cmmi/tools/index.cfm

Common interoperability goals are 'reuse', 'evolution', 'standard basis', 'scope', 'scalability', 'configurability' and 'explicitness'. The organizational interoperability goals are 'business focus', 'governance' and 'overhead to change'. The informational interoperability goals are 'data format us semantic', 'meta-data', 'ownership or rights' and finally 'common building blocks'. The technical interoperability goals are 'interface specification', 'functional decomposition', 'communication protocol', 'n-tiers architecture' and 'technical policy separation'. Interoperability Maturity Model also proposes a basic assessment framework.

Interoperability Maturity Model from LAPS

(D. Chen et Al. 2006) proposes a conceptual framework distinguishing different interoperability systems, interoperability types and levels of maturity.

The interoperability systems are:

- Decision: several decisional systems belonging to different partners are to be identified and it should be enable to make them work together
- Information: it concerns creation of relations between the different information systems of the partners
- Physical: several physical systems have to work together.

The interoperability types are conceptual (related to syntactic and semantic aspects of the information to be interchange), organizational (related to responsibility, authorization and organization definitions) and technological (related to compatibility aspects of information and communication technologies).

Maturity is then defined at the intersection of these two axes, making distinction between several maturity levels, described by Table 6 and illustrated by Figure 10.

Empirical	Behavior of the 3 systems is known but not modeled, information which can be externalized is not known and organizational structure of the enterprise is not clearly defined.
Unexploited	Systems are modeled but not exploited, information which can be externalized are known but non stored nor structured, organizational structure is know but not exploited.
Interoperable	Systems are modeled and unexploited, information which can be externalized is known, stored, structured with secured access and structural organization of the enterprise is clearly known and exploited.
Evolutionary	Models are adapted according evolution of the enterprise, information which can be externalized are known, stored, structured with secured access, and organizational structure of the enterprise is adapted to evolutions.

Table 6: Interoperability Maturity levels from (Chen et Al. 2006)

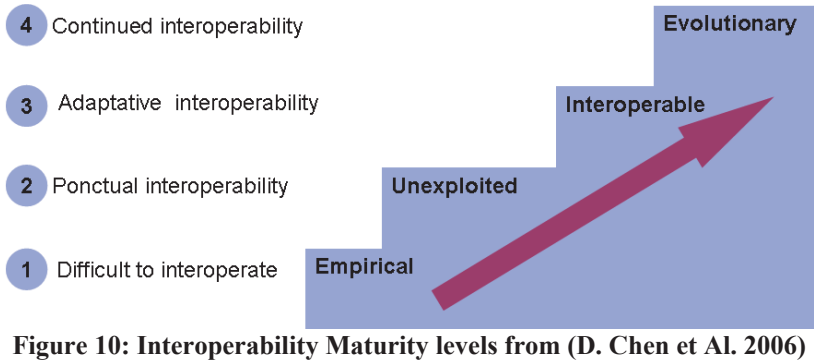


Figure 10: Interoperability Maturity levels from (D. Chen et Al. 2006)

Maturity levels can be interpreted to characterized interoperability as difficult, punctual, adaptive and continued. Finally “a priori” and “a posteriori” interoperability is distinguished, “a priori” interoperability depending on maturity.
The maturity related interoperability models are completing proposed models by focusing on maturity of organizations and teams willing to achieve interoperability. Final goal is to obtain continuous interoperability, with strong focus on preparing it in advance with accurate governance and decision making. Technological aspects are not considered.
All the previously described models don’t propose any approach to build interoperability considering together organizations, knowledge and technologies aspects. IDEAS reference model, enriched by ATHENA, proposes a holistic approach which encompass these different aspects when willing to achieve interoperability of enterprise applications.

2.9 - IDEAS and ATHENA Reference Models

The high level **ATHENA reference Model** deals with holistic approach to address interoperability of enterprise applications, taking into account enterprise modeling, ontology, model driven engineering, service oriented execution platform and cross organizational executable collaboration processes. It is one foundation of my research work on this thesis. This model was produced by the Enterprise Application Interoperability research community within the IDEAS Roadmap project, and then used within ATHENA and INTEROP programs and projects. It reflects that interoperability of applications must be achieved at enterprise, knowledge and technology levels, using semantic mediation between applications and between levels. Figure 11 illustrates it, describing considered layers: business, knowledge, applications and data.

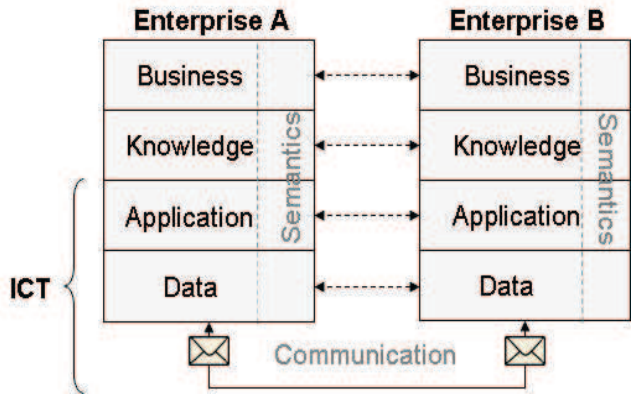


Figure 11: IDEAS reference model

A holistic approach is proposed, composing solutions coming from several communities such as Enterprise Modeling Community, Ontology Community, Model Driven Engineering Community and Service Oriented Architecture Community. Unlike National eHealth Transition Authority Interoperability Maturity Model, ATHENA model and associated program did not consider a satisfying way neither Information and Communication Technologies standards, neither early application and information systems architecture principles, neither interoperability weak points, neither legacy interoperability frameworks, neither interoperability governance, neither governed use of open standards nor continuous interoperability with auto adaptation to innovation. Numerous issues which are described within chapter 5 comes from analysis of ATHENA results (N. Figay , 2006a and 2006b), when applying it for Networked Collaborative Product Definition platform. Identified drawbacks motivated the launching of this thesis.

2.10 - Conclusion about interoperability related models

The different models for interoperability which have been developed these last years highlight evolution of interoperability research field and emerging needs and approaches to deal with interoperability.
Firstly, the main focus, which was initially on data and information exchange, evolves over the time, progressing from data format and structure to semantic consideration in one hand, from electronic documents to electronic computable models in the other hand.
Secondly, new set of technologies have been developed in order to address interoperability within extended and virtual enterprise, based on internet standards, and with growing importance of loosely coupling technologies and high semantic expressiveness of the new languages.
Thirdly, the focus is not only on technologies, but also on organizations and on processes to obtain interoperability. Communities of practices are emerging, leading to knowledge sharing on interoperability and on establishment of maturity models of domains and communities willing to reach continuous and operational interoperability.
Fourth the different models of reference also point out that different systems are to be considered jointly: information systems, organizational systems, decision systems, communication infrastructures and their combinations. The fact that the considered systems are not often clearly stated and defined is one of the issues identified within Chapter 5.

Using the different models for interoperability described within this chapter, I characterized the ideal collaborative system as detailed by Table 7. Within emerging networked organizations, collaboration methods, information systems and technologies are heterogeneous. Consequently, the ideal collaborative system should support seamless exchange of information between distributed information systems and collaborative organizations, with semantic integration and loosely coupled technologies and asynchronous communication. Interoperability of operation application should be constructed and programmed by mature organizations aiming to achieve adaptive and continued interoperability with evolutionary decision, information and physical systems.

Characterization of ideal collaborative system
NC3A: seamless exchange of information
LISI: distributed information systems
OIM: collaborative organizations
LCIM: semantic integration
MITRE: semantic Integration with loosely coupled technologies and asynchronous communication
SOSI: programmatic, Constructive and Operational interoperability
IMM(s): adaptive and continued interoperability with evolutionary decision, information and physical systems; matured selected architectures of reference and domain standards

Table 7: characterization of ideal collaborative system

It can be achieved only with selection of mature architectures of reference and domain standards. Most of the interoperability models highlight growing importance of the role of standards and architectural frameworks. However, most of the standards and interoperability frameworks are produced independently by competing standardization organizations on the basis of heterogeneous paradigms. Interoperability models don't provide any approach for selection of relevant standards, frameworks and architectures to prepare and support

operational interoperability for technical enterprise application within virtual enterprise for a given domain. The next chapters provide state of the art and legacy concerning standards and frameworks I identified as relevant in order to build expected interoperability within the particular domain of technical enterprise applications.

Chapter 3 – Standards for Interoperability

3.1 - The different kind of standards

"A standard is a document established a consensual way and approved by a recognized organism, which provides, for common and repeated usage, rules, driving lines or characteristics, for activities or their results that provide guarantees for an optimal level of order in a given context." (Wikipedia).

It is to be distinguished from a "De facto" standard:

"A **de facto standard** is a custom, convention, product, or system that has achieved a dominant position by public acceptance or market forces (such as early entrance to the market). *De facto* is a Latin phrase meaning "concerning the fact" or "in practice".

It is important to point out that "de lego" standards may not be used at all or of an insufficient quality to be used, while "de facto" standards are always used, are having already obtained a dominant position.

Currently, numerous enterprises are currently pushing specifications of their own proprietary products or processes to standardization in order to reinforce, to protect or to install their dominant position on the market. In such a case, they are most of the time forcing the consensus and are still keeping the control of the evolution of the specifications or of the processes. It is the reason why numerous discussions are occurring concerning definition of an "**Open standard**", with some difficulties to agree due to diverging interests of the numerous concerned stakeholders and to the different ways to interpret the terms "standard" and "open". An **open standard** is publicly available and has various rights to use associated with it. Openness can also be related on how it was designed (e.g. open process).

I consider that the perfect combination for enterprise application interoperability within virtual enterprise is to work with de jure standards which are at the same time de facto and open standards, such as those provided for and used by Internet. It allows balancing power of the different members of an ecosystem through a win-win situation, which is an important pre-requisite for long term collaborations.

In addition, several types of standards can be considered:

- Standardized definition is formally established terminology. It can be dedicated to people (e.g. dictionary) or for automation (e.g. schema).
- Standardized specifications, which are explicit sets of requirements for an item, material, component, system or service, are often used to formalize the technical aspects of a procurement agreement or contract. Such standards are often produced for programming by contract, when for example defining interface between software components, or when defining exchange protocols. They are also associated to service oriented approaches and architecture which are providing the framework for defining components of architecture and associated services.

- Standard test methods describe procedures which produces a test result. They may involve observation or technical measurement. Such standards are often used in order to qualify compliance of an implementation with a standardized specification.
- Standard practices or procedures give sets of instructions for performing operations or functions. They can be formalized for usage by people (e.g. methodology, quality process) or for automation (e.g. workflow model).

When computer dependant technologies, these different kinds of standards can be specialized according such or such used technology or artifact. It may refer to systems, hardware, file formats, protocols, programming languages, etc.
When considering disciplines or organization, the standards can be computer independent or computer dependant, in order to support computer aided activities. In this last case, it is most the time targeted to produce formal specifications which are specific proprietary solutions independent, through usage of standardized and open formalisms.
Numerous standardization frameworks exist, which are most of the time combining these different types of standards, but are usually focusing on one dominant aspect: information, service, process or architecture.
Numerous standardization organizations exist today, being "official" standardization bodies such as International Standardization Organization, or simple international consortia.
The important organizations to consider in the context of the thesis are those related to the WEB, to eBusiness, to software and system engineering and to the considered domains and disciplines (e.g. manufacturing). They are important because they are providing relevant models of reference which are to be used by the different stakeholders and actors involved for establishing e-business collaboration within the virtual enterprise.
As the considered vertical domain is manufacturing, organizations providing manufacturing standards dedicated to eBusiness are also important, encompassing ISO TC184 SC4[11], OMG MANTIS[12], OASIS PLCS[13] and INCOSE[14]. Within European Aerospace & Defense industry, ASD STAN[15] is to be added.

3.2 - The different kind of standards for interoperability

According different analyzed, used and compare standards and standard frameworks, I proposed in this section a categorization for the interoperability standards which will be useful to address the problem addressed by this thesis. Within the proposed classification, I also distinguish within each category the used languages, the associated constructs and what is described by the languages. For example, a standardized business process can be formalized using natural language, or several modeling and programming languages. A dictionary can

[11] ISO TC184 SC4: standard body within ISO that sets standards for Automation systems and integration with a focus on Industrial Data. Web site: http://www.tc184-sc4.org
[12] MANTIS: Manufacturing Technology and Industral System task force at Object Management Group. Web site: http://mantis.omg.org
[13] OASIS PLCS: OASIS Product Life Cycle Support (PLCS) technical committee, dealing with deployment of an international standards for product data exchange (ISO 10303) to support complex engineered assets from concept to disposal. Web site: http://www.oasis-opne.org/committees/tc_home.php?wg_abbrev=plcs
[14] INCOSE: International Council on Systems Engineering, a not for profit membership organization founded to develop and disseminate the interdisciplinary principles and practices that enable the realization of successful systems. Web site: http://www.incose.org/
[15] ASD STAN: Aerospace and Defence Industries Association of Europe, establishes, develops and maintains standards requested by the European aerospace industry. Web site: http://www.asd-stan.org

describe a business terminology, which will then be formalized as a model using a formal language for computation.

3.2.1 - Data interoperability

Data interoperability is related to definition of interoperability related to communication and information exchange, with a focus on data. I distinguish data interchange, data sharing and data composition. With **data interchange**, there is a transfer of control on data, which are transferred from one system to other system. It is for example related to exchange by mean of files through import and export mechanisms. With **data sharing**, the data are access by a system through the system controlling them. It is for example related to centralized database or file repository, remotely accessed by distributed clients. With **data composition**, several sets of data, under the control of different systems, are exposed and composed through establishment of links which are under the control of a "transversal" system.

Referring NATO C3 Technical Architecture, data can be unstructured or structured. Studying different kind of de facto open standards for data exchange, I consider that unstructured, semi-structured or highly structured data must be distinguished, when willing a useable classification for relevant legacy standards related to data interoperability and electronic collaboration. **Unstructured data** are for example text file, office document, etc. It means that information is mainly interpreted by the people and not by the computer, except for formatting and printing. **Semi-structured data** are for example XML document with document type definition. It means that it mixes of literal information interpreted by people and coded information dedicated to computer, with associated business domain semantic. **Highly structured data** are for example those within relational database, XML document with XML schema, STEP model, UML models, etc. It means it provides highly codified data dedicated mainly to usage by computer.

In practice, technologies are allowing **mixing the different kinds of data**. For example, it is possible to include XML content within text field within databases. As XML can be used both for electronic documents and exchange of data between databases, it is also possible to mix structured and semi structured data. The primary intention of XML was nevertheless to deal with electronic documents, and was then extended to numerous other purposes (data exchange between software and also serialization for programming or meta-language). Distinction between **syntactic and semantic formalism** can be used for highly structured data exchange, with usage of formalism allowing automated usage of taxonomies, semantic graphs or ontological models.

Finally, for any kind of interoperability, the business logic will be **described for people**, by mean of natural language, in order to provide precise and explicit terminologies, for different industrial domains and area of knowledge. They are also often formalized using such or such language, as vocabularies in XML, application protocols in EXPRESS or conceptual models using UML. But the business logic and semantic should remains the same even if the different used languages are not equivalent and are not used with the same intention.

For the manufacturing domain, product data management and product lifecycle management, the more important standards for data exchange and sharing are the STEP application protocols.
STEP is described within section 4.3.

3.2.2 - Service invocation

Service invocation is usually defined according object or system paradigms, by numerous open information and communication standards. Services are defined by mean of definition of interface definitions between interacting components and systems.

For **programming language**, it is usually defined by mean of set of functions. Standardized libraries exist. They can be made available by mean of source, or compiled libraries with published programming interfaces that can be reused by means of linkage or dynamic invocation. Consequently open standardized programming languages constitute a first family of standards to consider for specification of standardized services. These languages can rely on definition of functions, procedures or object interfaces. As an example, language C was used in order to specify first standardized interfaces of workflow management coalition. Another example is the Java binding of Standardized Data Access Interface (SDAI) of ISO 10303 (STEP). When willing to combine usage of code formalized with multiple languages, most of the languages are providing libraries allowing establishing bridges in order to invoke and use code written in other languages, at development time. Another way consists in possible linking with binary code produced with different languages.

Such ways to proceed don't consider **distributed applications** already deployed on different machines of an intranet or of an extranet. For this need, several languages are supporting definition of **remote operation invocation** between a service consumer and a service provider, written with the same language (e.g. Remote Method Invocation for Java which allows several applications running on different virtual and physical machines to communicate). Most of the time the used languages, exploitation systems and machines are heterogeneous, bringing the need for middleware systems allowing distributed heterogeneous applications, with communication buses and interface definition languages. These languages can be considered as programming language, but they are operating system and machine independent. The fact that such languages can also be software providers independent is also important within an eBusiness context. It allows communities of interest to develop open interface specifications which will enhance business development and avoid monopolistic situations. It is probably one of the reasons why organizations such as Object Management Group or World Wide Web were created.

For the technologies developed by Object Management Group, and in particular the Common Object Request Broker Architecture, remote services are made available though publication of object interfaces grouped in packages. Object paradigm is particularly accurate when willing to describe, design or implement a system, with decomposition description, encapsulation, inheritance… An object is grouping data types and object types attributes, and also the methods. It is important to consider that invocation of service, but also access to the attribute for reading or writing, can be considered as set of operations. For attributes, it will be done by providing getter and setter operations for each attribute. A method attached to an object can be considered as an operation that can be performed by this object. It is important because it allows making the link with other service oriented technologies, which are not object oriented. As a matter of fact, some communities considered that object orientation is too complicated for effective usage, and decided not to use it. It is in particular the case for the WEB community which developed XML technologies.

The WEB community, through the World Wide Web Consortium, developed a framework for web services. WEB community provided a language allowing defining a service as a set of

operations, with input and outputs, that can be invoke remotely. Communication is made using XML Messages (Simple Object Access Protocol) with content that can be formalized with XML or any other language.

In all cases, for collocated or distributed communicating components, being object oriented or not, a service can be considered as a set of operations which is provided by a service provider, and consumed by a service consumer. It is important to point out that for a given service, consumer and provider can be a person or any kind of system. It is important because, even if usually not considered by programming languages, persons are service consumers or providers. Such consideration is fundamental when considering human/machine interfaces or business processes supported by enterprise applications.

It is also important to consider that remote service invocation implies data exchange. Data transportation can be done using different means, such as exchange of messages or exchange of files. It can be done an automated way or a manual way, implying human operators. It will imply usage of infrastructure capabilities such as messaging, file exchange, etc.

Usually, standards for distributed services are providing architectural framework with communication and interaction protocols and services. Formal neutral languages are provided to define the service contract between service provider and consumer. The service contract can be defined a priori, and be the reference. It is then a standard. Interoperability is ensured by conforming with well specified standards. The service can also be defined a posteriori or during the design. Interoperability is ensured by being able to consume the service easily, quickly and with reasonable resources.

The **relevant information and communication technologies standards** to consider are those proposed by **Object Management Group** and by **W3C** open organizations. Object Management Group proposes in particular Common Object Request Broker Architecture, with associated Object Management Architecture, CORBA Component Model, and CORBA UML profile.
Web World Wide Consortium proposes in particular a set of XML based standards, with messaging protocol and language for defining services exchanging remotely XML messages, being a syntactical way (using XML based languages) or a semantic way (using ontology based languages).
Some specific services dedicated to software systems families are also made available by the other communities. It is for example the case of Standardized Data Access Interface which is specifies formally services provided by an information server. It is also the case for Product Data Management or Product Lifecycle Management systems, with PDM enablers, OMG PLM services or OASIS PLCS PLM services. Finally, for any kind of service and component, the business logic will be described for people, by mean of natural language, in order to provide precise and explicit description of operations, for different industrial domains and area of knowledge.
Object Management Group provides a **classification of services** within the **Object Management Architecture** I consider as highly relevant for addressed purpose: low level common services are foundation of communication infrastructure, while higher level services are concerning higher logic dedicated to people. The distinction is made between horizontal services, which can be used by any kind of industrial domain and application (e.g. workflow), while vertical services are dedicated to such or such industrial domain or area of knowledge (e.g. product data management enablers in the manufacturing area). For the manufacturing domain, product data management and product lifecycle management, the more important

standards related to services are OMG's Product Data Management Enablers, OMG's Product Lifecycle Management Services and OASIS' PLCS services. Strongly connected to STEP application protocols which provide semantic allowing to type input and output of operations, they are not as mature as standards for data exchange and sharing.
OMG's frameworks are described within section 4.6.

3.2.3 - Composition of services

A service usually provides a set of operations it can performs, with for each operation inputs and outputs. But the way to sequence several services, introducing time factor, is not considered by the standards for service specifications and associated languages. We need here a way to define how the components or systems are interacting in order to reach objective of the collaboration. **Sequencing services** is usually done by defining processes, which are sharing context and data. Some distinction exists between service orchestration and service choreography. The orchestration and the choreography distinctions are based on analogies: **orchestration** refers to the central control (by the conductor) of the behavior of a distributed system (the orchestra consisting of many players), while **choreography** refers to a distributed system (the dancing team) which operates according to rules but without centralized control. More practically, a more important difference exists. An orchestration language focuses on the view of a single business participant, whereas choreography language describes peer-to-peer collaborations of multiple business participants.

Several standardized languages and generic systems exist related to process formalization. Languages can be textual formal languages or graphical languages. Languages can be workflow modeling languages, design languages or execution languages. Some ambiguity exists when defining the system controlling execution of a business process. It can be a software system or an organizational system. When several organizations are involved with each their own process execution systems, we are in the context of transversal collaboration. Very few standardized formal processes exist for several reasons. Firstly, enterprises are considering that the know-how is within the processes, and don't want to share it in order to protect their intellectual property. Secondly, too many process modeling language exist which are quite different and used for different purposes. It makes it difficult to produce reusable models that can be deployed with numerous existing and used technologies. Finally, process model description dedicated to automation will require very detailed, precise and explicit definition. Modeling processes that can be interconnected is resource consuming and difficult to interconnect with others. Most of the standardized processes are formalized using natural language, and are dedicated to people and organization, not to computers. It is in particular the case for standards dedicated to quality management, such as ISO 15928 (System Engineering), CM II (configuration management) or ISO 9001 (quality).

Standardized frameworks for process modeling are described within section 4.7.

3.2.4 - Transversal collaboration

Transversal collaboration system is mainly used for automated executed interacting processes which are being controlled by several execution pools and organizations. The perfect example is provided by ATHENA Collaborative Business Process. It can also be implanted mixing notification services (based on triggering mechanisms) and process execution engines. Currently no standard is supporting approach developed within ATHENA. But it could be envisaged as an extension of existing standards for Business Process Modeling.

3.3 - The standards and associated formal language to consider

As willing to support a holistic way the eBusiness collaboration between organizations, interconnecting their internal business processes and supporting applications, all the kinds of standards for interoperability are to be considered. It includes previously described business data interchange and composition of distributed services in order to support business processes and transversal collaboration. Referring the NEHTA's Interoperability Maturity Model, I applied it to the manufacturing community. I consider that a set of open standards have been elected by this community in order to address interoperability of PLM system. I consider these standards as legacy items that have to be considered when dealing with any innovative interoperability solution. These standards are also reflecting state of the art for interoperability and eBusiness collaboration. Recapitulation of relevant standards to consider for eBusiness collaboration and interoperability of related enterprise applications is provided by Table 8.

	ISO 10303	OMG VDA	INCOSE	OASIS	Wfmc	Configuration Management Institute
Data schema definition	STEP application protocols			PLCS DEX PLCS Resource Data Libraries		
Service specifications	SDAI for information server	• PDM enablers • PLM Services • Workflow Services		PLCS Services	Five interfaces of workflow system	
Business Processes	Application Activity Model	VDA change management process	ISO 15928	Application Activity Model		Configuration Management II

Table 8: relevant standards

Table 9 provides a table of languages used to produce standardized information model, service specifications and collaboration processes.

	ISO 10303	OMG	W3C	OASIS	Wfmc
Data schema definition language	EXPRESS Ontology	Common Data Representation (part of Internet Inter Orb Protocol)	• XML DTD • XML Schema • Ontology in OWL		
Service language specification	Languages for which a binding is provided for Standard Data Access Interface	• Interface Definition Language • UML- Class and Component diagrams • UML use cases		Web Service Definition Language	
Service Composition language		• Business Process Modeling Notation • UML activity and sequence diagrams		Business Process Execution Language	XML process definition language (for workflow systems)
Transversal collaboration language		Business Process Modeling Notation (partially)			

Table 9: Languages to consider for formalization of relevant standards for interoperability

Most of the identified standards are associated with generic interoperability frameworks aiming to support the different actors to achieve operational interoperability as defined by System Of System Interoperability (SOSI) model. As pointed out by NEHTA's Interoperability Maturity Model, interoperability frameworks are important to achieve

maturity level 2. Chapter 4 describes the different interoperability frameworks associated to the relevant identified standards.

Chapter 4 – Interoperability frameworks

As pointed out by NEHTA, reaching second level of maturity for eBusiness interoperability required usage of interoperability frameworks. I have presented within the Chapter 3 a classification for interoperability standard, with associated example of standards which are relevant for the industrial context. Most of these open standards are provided with an associated framework. In this chapter, I firstly propose a definition of framework. I then describe the more relevant frameworks for interoperability which are relevant for our purpose, associated with standards of the previous chapter or associated to best practices I identified. The aim is to reflect current state of the art and of the practice. I then present state of the art concerning mixing and federating existing approach, standards and frameworks.

4.1 - Definition of framework

According Wikipedia, a framework is "a basic conceptual structure used to solve complex issues". Widely used in software context, a framework provides an abstraction in which common code or component providing generic functionality can be selectively overridden or specialized in order to provide specific functionality. They can be considered as reusable abstraction wrapped in a well defined interface, with some principles making them different from usual application or application programming libraries. Firstly the overall flow of control is dictated by the framework. Secondly, the framework proposes a default behavior. Thirdly, a framework can be extended. Finally, the code of the framework is not allowed to be modified.
A framework aims to devote the time to meet system requirements rather than dealing with the more standard low-level details of providing working system, thereby reducing overall development time. From architectural point of view, a framework will consist of frozen spots and hot spots. Frozen spots provide the overall architecture of a system, including basic components and their relationships. They remain unchanged for any instantiation of the framework. Hot spots represent those parts specific functionality to the system developed using the framework.

4.2 - Interoperability frameworks

An interoperability framework is a basic conceptual structure used to solve complex issues related to interoperability. It provides an abstraction in which common code or component providing generic functionality can be selectively overridden or specialized in order to provide specific functionality. It could be for example information server, application server, workflow system, service bus and enterprise portal. They are reusable abstractions wrapped in well defined interfaces, usually available as standardized formal specifications.
The overall flow of control is dictated by the standardized framework, not by the implementation of the framework. The standardized framework proposes a default behavior. Interoperability framework can be extended, in order to support specificity of local and private interconnected systems. And finally the formal code of the interoperability framework, being schemas or interfaces, is not allowed to be modified. Interoperability frameworks aims to devote the time to meet specific collaboration requirements rather than dealing with the

more standard low-level details of collaborating, thereby reducing overall development time of interaction between the enterprise applications.
From architectural point of view, interoperability frameworks will consist of frozen spots and hot spots. Frozen spots provide the overall architecture of a system, including basic components and their relationships. They remain unchanged for any instantiation of the framework. It is for example the architecture of reference for a workflow system according the Workflow Management Coalition, with associated formal interface specifications. Hot spots represent part developed by solution providers which can extend the specification in order to provide higher value to their product, or added for the users expecting further capabilities.
The more important interoperability frameworks to consider as part of the state of the art, but also as relevant component solution for interoperability for technical enterprise applications are following.

4.3 - ISO STEP Framework

IS0 10303 standard, called STEP (STandard for the Exchange, Sharing and Retention of Product model data), is a set of standards dealing with exchange, sharing and long term retention of data describing a (manufactured) product, between heterogeneous software (Computer Aided Design, Calculation or Manufacturing tools for authoring, management or collaboration) and partners (Customers, Clients, Primes, Supplier, Sub-contractor, etc).

Objective is to enable exchange all along the lifecycle of the product (conception, design, fabrication, assembly, test, delivery, support/exploitation, recycling), as illustrated in Figure 12.

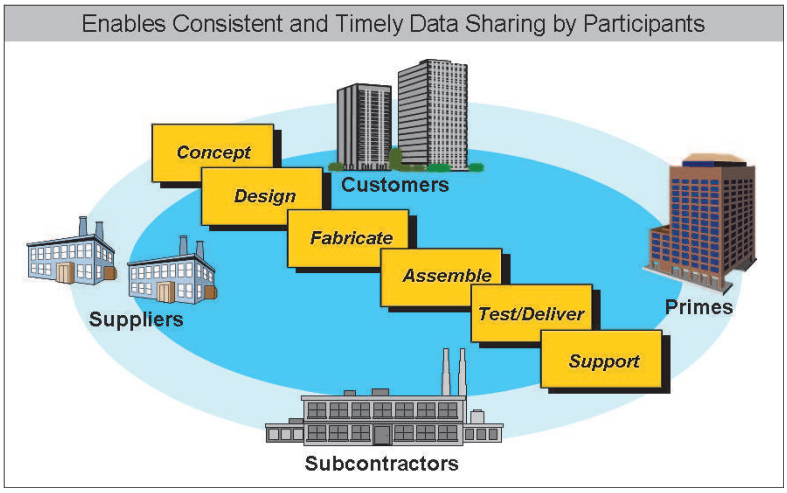


Figure 12: STEP Objectives

Different disciplines are concerned (mechanical design, electrical design, system design, etc) but also different industrial sector (Oil and Gaz, Aeronautic, Automative, Furnitures, Building, Shipping…).

Figure 13 provides the actual list of all application protocols which have been defined by the manufacturing community.

I 201 Explicit draughting [ATS 301 = X]
I 202 Associative draughting [C]
I 203 Configuration controlled 3D design (c2=I,a1=I)[X]
I 203 Configuration controlled 3D design (TS1=I)
I 204 Mechanical design using boundary rep [I]
X 205 Mechanical design using surface rep [W]
X 206 Mechanical design using wireframe [X]
I 207 Sheet metal die planning and design [c1=I]
X 208 Life-cycle product change process [X]
I 209 Composite & metal structural anal & related design[X]
I 210 Electronic assy, interconnection & packaging design [X]
W 210 e2 Electronic assy, interconnection & packaging design
X 211 Electronic P-C assy: test, diag, & remanuf[X]
I 212 Electrotechnical design and installation [X]
X 213 Num control (NC) process plans for mach'd parts [X]
I 214 e2 Core data for automotive mech design processes [X]
A 214 e3 Core data for automotive mech design processes
I 215 Ship arrangement [X]
I 216 Ship moulded forms [X]
X 217 Ship piping [X]
I 218 Ship structures [W]
C 219 Dimension inspection [X]
O 220 Proc. plg, mfg, assy of layered electrical products [X]
E 221 Functional data & their schem rep for process plant [X]
X 222 Design-manuf for composite structures [X]
C 223 Exch of design & mfg product info for cast parts [X]
@ 224 e3 Mech pdt def for p. plg using mach'n'g feat e3 (e2=I) [I,W]
I 225 Building elements using explicit shape rep [C]
X 226 Ship mechanical systems[X]
I 227 Plant spatial configuration(e2=E) [X]
X 228 Building services: HVAC [X]
A 229 Exchange of product info for forged parts[X]
X 230 Building structural frame: steelwork [X]
X 231 Process-engineering data [X]
I 232 Technical data packaging: core info & exch [I]
W 233 e2 Systems engineering data repr
X 234 Ship operational logs, records, and messages[X]
C 235 Materials info for des and verif of products [X]
E 236 Furniture product and project data[W]
X 237 Computational Fluid Dynamics
E 238 Application interpreted model for computer numeric controllers
I 239 Product life cycle support
I 240 Process plans for machined products

Legend: Part Status (E, F, I safe to implement)
0=O=**Preliminary Stage** (Proposal-->appr for NP ballot)
10=**A =Proposal Stage** (NP ballot circ-->NP approval)
20=**W=Preparatory Stage** (Wkg Draft devel.-->CD registration)
30=C =**Committee Stage** (CD circulation-->DIS registration)
40=E =**Enquiry Stage** (DIS circ.-->FDIS registration)
50=F =**Approval Stage** (FDIS circ-->Int'l Std registration)
@=**At ISO, approved for publication** (ISO status 40.95 or 50.99)
60=I =**Publication Stage** (Int'l Std published)
98=**X=Project withdrawn**

Figure 13: list of STEP application protocols

The elements of the standard are the set of application protocols, which are domain specific, with set of common integrated resources (domain independent) and implementation methods, as described by Figure 14.

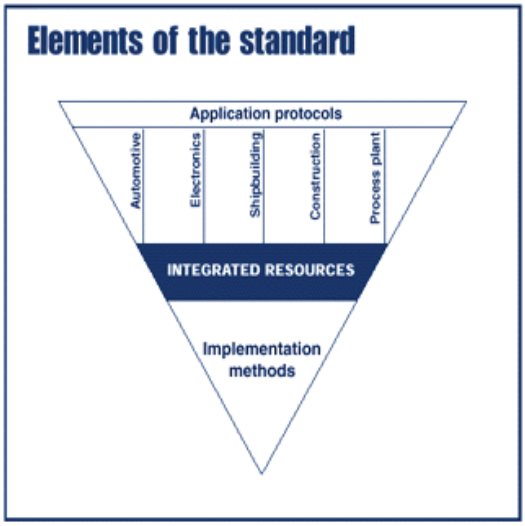


Figure 14: Elements of the STEP standard

As described by Figure 15, these elements constitute numbered parts of the standard. Application protocols and associated abstract test suites are numbered part "2xx". The common resources include application modules, integrated application resources (part "4x" and "5x"), application interpreted constructs (part "5xx"). Implementation methods include exchange, sharing and binding methods. The description methods (part "1x") include overview and fundamental principles, a language to describe schemas (EXPRESS), a language to describe data (EXPRESS-I), a language to formalize mapping that can be executed (EXPRESS-X) plus architecture and methodology.

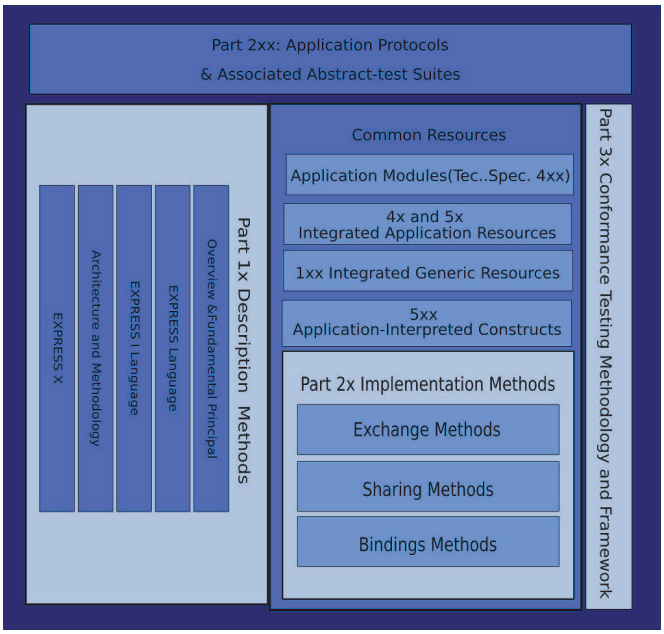


Figure 15: Parts of the STEP standard

So STEP constitutes a set of standards of all types, i.e. definitions, specifications, tests and practices. An application protocol is a document (paper based, digital document or XML document) providing firstly text based definition of the information model for people, from users' requirements analysis. The information is categorized according a functional classification (Unit of functionalities or UoF) and UoF are categorized according a software tool classification (Conformance Classes). An application protocol also provides context description based on a generic and not standardized process (Application Activity Model as illustrated by Figure 16) defined using IDEF 0 graphical language for process description.

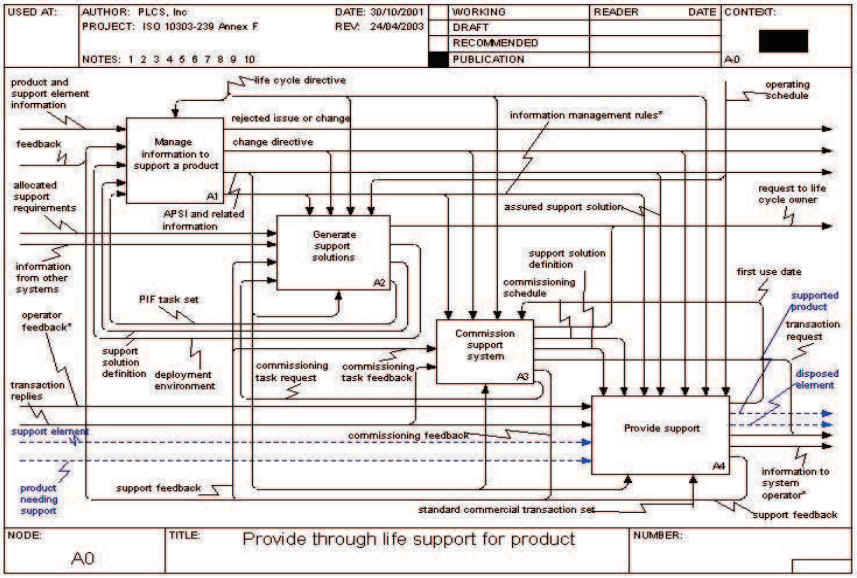


Figure 16: example of Application Activity Model

Then a formal model of the information is defined with the EXPRESS language, the Application Reference Model (ARM) and a translation (mapping) on the basis of the integrated resources in order to obtain a computable neutral schema which is domain independent, the Application Integrated Model (AIM). The idea is to avoid different definitions of the same thing within different application protocols. The normalized model for

exchange and sharing between applications is the Application Integrated Model. The full process, with the different components of an application protocol, is represented by Figure 17

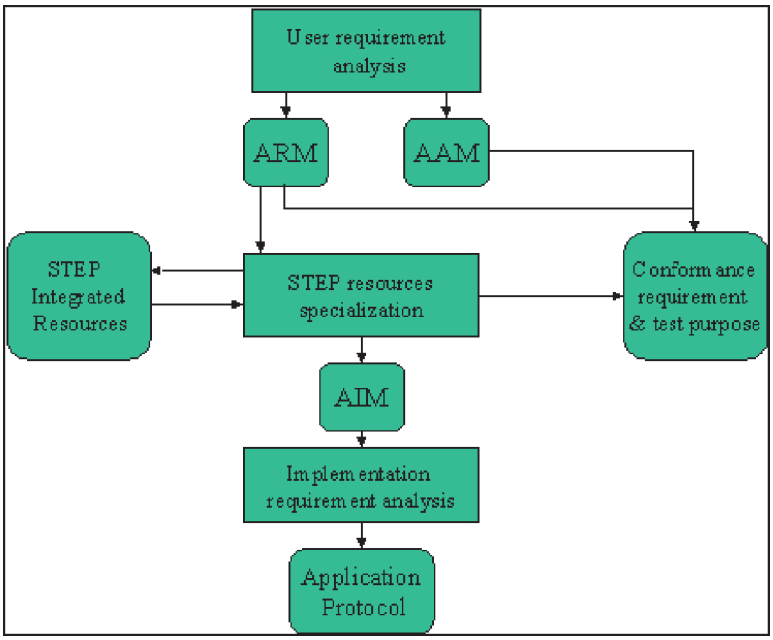


Figure 17: The different phases of the creation of an application protocol

Additionally, graphical representations using language EXPRESS-G is provided for people, which is not normative as graphical languages are not computable. Figure 18 is a screenshot of the content of an application protocol, providing navigable list of terms, corresponding EXPRESS schema and hyperlink to graphical representation in EXPRESS-G. Such a figure is important as it highlights that application protocols are document defined by people for people first.

Figure 18: Application protocol extract as an eDocument for people

On the basis of the Application Integrated Model schema, implementation methods are provided allowing to exchange data (Part 21 file format) or to share data (formal service for data sharing through definition of a Standardized Data Access Interface).

Some binding methods are provided to make the link with Information and Communication Technological standards for data exchange, application specification and software implementation. STEP standards aims to be business process independent (a business process can't be shared as it is the know how of the enterprise) and Information and Communication Technology (ICT) independent (the aims is not to replace usual solution providers and to develop software). For test conformance, some methods are provided allowing testing conformance on the basis of the Conformance Classes defined within the Application Protocols.

In order to fasten time required for developing an Application Protocol, a subset of Application Protocols are modular XML documents, allowing sharing document components between several documents and to ensure a better coherency between Application Protocols covering all the lifecycle of a product, from requirement engineering (AP233 "System Engineering") to customer support (AP239: "Product Lifecycle Customer Support") through design (AP203: "Configuration Controlled 3D Design").

		Product Life Cycle					
Analysis per main categories of information	**STEP standards**	As Specified	As defined (Simulation & Analysis)	As Design	As Planned	As Prepared	As Maintained / As Flying
PDM	AP 214		AP214	**AP214**	**AP214**	**AP214**	
	Modular and consistent APs	**AP233 + AP239**	**AP233 + AP209**	AP203 ed2 (AP239)	AP203 ed2 (AP239)	AP203 ed2 / **AP 239**	**AP239**
3D with GD&T	AP 214						
	AP 203 ed2						
Target of use in operation	**Scope of AP 203 ed2 (*)**	**2008/2009**					**2008/2009**
	Scope of AP 214			**NOW**			
	(*): Integrated and fully consistent with STEP AP 233 and STEP AP 239						
	The PDM part of STEP AP 203 ed2 is included in STEP AP209 ed2, STEP AP233 and STEP AP239						
AP 233: Systems Engineering						Recommended	
AP 239: Product Life Cycle Support						Alternative solution	
AP 209: Composite and metal structural analysis and design							
AP 203: Configured Controlled 3D Design					GD&T: Geometric Dimension & Tolerancing		
AP 214: Core data for automotive mechanical design processes							

Figure 19: Positioning of different sets of application protocols (AIRBUS, SEINE)

Maturity assessment of application protocols has been undertaken by European Aerospace and Defense community in order to cover the entire lifecycle of a product. Figure 19 is positioning several application protocols coming from United States and Defense and European Automotive industry. Communities developing STEP Application Protocols identified that providing only neutral information schema was not sufficient in order to insure interoperability between commercial software products. So implementer forums have been organized, allowing providing recommended practices based on lessons learnt from implementation. The technologies developed by the STEP community are specific to this community, and are usual Information and Communication Technologies independent, in order to avoid continuous and quick evolution of the model of reference. Some bindings are provided to be able to standardize the way to use STEP with the most used technologies within eBusiness community.

As graphical languages can't be interpreted by computers, it was decided to make the EXPRESS schemas normative, and EXPRESS-G graphical representation indicative. A schema in EXPRESS describes a set of entities with attributes, which can be simple types or complex build types. In addition, inheritance is used in order simplifying descriptions. EXPRESS language also includes constructs allowing formalizing rules and constraints. The

language modeling concepts are close to those allowing describing conceptual models and relational databases. They are also close to those defined with XML schema specification. EXPRESS-G allows describing a schema as a semantic graph. Each attribute is represented as an arc relating a node representing the type of the attribute and related to the entity the attribute belong to. The mapping language that can be used for transformation, EXPRESS-X, extends EXPRESS, and allows to formalized maps or views. It is similar to XML transformation language. Standard Data Access Interface provides the set of operations provided by an information server, allowing managing a STEP models repository. Bindings are provided in C and Java. Bindings to XML and UML are provided, allowing to take advantage of Service Oriented Architecture framework based on World Wide Web Consortium and Model Driven Architecture frameworks based on Object Management Group' s specifications. Links are also being established with the semantic web through OASIS Product Lifecycle Management program, establishing links with ISO 15926 (Reference Data and Exchange Lifecycle Integration).

It is important to point out that ISO STEP framework is decoupled from software engineering, as it does not standardized any software component or development process. It consequently provides an efficient support neither for software engineers nor for web infrastructure. Such support is provided by frameworks proposed by the Object Management Group and by the World Wide Web consortium, which are described in the next sections.

4.5 - World Wide Web Consortium framework

In order to provide Web infrastructure, the World Wide Web Consortium (W3C) has been providing numerous interrelated recommendations. Most of them are founded on top of Universal Resource Identifiers (URIs), Hyper Text Protocol (HTTP), eXtended Markup Language (XML) and Resource Description Framework (RDF). W3C recommendations support pursuits in five areas: themes of accessibility, internationalization, device independence, mobile access and quality assurance pervade W3C technologies. A representation of the complete stack of W3C recommendations is provided by Figure 20.

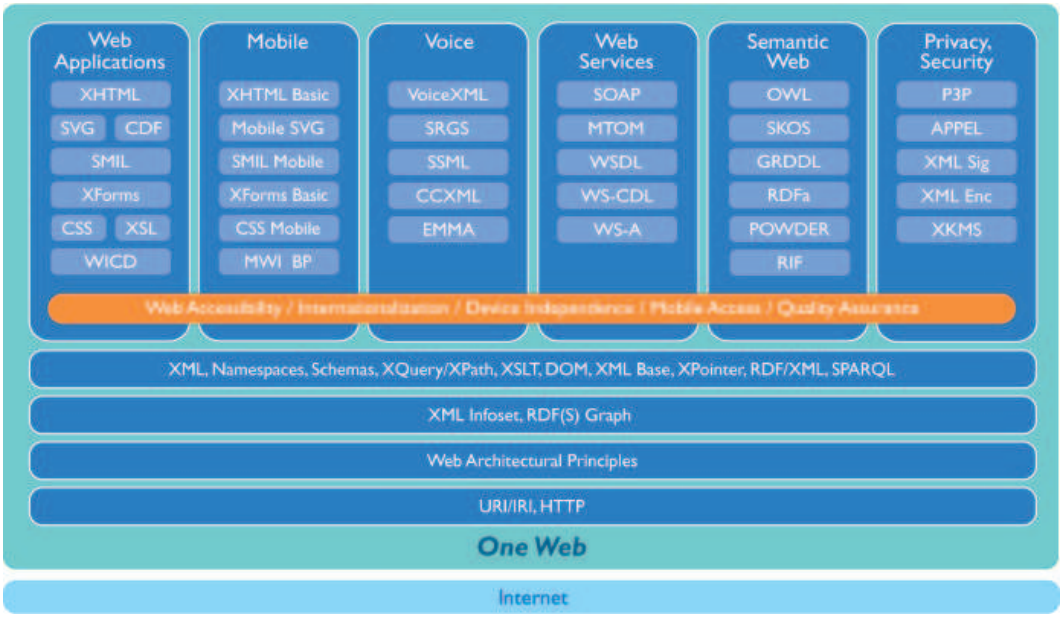


Figure 20: World Wide Web Consortium recommendations (W3C)

Through usage of numerous implementations of these recommendations, Internet is being becoming an open interoperable technical infrastructure. One issue comes from the fact that the recommendations are formalized at syntactic level, and not at semantic level. It is why W3C is being developing also semantic WEB recommendations on top of XML and RDF, in order to obtain a more "intelligent" WEB (with semantic graphs of heterogeneous resources and reasoning engines).

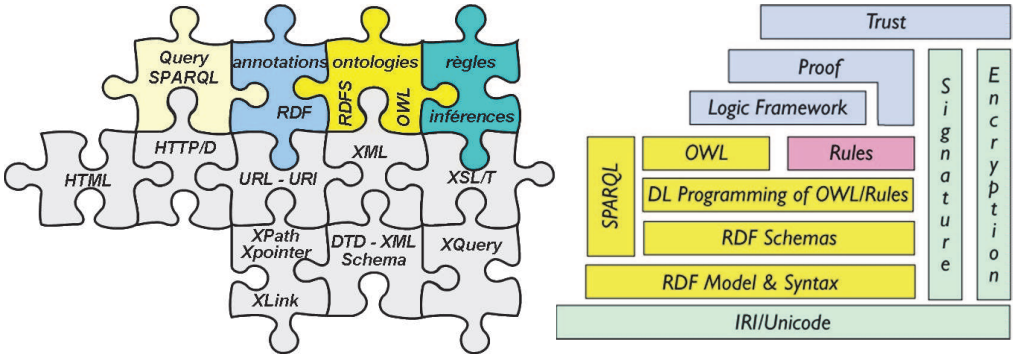
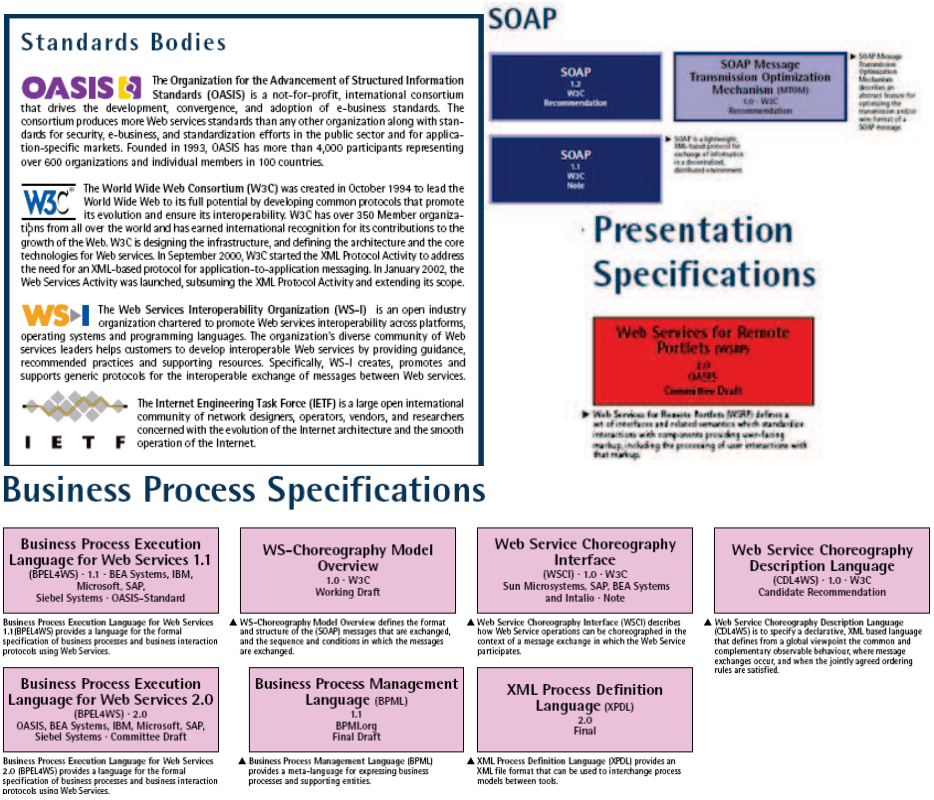


Figure 21: Semantic W3C standards and semantic stack (Infamous layer cake)

WEB services and WEB applications recommendations are completed by other specifications coming from OASIS, WS-I and IETF, such as WEB Service Specification Language (WSDL) for remote service specifications, Business Process Execution Language for orchestration of web services, WEB services choreography specifications, etc.
Figure 22 provides an extract of specification completing W3C specification in order to obtain secured service oriented WEB infrastructure.



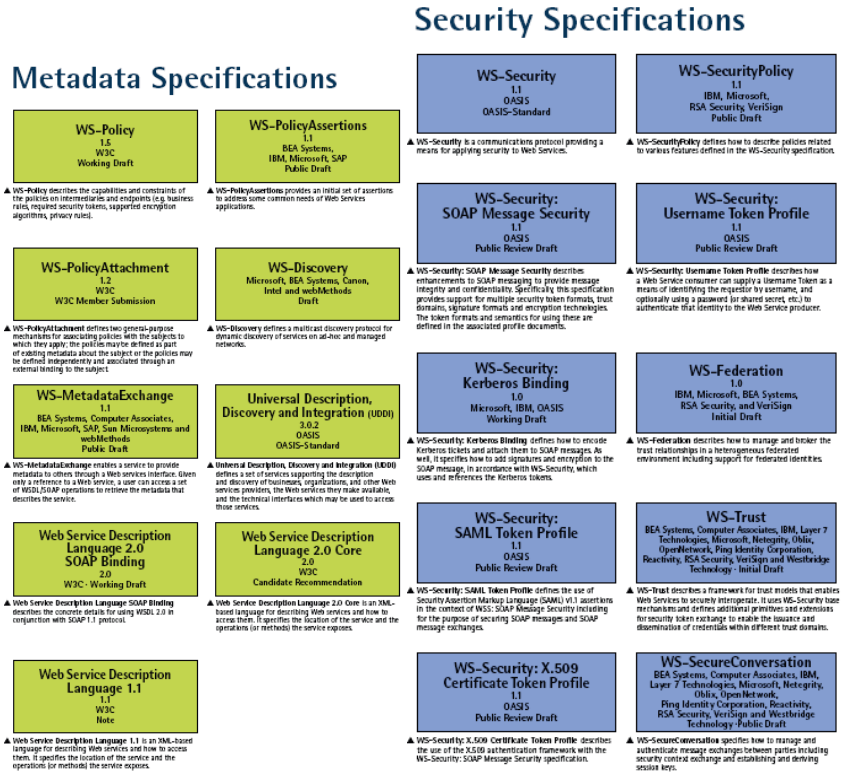


Figure 22: OASIS, IETD and WS-I complementary specifications for Service Oriented infrastructure

4.6 - Object Management Group Framework

Object Management Group (OMG) is a consortium, originally aimed at setting standards for distributed object-oriented systems such as Common Object Request Architecture Broker (CORBA) middleware. OMG is now focused on modeling of programs, systems and business processes. It produces model-based specifications, based on Universal Modeling Language, and associated mechanisms aiming to enforce true interoperability.

As stated by D.C. Schmidt[16], "the Common Object Request Broker Architecture (CORBA) is a standard defined by Object Management Group that enables software components written in multiple computer languages and running on multiple computers to work together. CORBA automates many common network programming tasks such as object registration, location and activation, operation dispatching, message translation between components…"

[16] Doublas C.Schmidt is Formerly Associate Professor of Computer Science, Washington University, and most of the CORBA and CCM description, including figures, is extracted from his web site at http://www.cs.wustl.edu/~schmidt/

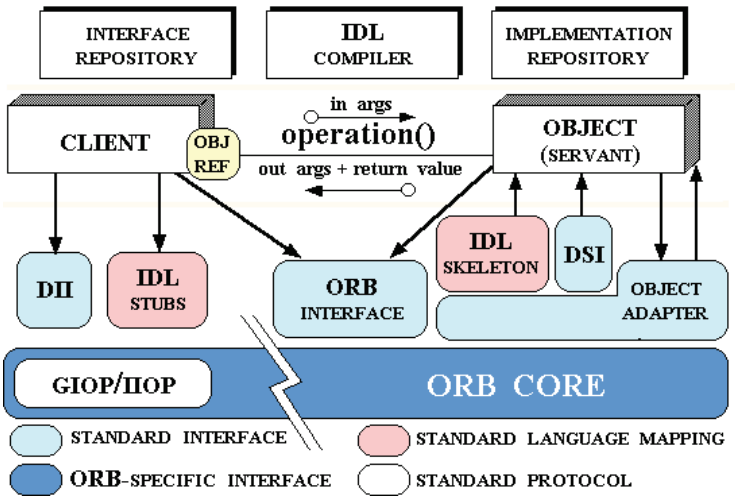


Figure 23: CORBA framework architecture

The primary components of CORBA architecture represented in Figure 23 are:

- Objects: programming entities that consists of an identify, an interface and an implementation, know as a "Servant"
- Servants: implementation programming language entity that defines the operations that support a CORBA IDL (Interface Definition Language) interface – servants can be written in a variety of languages.
- Clients: program entity that invokes an operation on an object implementation. Accessing the service of a remote object should be transparent to the caller and a simple as calling a method on an object
- Object Request Broker (ORB): provides a mechanism for transparently communicating client requests to target object implementations. The distributed programming is simplified by decoupling client from details of the method invocations. The client request appears as local procedure calls. When a client invokes an operation, the Object Request Broker is responsible for finding the object implementation, transparently activating it if necessary, delivering the request to the object, and returning any response to the caller.
- Object Request Broker interface: Object Request Broker is a logical entity that may be implemented in various ways. To decouple applications from implementation details, abstract interface of the Object Request Broker is defined. This interface provides various helper functions such as converting object references to strings and vice-versa, creating argument lists for request made through the dynamic invocation interface
- CORBA IDL (Interface Definition Language) stubs and skeletons: the "glue" between the client and server applications respectively and the Object Request Broker. The transformation between CORBA IDL definition and the target programming language is automated by a CORBA IDL compiler, allowing reducing potential inconsistencies between client stubs and server skeletons and increasing opportunities for automated compiler optimization. Note: with modern application servers based on CORBA Component Model implementations (e.g. Enterprise Java Beans), the interface is directly generated in IDL (Interface Definition Language) from component description at deployment time as the helpers; it is just required to tag the object implementation description with a pre-defined annotation.)
- Dynamic Invocation Interface (DII): it allows a client to directly access the underlying request mechanism provided by an ORB. Applications use the Dynamic Invocation Interface to dynamically issue request to objects without requiring IDL (Interface Definition Language) interface specific stubs to be linked in. Unlike IDL (Interface

Definition Language) stubs(which only allow RPC-style requests), the Dynamic Invocation Interface also allows clients to make non-blocking deferred synchronous (separate send and receive operations) and one-way (send only call).

- Dynamic Skeleton Interface (DSI): server side's analogue to the client side's DII. The Dynamic Skeleton Interface allows an Object Request Broker to deliver requests to an object implementation that does not have compile-time knowledge of the type of the object it is implementing. The client making the request has no idea whether the implementation is using the type-specific IDL (Interface Definition Language) skeletons or is using dynamic skeletons.
- Object Adapter: assists the Object Request Broker with delivering requests to the object and with activating the object. More importantly, an object adapter associates object implementations with the Object Request Broker. Object adapters can be specialized to provide support for certain object implementations styles.

On top of CORBA, The Object Management Architecture Reference Model forms a conceptual roadmap for assembling technologies that satisfies the Object Management Group's Technical objectives.

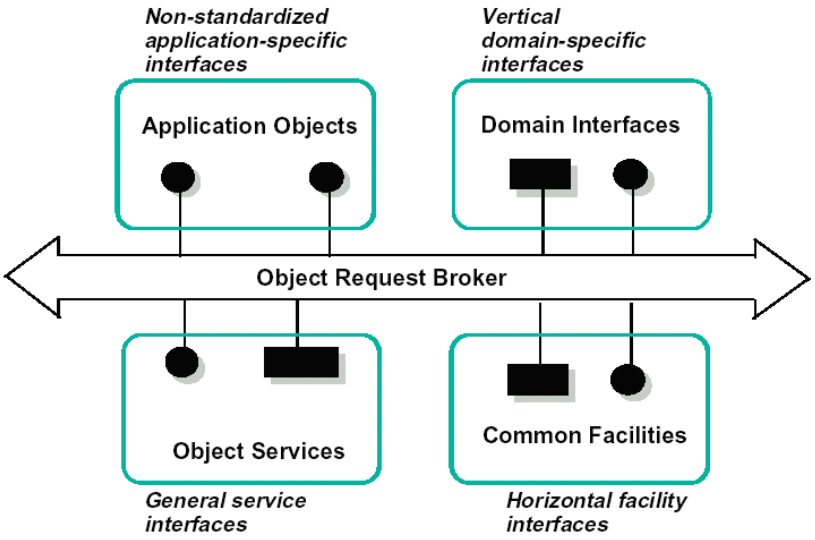


Figure 24: Object Management Architecture

Object Request Broker is the central component and is the key for interoperability. Common Object Services is a collection of services with object interfaces that provide basic functions for realizing and maintaining objects. Table 10 provides the list of general services defined within Object Management Architecture with available specifications from Object Management Group, also called Common Object Services.

Service Name	Service Description
Naming service	It provides ability to bind a name to an object relative to a naming context (object that contains a set of name bindings in which each name is unique), navigate graphs of distributed naming contexts…
Event service	It provides capabilities that can be configured together in a very flexible way, to deal with asynchronous events, event, notification and their push or pull delivery.
Lifecycle service	It conventions for creating, deleting, copying and moving objects.
Persistent Object Service	It provides a set of common interfaces to the mechanisms used for retaining and managing the persistent state of the objects
Transaction service	It supports multiple transaction models, included flat and nested, and support interoperability between different programming model
Concurrency	It enables multiple clients to coordinate their access to shared resources.

Control service	
Relationship service	It allows entities and relationships to be explicitly represented. Entities are represented as CORBA objects. The service defines two new kinds of objects: relationships and roles. A role represents a CORBA object in specific attributes and operations. Type and cardinality constraints can be expressed and checked.
Externalization service	Records the object state in a stream of data (in memory, on a disk file, across the network, and so forth) and then be internalized into a new object in the same or a different process.
Query service	Allows users and objects to invoke queries on collections or other objects. Queries are declarative statements with predicates, and include the ability to specify values of attributes. The query service provides architecture for a nested federated service that can coordinate multiple, nested query evaluators.
Licensing service	It provides mechanism for producers to control the user of their intellectual property.
Property service	It provides ability to dynamically associate named values objects outside the static IDL type systems
Time service	It enables to obtain current time together with an error estimate associated with it, to ascertains the order in which events occurred, generate time based events based on timers and alarms, compute interval between two events …
Security service	It deals with identification and authentication, authorization and access control, security auditing, security of communication , non repudiation and administration
Object trader service	It provides a matchmaking service for objects with partitioned and federated trading domain
Object Collection service	It provides service for sets, queues, lists binary and tress.

Table 10: Common Object Services specified by OMG

Common facilities are a collection of classes and objects that provide general purpose capabilities useful in many applications. It is subdivided in horizontal domain facilities (services that can be used by any kind of business domain, such as workflow capabilities, user machine interface, etc) and vertical domain facilities (services that can be used by a specific domain e.g. manufacturing, telecom, health, etc). Product Data Management Enablers or PLM services are example of specifications being part of vertical domain facilities.

Due to several limitations of the second version of CORBA, the application implementations were most of the time brittle, non scalable, hard to adapt and maintain, leading to increase of time to market. The proposed solution was the CORBA Component Model, which propose to use a standardized “virtual boundary”, called “containers”, around application component implementations that interact only via well defined interfaces. It is illustrated by Figure 25.

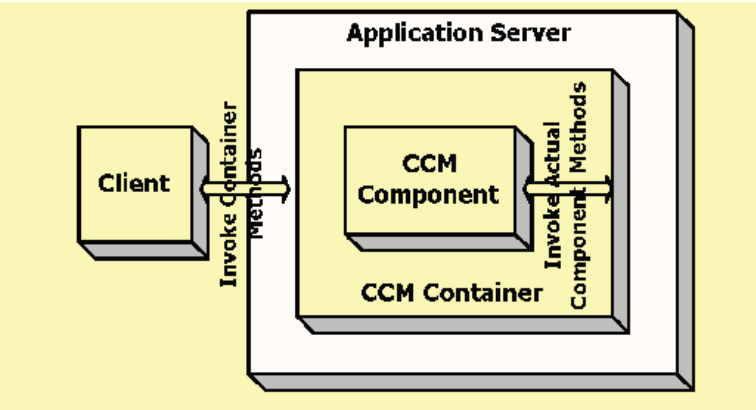


Figure 25: Application server based on CORBA Component Model

Standard container mechanism are defined which allow to execute components in generic components servers, and allows to specify a reusable standardized architecture for configuration and deployment of components throughout a distributed system.
The components encapsulate application "business" logic. Components interact via ports with provided interfaces as "facets", required connection points as "receptacle", event sink, sources, and attributes.
It is important to point out that CORBA Component Model is still dealing with communication between software, and that the user front end (interface human machine) is not considered.

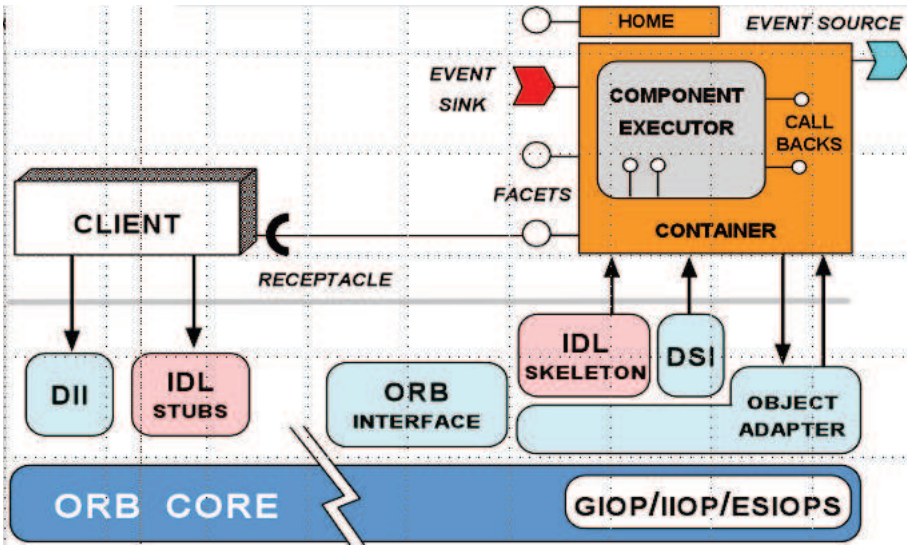


Figure 26: Application server communicating by means of middleware bus

The containers provide execution environment for components with common operating requirements. Components/containers can also communicate via a middleware bus and reuse common middleware services, as illustrated by Figure 26.

Finally, as Object Management Group's specifications were not universally adopted, and as several other middleware were coexisting with CORBA, Object Management Group develop a model driven approach, on top of his Universal Modeling Language (UML), producing what is so called Model Driven Architecture (MDA).
The three primary goals of MDA are **portability, interoperability and reusability**.
As defined by OMG's MDA Guidelines, the Model-Driven Architecture starts with the well-known and long established idea of separating the specification of the operation of a system from the details of the way that system uses the capabilities of its platform. MDA provides an approach for. It enables tools to be provided for specifying a system independently of the platform that supports it, specifying platforms, choosing a particular platform for the system, and transforming the system specification into one for a particular platform.
With MDA, the applicative components of a system are defined using a platform independent model (PIM) defining a functional model, using an appropriate domain specific language (coming from example from Manufacturing, Finance, etc). Then, given a model defining the targeted execution platform(s) (Platform Definition Model), such as CORBA, .net, application server, relational database, etc… providing pervasive services, the PIM is transformed to one or more platform-specific models (PSMs) that can the used to generate set of code, according one or several languages, that can be deployed on targeted platforms in order to obtain to enact and to instantiate expected service and process models.

MDA is related to multiple other OMG' s specifications. In particular, modeling is done using Unified Modeling Language (UML). Meta-modeling based transformation and serialization are enabled by Meta-Object Facilities (MOF) and XML Metadata Interchange (XMI), as illustrated by Figure 27.

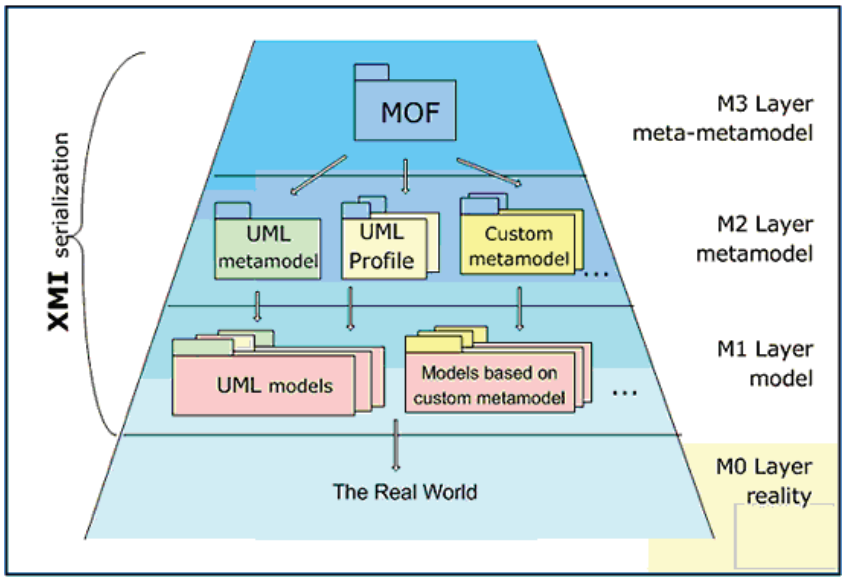


Figure 27: MDA modeling architecture

4.7 - Framework for Business Process and workflow systems

Very numerous standards and frameworks exist which are dealing with business process modeling and execution. It is important to state that "business process" and "workflow" terms are overloaded, and use for very different kind of models, usage and systems. As very various needs exist, it is very difficult to have a single standard. It is for example reflected by the ISO standard called Process Specification Language, which is providing core ontology and then several specializations according context of usage.

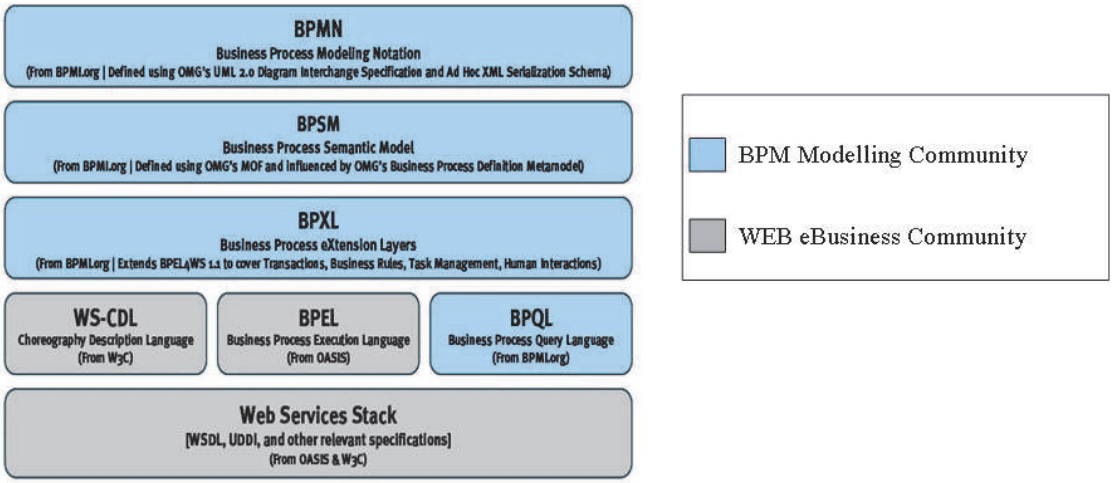


Figure 28: BPMI.org classification of Business Process related standards

In addition, numerous standardization organizations are competing in order to produce the leading open standard within the area of modeling executable business processes. The

BPMI.org community provided a positioning of the different standards without considering Workflow Management Coalition's standards Figure 28, while Workflow Management Coalition community was proposing a positioning of the standards from definition to execution languages, and distinguishing internal models and eBusiness models. As a result, the Business Process Modeling landscape is more and more complex, as shown by Figure 29.

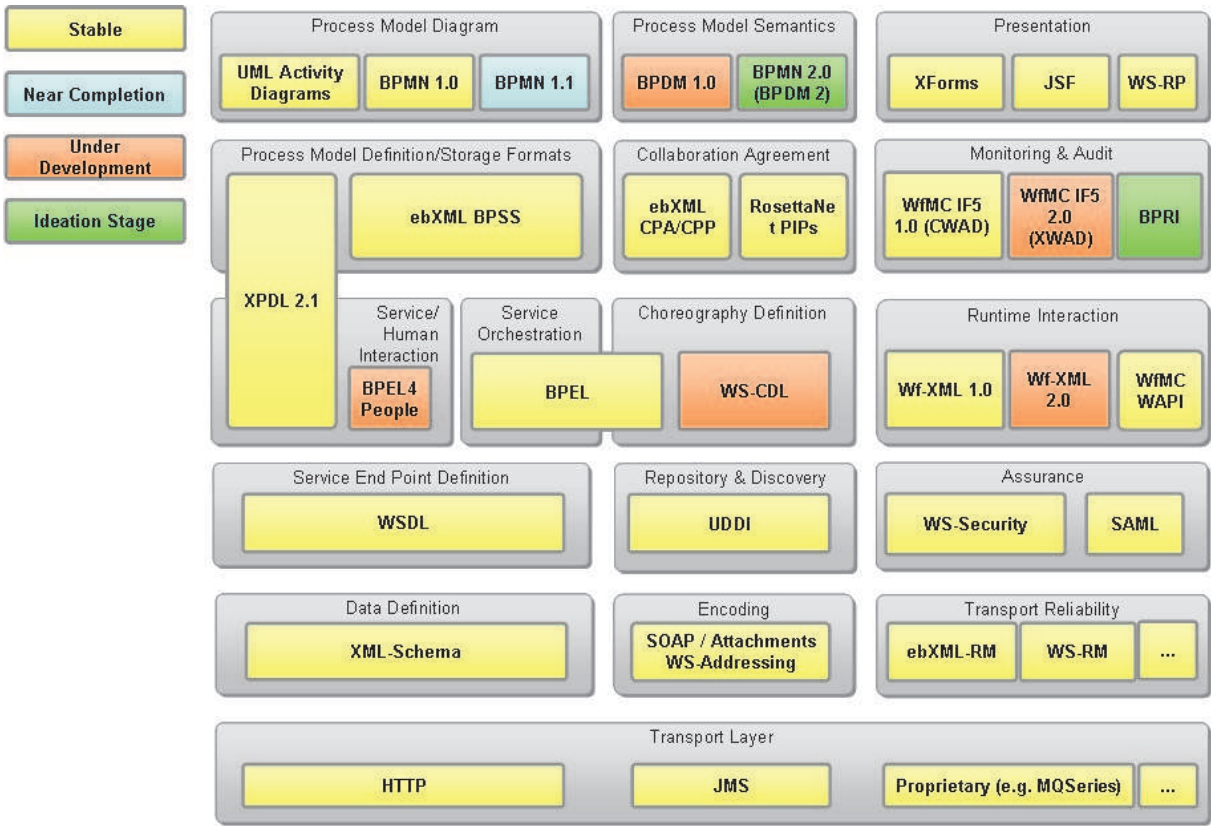


Figure 29: Business Process landscape (Swenson 2008)

The Workflow Management Coalition, founded in August 1993, is a non-profit, international organization of workflow vendors, users, analysts and university/research groups.
The Coalition's mission is to promote and develop the use of workflow through the establishment of standards for software terminology, interoperability and connectivity between workflow products. It is achieved through definition of workflow terminology, interoperability and connectivity standards, conformance requirements.

Wfmc's working groups are loosely structured around the "Workflow Reference Model" which provides the framework for the Coalition's standards program. The Reference Model identifies the common characteristics of workflow systems and defines five discrete functional interfaces through which a workflow management system interacts with its environment - users, computer tools and applications, other software services, etc".

The architecture of reference provides six main components of a workflow system, and identifies the five types of interfaces (process definition interface, administration and monitoring interface, interoperability interface, client application interface and invoked application interface).

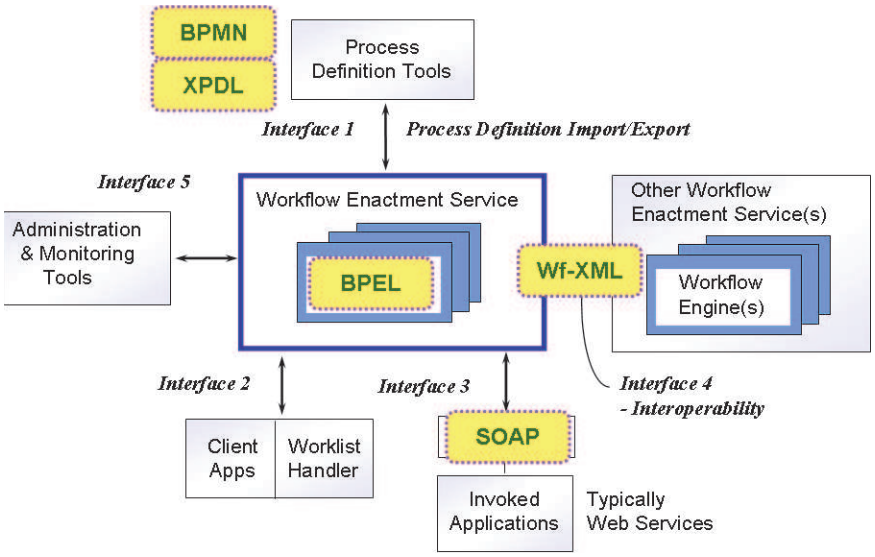


Figure 30: Wfmc Workflow architecture of reference with mapped BPM standards (Swenson 2007)

Key concepts for a workflow system are Workflow Process Model definition, Workflow Process Model Instances (created through enactment services on the basis of available Workflow Process Models) and work list (the list of work items published or notified by the enactment service that each workflow participants have to perform in order to complete the workflow model instance).

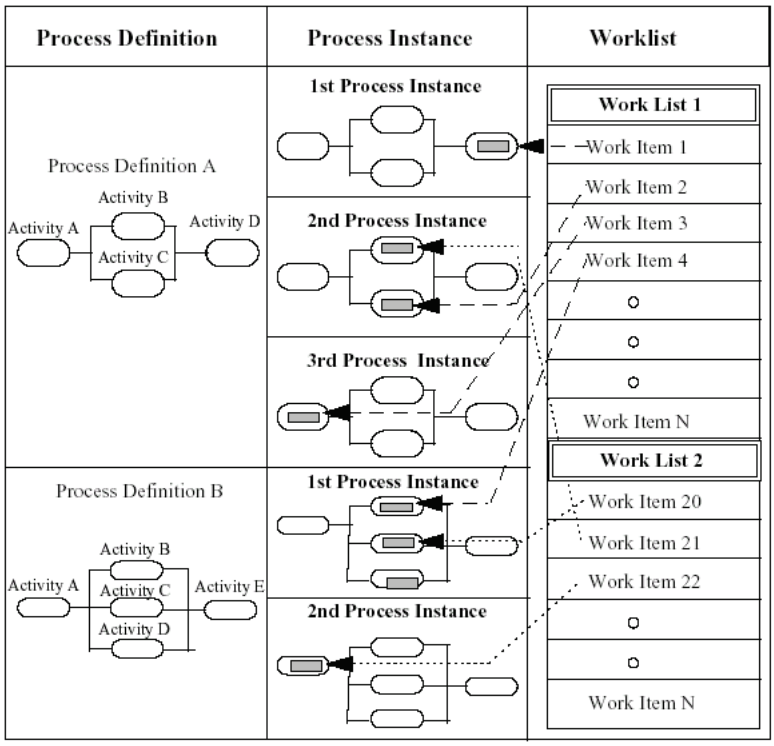


Figure 31: workflow concepts (from Wfmc Standards)

The workflow model definition is done using a process definition tool, and the model is then deployed (and parameterized) within a workflow engine, which is one component implementing enactment service. When the workflow process model is enacted, it is possible for authorized users to launch instances of the workflow process model. The workflow engine dispatches then actions between workflow participants, being persons (through worklist

handlers) or applications, according logic of the workflow process model as illustrated by Figure 31.
The process definition interface is realized through usage of a standardized XML language allowing to define Workflow Process Model and to interchange it between modeling tools and engines, XML Process Definition Language.
It is also realize through an API allowing modeling tool to interrogate an engine in order to obtain description of deployed workflow model (Wf-XML). The invoke application interface is based on WAPI specification, which provides half-gap solution: it allows adaptation for several technologies, and independence of the workflow model definition.
No graphical language exists from Wfmc in order to describe workflow process model. Last years, the Business Process Modeling Notation proposed by BPMI.org and now encompassed within OMG, was aligned with last version of XML Process Definition Language specifications. It is important to point out that the Business Process Modeling Notation does not allows only workflow process modeling, but can be used for any kind of Business Process Modeling. It means that a process modeled with the Business Process Modeling Notation is not necessarily a suitable workflow process model, according Wfmc specifications.
Since version 2 of the XML Process DL, an alignment of XML Process Definition Language exists with Business Process Modeling. XPDL was only addressing serialization of process models, and Wfmc decided to use Business Process Modeling Notation as graphical representation of the workflow process models in order not to reinvent one. It is the reason why Wfmc produces a roadmap Figure 32, which allows governing coherent evolution of these standardized specifications. It highlights emerging interdependency between several organizations in order to address eBusiness interoperability, but also emerging complexity. Numerous languages and specifications are to be known, and to be managed in configuration.

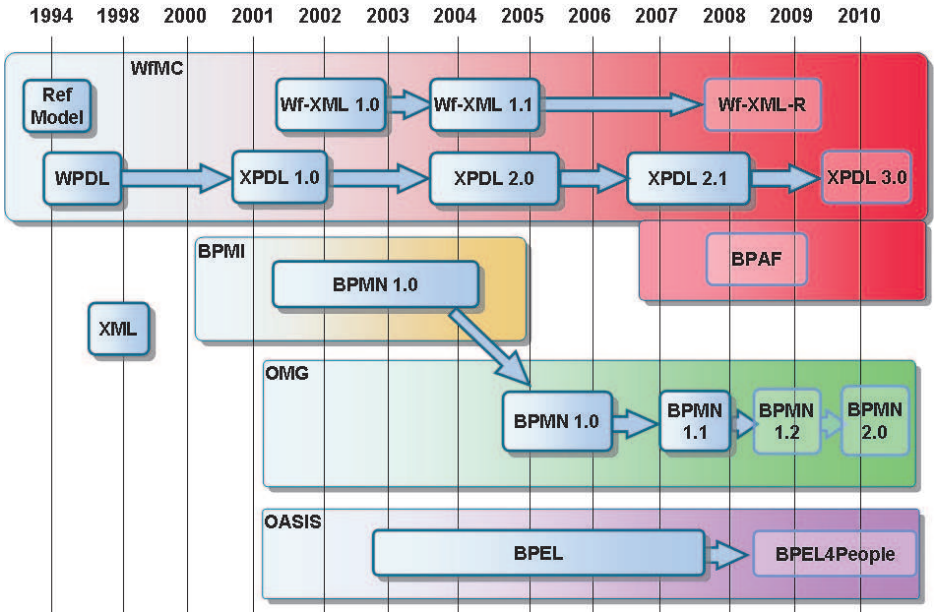


Figure 32: Wfmc roadmap for coherency between different standards (Swenson 2008)

4.8 - Combining model of reference, standards and framework

Numerous interoperability frameworks, standards and models of reference exist. Sometimes they are overlapping. Sometimes they are complementary. Not of them are covering the whole spectrum of needs for interoperability of enterprise application for eBusiness Collaboration within heterogeneous environment. New families of standards are produced combining usage of different standardization frameworks, as illustrated for example by Product Lifecycle

Management set of standards coming from Automotive or Aerospace. Numerous researchers are also investigating how to combine and federate legacy standards, frameworks and models of reference in order to respond to emerging needs for interoperability and higher level framework.

4.8.1 - STEP and integrated knowledge

Some frameworks mixing usage of standards have been produced by research activities such as EPISTLE that was exploited within the POSC-CAESAR community and PLCS (Product Lifecycle Customer Support) community, mixing usage of STEP application protocols, web services and OWL based ontologies. A first EXPRESS to OWL binding was proposed (STEP4free) through intermediate transformation in UML (Unified Modeling Language). Using existing standardized binding between EXPRESS language and UML, it occurs that the proposed transformation implies important lost of information, as no equivalence was established between the different constructs of the different modeling languages. In addition, the proposed transformation is unidirectional. Finally, the targeted processes are not clearly established, making it difficult to understand how and when to use it within application development process in order to address interoperability of enterprise applications. The currently adopted approach consists in defining external classification within STEP models which are referring external classifications defined within ontology in OWL. The mechanism is quite complex and makes it difficult to validate information which is distributed between two representations based on two different formalisms.
Through the S-TEN project (Klein 2008) (Leal D. 2007) (S-TEN), an integration of structured domain standards by means of ontology is proposed, through a binding of STEP (EXPRESS – i.e. schema- and part 21 – i.e. data) to OWL (Web Ontology Language). This mapping is done through a specific implementation model developed within the JSDAI tool, allowing to mix STEP Application Reference Model and Application Integrated Model definition, and to improve models defined within STEP Application Protocols from an implementation point of view. After analysis, some issues exist due to the fact the focus is made on implementation optimization more than on information interchange and knowledge sharing within networked organization.
(Ghodous 2002) proposes an original methodology for the integrated representation of knowledge related to the product, encompassing associated existing services and processes. This methodology allows reuse of existing product models, service models and process models by different experts involved in product development. It also allows transformation of models in standardized models in order to share them. (Ghoudous 2002), (Slimani2006) and (Kuhn et Al. 2008) explains how to develop an environment using web distributed semantic techniques which could support such integrated representation. Such an environment is based on (Slimani 2006), which describes a knowledge exchange and sharing system supporting collaborative design. Within such an environment, (Ferreira Da Silva 2007) proposes on innovative ways to discover semantic alignment, and apply it on top of product data within "Building" domain, based on STEP application protocols. An EXPRESS to OWL DL mapping is proposed, in order to illustrate and validate this method through the FUNSIEC project. (Kuhn et Al. 2008) illustrates how conflicts mitigation in collaborative design can be achieved within the described intelligent environment. In parallel, (Ghafour et Al. 2007) and (Dartigues et Al. 2007) elaborate a methodology for ontology based integration between design phase and production phase, including interoperable semantic knowledge based "Features" ontology (Ghafour 2009). This notion of contextualized interoperability is further developed by (Hoffmann 2008) through establishment of a method aiming to established context-based semantic similarity across ontologies, which is used for elaboration of multi-

view semantic models for product feature description (Hoffmann et Al. 2008). The focus of all these activities is mainly establishment of federative knowledge bases for the engineers, but don't deal with interoperability of technical enterprise applications. The considered standards and associated frameworks are coming from the communities I consider, i.e. manufacturing (STEP) and Semantic Web (OWL). These activities highlight the potential of these technologies to support computer aided Product Lifecycle Management.

Considering all the numerous research activities related to STEP and knowledge technologies, it appears that the Web standard for definition of ontologies, OWL, is very often used in conjunction with STEP, with very different intention and proposed bindings. No consensus exists today to propose standardized binding. Proposed bindings are most of the time unidirectional. They don't cover all the EXPRESS constructs. Finally, they are not based on semantic equivalence of the modeling constructs of the languages which should allow bi-directional transformation allowing obtaining equivalent models, which can all be used as reference models.

4.8.2 - STEP and MDA

The common developed idea is to derived UML (Universal Modeling Language) informational model from STEP application protocols, based on the standardized binding produce by ISO STEP (part 25). The informational model is completed by a computational model, defining some services, and then used to produce code for specific platforms, most of the time eBusiness platform with generation of XML (and XML Schema) and Web Service Definition Language. (R Jardim-Goncalvez, et Al. 2006) studies other schema languages such as Schematron in order to add rules which are not supported by XML schema.

(Price et Al. 2006) proposes a scenario where EXPRESS language will be mapped with UML and formalized as a MOF model, in order to include it in the set of languages allowing to define ontologies which are considered by the OMG's Ontology Definition Metamodel[17]. Doing so, it should be possible to take advantage of mappings formalized by means of the Query View Transformation OMG's specifications in order to translate a model from a language to another. Strong collaboration has been established between the project dealing with this activity, the MEXICO Project, and research community dealing with Enterprise Applications Interoperability. (Krause, Kaufmann 2007) describes the proposed approach, which led to formalization of several research demonstrators[18], and to the under-definition specification at OMG, the reference metamodel for the EXPRESS information modeling language. The validation process is running and will finish in December 2009 (dtc/2009-05-13).

In parallel, OMG PLM Services and PLCS PLM Services are building their specification on top of UML, trying to take advantage of the model driven approach. As a result, several specifications were made available at OASIS and OMG, with implementation of reference or commercial implementation of integration platforms. So far, without an available agreed profile, mappings are most of the time performed by hand, without formal specifications of the mappings. In addition, numerous adaptations are made related to implementation improvement viewpoint, which are difficult to capture. Consequently it is about impossible to

[17] It includes OWL, RDF schema, ISO 24707 Common Logic, Topic Maps, mindmaps, etc.
[18] The demonstrators are available on the ModelAlchemy web site, which gives access to the Mexico Document Management server at http://www.modelalchemy.com

define which representation is really the reference. In addition, bad support of standardized specification by software tools makes it difficult to take advantage of these formal specifications and to reach easily expected interoperability. Finally, due to evolution of the different used formalisms (e.g. UML 1.4 to UML 2.1), the specifications should be regularly updated. It is most of the time not the case, due to the cost of maintenance and configuration management of all the interrelated specifications.

4.8.3 - MDA and SOA

Within ATHENA program, a group of researchers (G. Gorka, et Al., 2008) adopted a holistic perspective on interoperability in order to achieve real, meaningful interoperation between enterprises, based on the need for a structured approach to collect, identify and represent the current state of the art, vision statements, and research challenges. The starting point was the IDEAS model of reference. Several integrations are to be achieved. Conceptual integration focuses on concepts, metamodels, languages and model relationships. Technical integration focuses on software development and execution environments. Applicative integration focuses on methodologies, standards and domain models. Different aspects are to be covered: services, information, process and non-functional aspects. By applying and integrating ATHENA results, I identified numerous drawbacks I described within (), which led to me to launch the thesis.

4.8.4 - Holonic metamodel for computer aided manufacturing

Working on holonic metamodel for computer aided manufacturing, (H. Panetto 2006) proposed an interoperability classification is based on four perspectives. First perspective is the level of interoperability according LISI (Levels of Information Systems Interoperability).
The second perspective is the type of interoperability, according LCIM (Levels of Conceptuals Interoperability Model). The third perspective is the abstraction level of model, according MDA (Model Driven Architecture). The last perspective is the type of model, according Zachman framework. The build framework allowed developing the concept of "holon" for reconciliation of organization and product viewpoints when dealing with product manufacturing. More precisely, holon concept definition was adapted in order to solve the problem of synchronization between physical views and informational views of the same object: "an aggregation of an informational and a physical part" (H. Panetto et Al. 2007). Holon concept is used for horizontal interoperability, i.e. for two or several systems or applications from the same level in the enterprise hierarchy needing to exchange information or data in order to perform a common objective. MDA is used for vertical interoperability, i.e. the interoperability between applications from different enterprise levels. ISO STEP Application protocols are eventually used to structure "Product" data. The undertaken research is more focusing computer aided manufacturing than interoperability of technical enterprise applications.

4.8.5 - Interoperability testbed

(Ray, 2002) points out that *"the number of communication standards is growing geometrically"* and that the standards have to represent more and more complex information structures. The current way of preparing standards is reaching its limits. Indeed, it takes a long time to have concerned organizations reach an agreement on which technology to adopt as a standard, and then to develop and test it. The increasing demand for standards is *"outpacing the ability to keep up"*.

From this starting point, he described a three way approach for manufacturing interoperability, based on standardization, fundamental research and assistance to the industry, in particular through development of a research testbed allowing to deal with semantic equivalence metrics, integration negotiation protocols, semantic resolution in Business to Business, service coordination and composition, core product model, modeling language, knowledge description language and quantification of software uncertainty.
(Ray 2002) identifies two overarching issues to achieve interoperability, one being need for more rigor (less ambiguity) in exchange standards, the second being the rapid growth in the number of standards needed. The pursuit of rigor in data standards can be characterized by evolution from old style standard specifications, with definition using natural language, to data model standards (e.g. using ISO STEP EXPRESS) and then to semantic-model standards (e.g. Process Specification Language). Usage of semantic-model is required due the facts that system integration is hard, interoperability continues to grow as a problem among increasingly IT-dependant systems and in order to resolve interoperability problem category related to semantic (conceptual factorization conflicts, conceptual scope conflict, interpretation conflict, reference conflicts) and functional conflicts (model, scope, intention, embedding).

4.8.6 - Interoperability and language expressivity

(Feng 2003) reflects the work done the Manufacturing Engineering Laboratory (MEL) in collaboration with the National Institute of Standards and Technology (NIST). It provides an overview of "information based manufacturing". Within manufacturing domain, complex systems will increase importance of interoperability, especially of distributed systems. An important work is done concerning content, language and expressivity, in relation with standards. (Feng 2003) provides cartography and typology of standard. The cartography indicates for which kind of functions the standards are used. The typology of standard is based on definition of four types. Type Zero corresponds to standards for implementation language (e.g. Java). Type One corresponds to Information modeling standards (e.g. EXPRESS). Type Two corresponds to content standards, domain of discourse, using Type one or an extended version (e.g. Product information modeling and exchange standards, etc. Type Three correspond to architectural enterprise framework standards such as Reference Model for Open Distributed Process. The way to combine together standards when federating heterogeneous platforms in order to achieve interoperability of technical enterprise applications is not addressed. A stronger focus is putted on authoring technical applications and on product model artifacts, which is out of the scope of my thesis.

4.8.7 - Systematic discipline of ontology evaluation

(Ray 2006) promotes creation of a systematic discipline of ontology evaluation to ensure quality of content and methodology, with a result evaluation of several languages as upper ontology providers and promotion of Common Logic[19] (Sowa 1984) ,as a way to aligned several existing languages. Numerous formal language dedicated to machine interfaces and other languages, dedicated to human interfaces, are related to common logic with is an international standard at ISO.

[19] Common Logic: ISO/IEC 24707: 2007 – Information technologiy – Common Logic (CL) : a framework for a family of logic-based languages ».

4.8.9 – MDA and ontology for Digital Business Ecosystem

(Ferronato, 2004) describes the approach adopted within the "Digital Business Ecosystem" project. The objective is to enable evolutionary, self-organization, learning and adaptive behavior of the Digital Business services and system dedicated to Small and Medium Enterprises (SMEs). Needs for service execution environment, mixed with usage of Model Drive Architecture, is highlighted. Several high languages for Business Process Modeling are studied, as Domain Specific Languages. The concept of "Evolutionary Model Repository" is developed, which respond to numerous needs identified for digital collaboration within the extended and virtual enterprise. But lack of maturity of MDA technologies is also pointed out. (Malone, 2007) propose a conceptual framework for Digital Business Ecosystem services, combining different languages for description of modeling languages (UML, XMI), knowledge models (OWL) and execution language for service oriented platforms (XML), within a book dedicated to Digital Ecosystems. Promoted by European Commission, this project strongly focused on technology transfer. As a result, open source solution (DBE Studio[20]) was made available, and formalizations of OMG specifications in OWL were freely made available on the web[21]. However, this research was dealing neither with product modeling and system engineering, nor with the standards, languages and frameworks used by the manufacturing community. How the solution proposed will allow to interconnect legacy enterprise applications is not addressed a satisfying way in order to envisage federated Product Data Management Systems, according the analysis I performed.

4.9 - Conclusion

Numerous existing standard based frameworks addressing interoperability have been studied, as the research activities concerning their federation and joint usage. If standardization and research communities are very active on interoperability topic, effective and continuous interoperability implying joint usage of heterogeneous technologies and frameworks is still far from being achieved. Next chapter list the brakes and issues I identified which are going against achieving interoperability of enter, and which are not addressed a holistic way according performed state of the art and state of the practice.

[20] DBE Studio: http://dbestudio.sourceforge.net

[21] These ontologies are available at http://www.tssg.org/public/ontologies/omg , and are registered on Swoogle at http://swoogle.umbe.edu

Chapter 5- Identified brakes and issues for Interoperability

During the thesis, the different innovative approaches proposed by different concerned communities, being research, standardization and solution providers, were studied and assessed, through iterative prototyping and piloting activities.

The starting point was application of the approach proposed by ATHENA and that I evaluated at the beginning of my thesis. This approach is a holistic approach with simultaneous usage of enterprise modeling, model driven architecture, service oriented execution platforms and ontology. Model driven engineering allows generating business components and to deploy them on service oriented execution platform. Ontology is used for semantic mediation for interchanged documents and messages. Finally, inter organization collaborative process was conceptualized and demonstrated, which allows aggregation of public views on partners' internal private executable processes.

Networked collaborative product development environment have been set-up on the basis of prototype reusing systematically existing standards, allowing identifying important brakes and issues related to interoperability.

Some brakes have been identified, and interoperability issues formalized, which are requiring innovative solutions. Sharing these issues and brakes is a required step in order to achieve third level of interoperability maturity according NEHTA's Interoperability Maturity Level. I produce my personal list of issues and brakes coming from gain experience through research projects I contributed to and from analysis of the state of the art. The listed is provided by Table 11.

N°[22]	Designation	Description
I1	Multiple formalisms and non preservation of the semantic	Multiple formalisms and non preservation of the semantic between applicative components but also engineering artifacts when using model driven approach leads to poor communication between systems and loss of data
I2	Break with legacy system	New paradigms are leading to breaks with legacy systems and creation of silos
I3	Multiple dissociated boundaries	Virtualization leads to multiple dissociated boundaries for the different concerned systems: organizational, decisional, knowledge domain and technological system
I4	Non alignment of identifications, decompositions and types	Non alignment of representations of a same real thing within different interconnected systems, in terms of identification, decomposition and typing makes data exchange and sharing difficult.
I5	Application development logic	Application Development logic when systematically using Commercial Off-The-Shelves leads to "black box" which is going against applications interoperability and reduces the level of maturity of enterprises.
I6	Cost to establish and maintain interoperability	The cost for establishment and maintenance of interoperability are too important with current practices, architectures and non flexible technologies.
I7	Diverging interest concerning interoperability	Diverging interests concerning stakeholders concerned by enterprise applications interoperability is going against the adaptation of relevant solutions.

[22] Note: the different issues of the table are referred in the document as Table 11-Ixx.

N°	Designation	Description
I8	Non flexible and reconfigurable solutions	Non flexible and reconfigurable solutions components are too often used.
I9	Babel syndrome	Babel syndrome is not considered enough.
I10	Closed solutions	Closed solutions are going against interoperability.
I11	Standards oriented according a single viewpoint	The standards are often oriented according to a single viewpoint, which is going against a global solution.
I12	Standards oriented according a single type of model	Standards are often oriented according a single type of model (information, service or process), which is going against a global solution.
I13	Interoperability considered too late	Interoperability is considered too late, leading to overcost.
I14	Important cost of creation of ontology for semantic mediation	Important cost of creation of ontology for semantic mediation is often not considered by researchers.
I15	Complexity	Complexity of the concerned systems is challenging when willing to achieve interoperability.
I16	Sliding of formalism usage	As soon as formalism is created to cover such or such phase of lifecycle of application, a trend exists to use this formalism for the following phases of the lifecycle, leading to multiplication of languages and technological silos. We call it “sliding of formalisms usage.
I17	Impact of COTS parameterization	Impact of Commercial Off-The-Shelves parameterization during deployment and exploitation phase is most of the time not considered, while operational interoperability concerns only applications used in operational environment and not software products.
I18	Legacy components not considered by conceptual frameworks	Most of the innovative model driven approaches proposed by research are not considering reuse of legacy implemented solutions and operational application, which imply retro-engineering and retro-conceptualization
I19	Missing or insufficient governance of standards	For numerous organizations, there is missing or insufficient governance of standards
I20	Graphical and textual languages	The drawback of graphical formal languages and their association with textual languages without conceptual alignment leads to inappropriate and useless sets of emerging standards.
I21	Federation of systems	Federation of managed resources being controlled and filtered by several heterogeneous systems is not addressed by interoperability research.
I22	Data exchange: not only translation but transposition	Data exchange is too often considered as simple translation while it requires transposition within several contexts using different independent sets of business rules.
I23	Numerous incompatible and coexisting frameworks	Several incompatible and coexisting frameworks dealing with or impacting interoperability lead to technological silos.

Table 11: Interoperability brakes and issues

All the research activities identified and followed during the state of the art are not addressing these issues, or are not addressing simultaneously these issues. We have consequently a set of partial solutions, which are often overlapping, redundant and incompatible.

Some of the issues are quite obvious, well known, already described within the literature or put in evidence within the previous chapters of this report. I will not detail them.

It is the case for numerous incompatible coexisting frameworks (c.f. general introduction and the interoperability framework section), missing or insufficient governance of standards (c.f. importance of governance within NEHTA Interoperability Maturity Model), legacy, complexity, important cost of creation of ontology for semantic mediation, interoperability considered too late, standards oriented according a single viewpoint (c.f. general introduction, in particular concerning the SAVE project), Standards oriented according a single type of model (c.f. the classification of standards within chapter), closed solutions, Babel syndrome, non flexible and reconfigurable solutions, cost to establish and to maintain interoperability., diverging interest concerning interoperability,

Some other issues and brakes were identified during my practical experimentations of solutions proposed by research projects with usage interoperability frameworks and standards.

Encountered difficulties have been analyzed and lead to formalization of issues which are to be considered if willing to achieve interoperability of enterprise applications within networked organizations, applying approach proposed by ATHENA.

It is in particular the case for preservation of the semantic when using multiple formalisms (Table11-I1), multiple dissociated boundaries (Table11-I3), Non alignment of identifications, decompositions and types (Table11-I4), application development logic (Table11-I5), sliding of formalism usage (Table11-I16), impact of COTS parameterization (Table11-I17), legacy component not considered by conceptual frameworks (Table11-I18), Graphical and textual languages (Table11-I20), Federation of systems (Table11-I21), data exchange – not only translation but transposition (Table11-I22),

5.1 – Preservation of the semantic when using multiple formalisms

All along the lifecycle of an application, and within each phase of the lifecycle, multiple, heterogeneous and non equivalent formalism (languages) are used by the different implied actors to describe application and associated information resources. These languages are often based on heterogeneous paradigms. An example can be, for a single application, description of the concepts of the domain concerned by the application using entities/relationships model (requirement phase, paradigm A), then object orientation for the models produced by the software engineer (design phase, paradigm B), then usage of a software development language which is not object oriented (paradigm C), with usage of relational database for data storage (production phase) and finally usage of ontology for semantic annotations by users (exploitation phase, paradigm A, B, C, D…). Nevertheless, as illustrated by Figure 33, the reality described by these different formalisms can be exactly the same, and refer to the same conceptual business or discipline model. It is consequently difficult to establish logical and formal mappings between the different representations. These mappings are most of the time made by people, without help of computerized application, on the basis of literal document,

non explicit descriptions which can be interpreted different ways. As a result, misinterpretation is often done during the development process.

During the exploitation phase, the documents associated to the application are non-maintained and finally not aligned with the application. It makes it difficult to maintain and to integrate applications, and to insure interoperability of applications, without accessing the code. Usage of different languages and paradigm leads to what the community who mapped objects and relational databases called "impedance mismatch", i.e. necessary lost of information when translating from one language to other language. For example, considering three technical applications (A, B and C) used at different phase of the product lifecycle (PLM) and a neutral reference model for data exchange, as some concepts can be formalized only for some applications and not for the others, there will be a lost of information when transferring and exchanging the information between the applications, as illustrated by Figure 34.

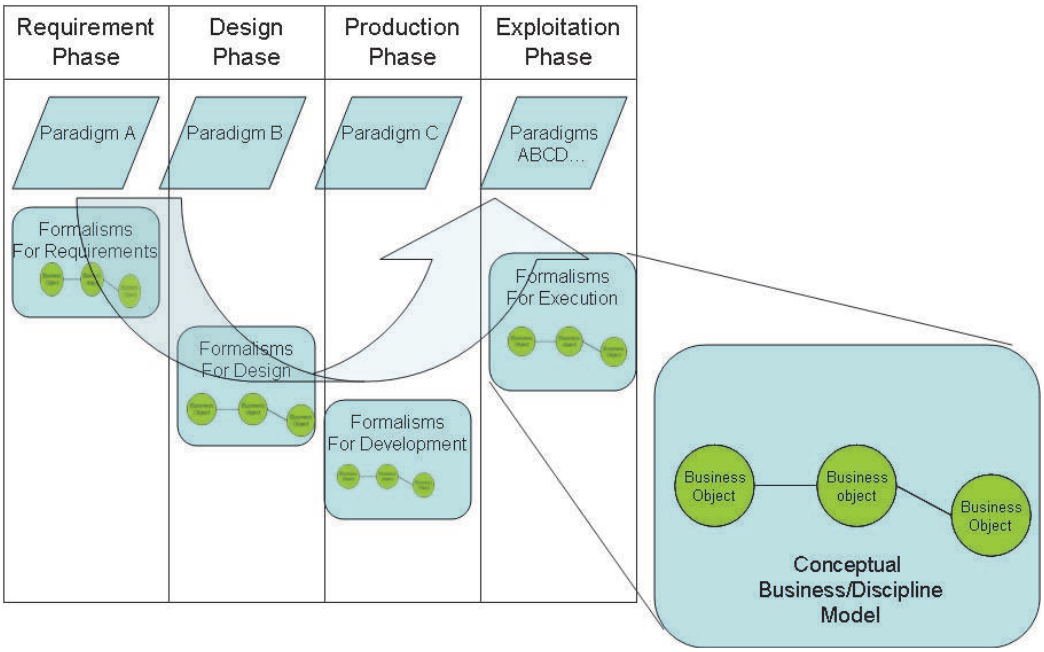


Figure 33: Non preservation of semantic all along application lifecycle

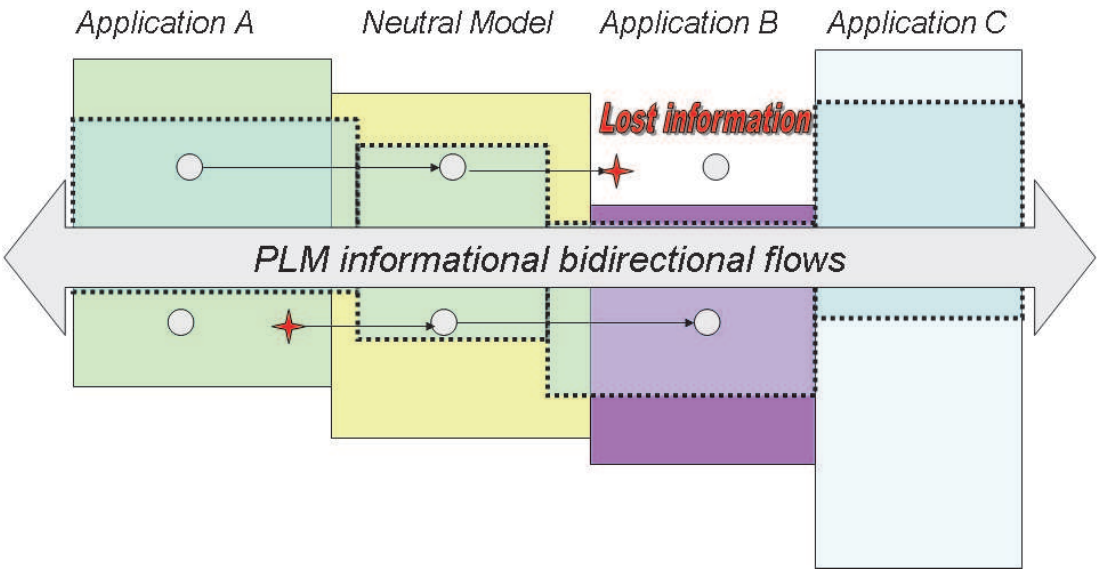


Figure 34: Lost of information between PLM applications

How consequently insure quality of data exchange between application at a reasonable cost and avoiding loosing information? When using model driven approach, the Computer Independent Model, the Platform Independent Model, the Platform Specific Model, the code and the data exposed by the applications should also preserve business concepts, despite usage of heterogeneous language. Creation of new and more expressive languages is not a solution, as it will necessarily imply differences with less expressive languages and lead to semantic lost.The different language mappings provided as open specifications or standards are most of the time uncompleted and made only in one direction. Retro engineering is consequently not supported. It consequently leads to incoherent results when using different chains of transformation. For example, given four languages L1, L2, L3 and L4 which are used to model the same business concepts, representing a transformation according a given mapping with the "⇨" symbol, the chain L1 ⇨ L2 ⇨ L4 and the chain L1 ⇨ L3 ⇨ L4 may lead to different results.

Using model driven engineering and iterative federation of application, we should expect bi-directional transformations, as illustrated by Figure 35. Taking example of Express, UML, Java and XML languages, we should be able to port modeling construct for any flow of data, but we have in reality important loss of data.

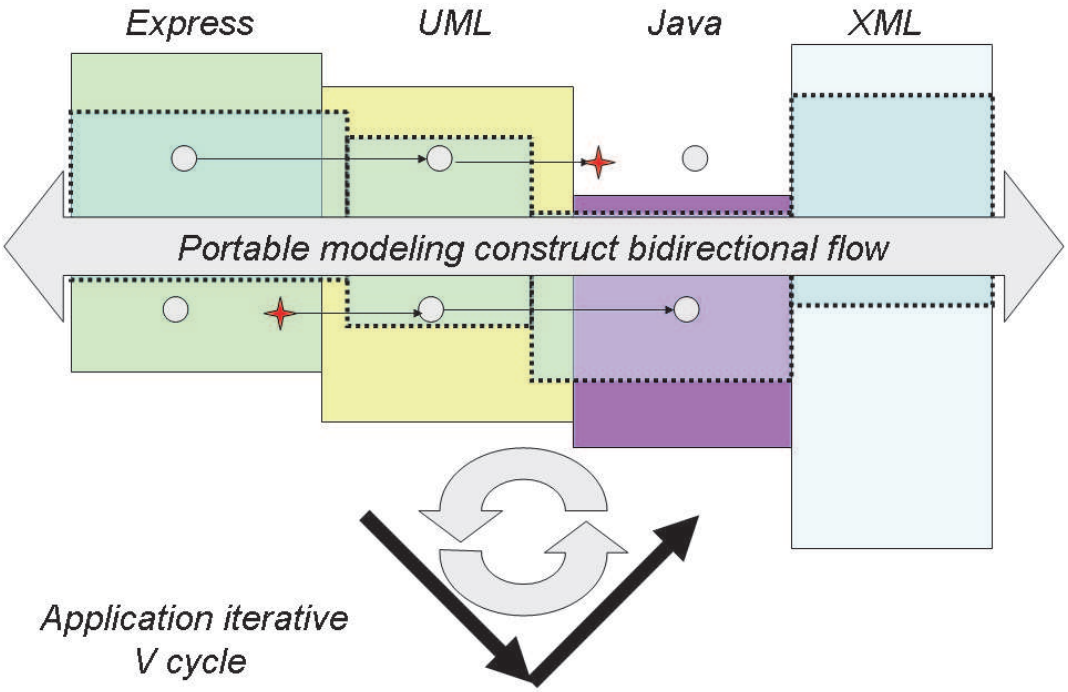


Figure 35: Information lost due to languages mismatch

So semantic preservation, when using several heterogeneous language and bidirectional transformations, is currently not efficiently addressed within the state of the art. The proposed standardized bindings are most of the time unidirectional and do not characterized efficiently lost information. Finally configuration management of the different languages and mappings is most of the time not ensured, as impact of evolution of languages is not reflected on mappings. It is a serious drawback for usage of Model Driven Engineering and ontology to address semantic interoperability.

5.2 - Multiples dissociated boundaries

Numerous boundaries (enterprise, department, domain, project, discipline/business domain, systems) exist with associated internal needs, constraints, interests and processes, and clear establishment of typology of actors and stakeholders. Out of boundaries, for transversal collaboration, a consensus is required, which is not often easy to obtain.

Organizational and decision domain boundaries

They are related to the way the work is organized and how decisions are taken. When work is organized in a project mode, different boundaries are created related in particular to who pay, who define the objectives and who will use the result. As a project always have a limited set of resources, it implies to clearly define the scope and limits of the project. The implied stakeholders decide about intellectual property, distribution and usage rights. Therefore, most of the time, it is not well accepted to pay for the others and to freely give access to the results of projects. Numerous restrictions are then put in place, which is not accurate for products belonging to wide and open communities. Finally, as projects are limited on time, they most of the time don't cover exploitation phase where changes occurs and when interoperability is to be ensured. It is the reason why the project mode has a negative impact on obtaining interoperable solutions. Support activities are absolutely required allow to ensure interoperability of federated systems but are most of the time outside of organizational and decisional boundaries constituted by the collaborating enterprises. Consequently resources for interoperability are most of the time not shared nor maintained. It constitutes causes of non interoperability.

Knowledge domain boundaries

Some boundaries also exist between knowledge domains or disciplines: physics, mathematics, geology, biology, manufacturing, etc. Each domain has specialized specific vocabularies, models, tools, etc. Exchange of information between these domains is most of the time not required, except for multi-discipline approaches. In such a case, common set of concepts should be used to realize a bridge between the used models.

Technological boundaries

It may be required to use together applicative components developed on the basis of different technologies. Or the technologies were not design to work together and are not necessarily compatible. It is then difficult very difficult to use them together. An example comes from the ATHENA program, where integration of solutions coming from Ontology domain (based on usage of RDF) and Service Oriented Platforms (based on WSDL and consequently XML Schema) implies establishment of XMLtoRDF and RDFtoXML translators. Due to impedance mismatch (non semantic preservation), it is most of the time impossible to avoid data lost and exchange traceability. All the issues discovered during appliance of ATHENA results was described on a paper I wrote for my thesis (N. FIGAY 2006b). Similar difficulties exist if using technologies based on MDA (with UML and XMI), relational databases, object oriented systems, etc. Sometimes, mapping technologies are available and are to be integrated to obtain complete solutions. It is the case for example for Object to Relational mapping with numerous solution existing on the shelves (c.f. http://en.wikipedia.org/wiki/List_of_object-relational_mapping_software). Several mappings can be implemented, but are to be selected

and tuned according wished robustness and scalability of the targeted solutions. Some constructs are difficult to map, and are most of the time not. As these technologies are developed by different and competing communities, they are not necessarily willing to produce "compatible" technologies, aiming to produce shelf sufficient solutions. Looking at OMG and at W3C technologies and standards, important overlapping exists (e.g. remote procedure call through OMG's CORBA and W3C's WEB services). Such boundaries are not limited to execution platforms, but also to modeling tools, with solution based on UML, SysML, GME (Generic Model Environment), etc. For a business standard dedicated to produce computable solutions, usage of such or such formalism is always leading to link it to a technological boundary. Choice of CORBA for PDM enablers and WEB services for OASIS PLCS PLM service is an example.

Systems boundaries

Decomposition when defining a system leads to define boundaries. Considering object oriented approach which is derived from systemic, an object expose a public interface and hide internal structure (private methods and attributes). For procedural languages defining functions, the function is known by its signature, but the signature doesn't reveal how the function is internally defined. Internal variables exist which are know only inside the function, and can't be accessed from the outside. We can retrieve this for any language defining processes, being for workflow or orchestration of services. For data serialization, when using files, we can also define what is inside the file and what is outside the file. File management systems are managing information about files from the outside, with set of metadata allowing managing it (e.g. name, size, physical location, kind of file, etc). But the file itself is considered as a black box, and the content can only be accessed with a dedicated processor able to deal with the internal structure of the file. Looking at eBusiness architectural pattern, a distinction is made between what is inside, hidden and private – the back office – and what is exposed and visible from the outside of the enterprise – the front office. For an application, a set of interfaces is provided for human users or external application, which does not reveal internal structure of the application. A principle of the modern portals is to hide underlying complexity of the used systems to the user, presenting a unified public interface and hiding internal structure. In all cases, we are always facing what is system internal and system external. The external interfaces are not always aligned with internal structure.
In some cases, they can be derived from the internal structure having in mind to hide internal complexity and to obtain a component which is easier to use, by hiding internal complexity and proposing interfaces easy to understand by those who will use the component.
In some other cases, the interface can be defined from the outside, and usage of already existing components will impose to wrap these components, i.e. to implement a mapping allowing exposing the components as an implementation of the interface.
In all cases, a semantic mapping will be performed, dedicated to different kind of human actors, being designers, developers or users of applications.

Virtualization effect on multiple boundaries

As stated within the introduction, the aspects or the viewpoints which are defining the boundaries are subject to virtualization, i.e. decoupling between logic and physic, in particular for location. It leads to define an application not as a system, nor as a system of system, but as intersection of different interrelated systems with disconnected boundaries.
Adopting a holistic approach is consequently not considering systems of systems, with a hierarchy of the different viewpoints or aspects considered. For example, an application can't

be only considered as a component of an enterprise within the application context, as an application supporting collaboration between several enterprises is outside of the boundaries of these enterprises.
Consequently several systems of systems should be considered (Virtual Enterprise, Product, Information and Communication physical infrastructure, Information and Communication logical infrastructure) which are interrelated and aggregated within the application context.
The different models the enterprise is dealing with can be application independent, i.e. not related to application engineering, or application dependant (as for example Computer Independent Model, Platform Independent Model, Platform Specific Model defined by Model Driven Architecture). Computer Independent Model are not considered as application independent model, as they are artifacts produced within the scope of application engineering, in order to defined context and requirements of an application.
In the reverse, some application independent model can be modeled using a modeling tool, in order to take advantage of computers for Computer Aided Modeling. But in such case, the produced artifacts are still not dedicated to application engineering. These models nevertheless can be interrelated and they are "reflected" within each other. For an application, reflection means that representations of the models which are outside the application are internally represented within the application. For example, a process supported by an application, where the application is a resource, can be reflected within this application as an executable model, allowing the application to control execution of the process (it is usually the case for a workflow system). But the business process model and the workflow model are two different models, as one is descriptive and the second is executable.

Virtualization and multiple boundaries lead to different decompositions and breakdowns for these models which describe the same reality, but for different purpose. Figure 36 aims illustrating these different interrelated systems of systems: enterprise system, product system, applicative system and physical information and communication infrastructure. For each system, different model artifacts are produced which are produced independently by different disciplines and actors. Enterprise models are produced by analysts. Product models are produced by engineers. Information and Communication models are produced by software engineers for people who will use it within the context of their activities. The Information and Communication physical systems are produced for people who are in charge to set-up and support the interconnected computers infrastructure. An application is at the intersection of all these systems. It aggregates different dedicated models which reflect enterprise, product, Information & Communication infrastructure and business logic of the application. An application can't be consider as a single close system when considering all the environments it belongs to.
Existence of multiple boundaries and more and more systematic virtualization, as stated within the introduction, is making it difficult and complex to insure coherent aggregation of models all along their lifecycle, and to insure continued and adaptive interoperability as expected for eBusiness collaboration. It is also a brake to interoperability because it is not possible to control changes which may impact interoperability within the multiple interdependent boundaries. Current existing frameworks for interoperability are not currently addressing such aspect

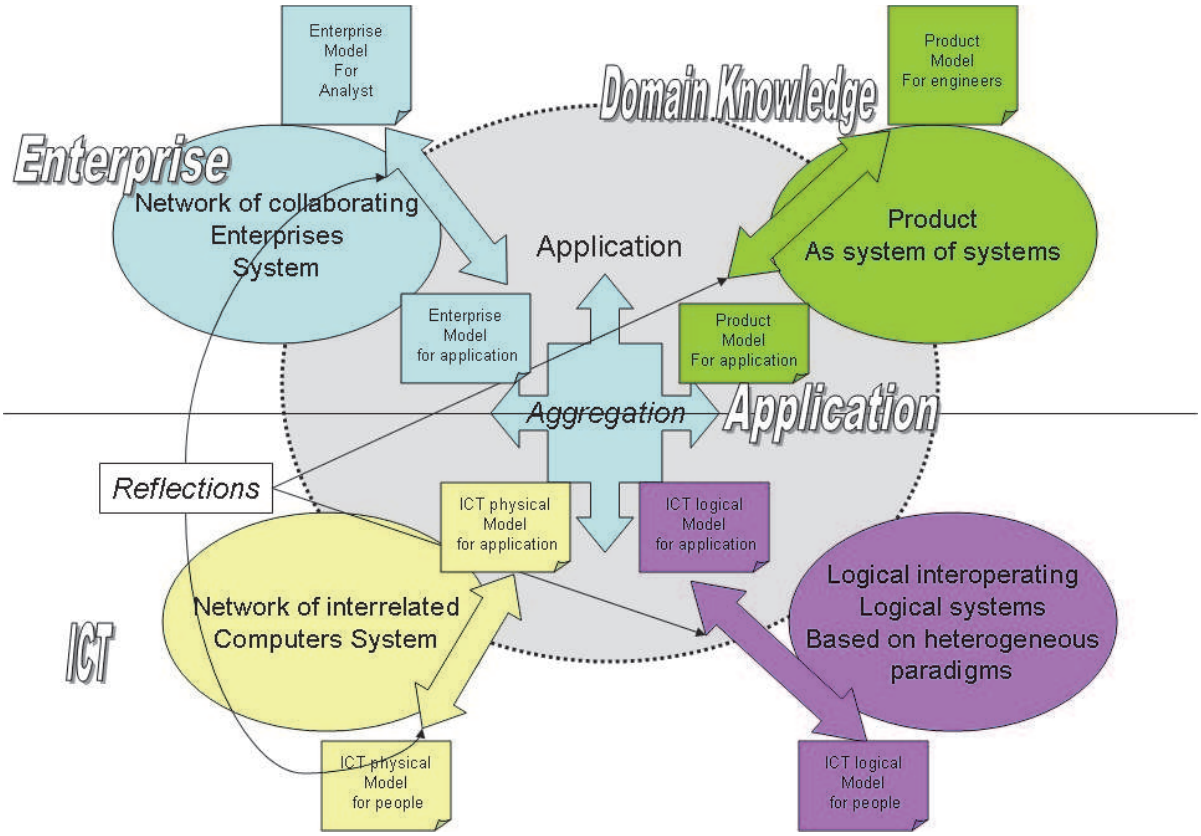


Figure 36: Application and considered systems

5.3 - Non alignment of identifications, decompositions and types

An important brake is the non alignment of identifications, decompositions and typing of items contained by the different systems and reflected within applications. As discussed within the previous section, an application is at the intersection of several systems (enterprise system, informational system, network of computer system, etc) which exist independently. For each system, the "things" to consider, the way to identify type and organize them, using classification, decomposition, aggregation, interrelation, layering, etc …, are different. In particular, for different representations of a system, several decompositions can be considered leading to heterogeneous breakdowns with different levels of granularities. It may lead to layers with different levels of granularity. It may also lead to several "disconnected" layers, as for example when considering decomposition in atoms for physic, molecules for chemistry, cells for biology, organs for medicines, etc.

Let's take a concrete example to illustrate how it impact interoperability of enterprise application. The considered "think" within the real world is a person, as illustrated by Figure

37.

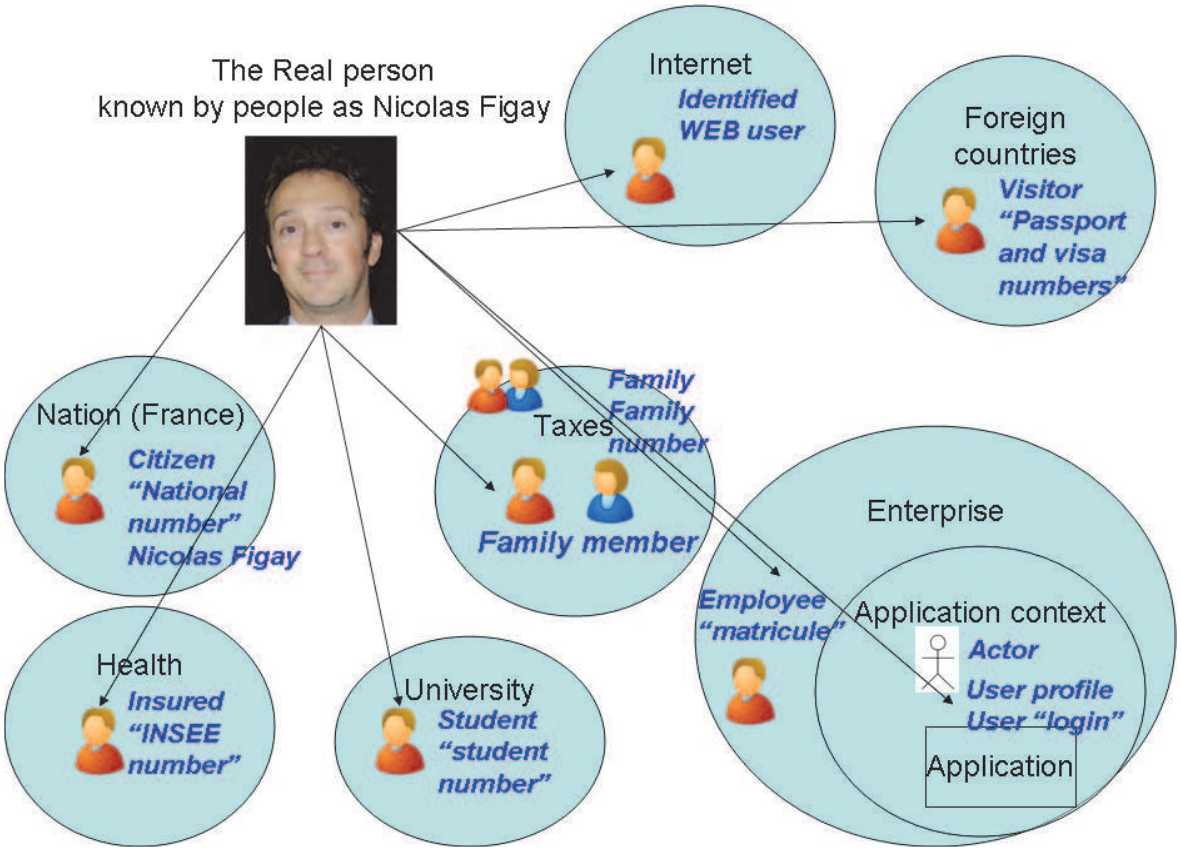


Figure 37: From the real world to information system

A person exists within an organization only if identified, referenced and typed. Within modern societies, a person is first a citizen. At the birth, each person is registered, with declaration of a composite name (first name and family name) which will be used by people. In addition, a set of identifiers and types will be created for the different organisms we are referenced by. As example, in France, it will be national identity number with person considered as a citizen, social security with the person considered as insured person, etc. When entering an enterprise, the person is considered as an employee with as identifier the "employee number". When traveling in foreign countries, we need to be registered with a passport number and to obtain a visa, in order to be recognized within the foreign countries where we are going, as visitor or temporary resident. For taxes payment, the person is considered as a member of a family (called "Foyer Fiscal" in France). So several typed representations of a same real "thing" exist which are reflected within the applications as typed data, with contextual information related to purpose of the organization referencing this "thing" within its information system. Applications are managing and processing only data (and not the real things) and it is needed to ensure that information within the applications is aligned with the real world (facts) and up to date. This is not always insured, due to time required for alignment which can be long or inefficient when non accurate processes are applied. Finally the different sets of information, within the different organizations, should be aligned and coherent as much as possible and when needed, in particular for collaboration.
To insure coherency and information exchange or sharing, some semantic links are to be insured between the different categorizations used by the different organizations. For our example, it means coherency of information related to a given person as citizen, employee, student, etc. When using computers and software, the different heterogeneous formal languages used for machines, with different used syntaxes, grammars, codifications and physical data types, are also to be coherent.

Finally, links are to be established between the different used identifiers, being internal to an organization (e.g. country, enterprise, etc) or internal to an automated system based on computer usage. It could be for example line number within a database, Interoperable Object Reference for distributed object instance, Universal Resource Identifier and Locator for an Internet resource, data object number for STEP part 21 file, etc. Let's note that application users should not be impacted by how information about a person is identified structured or types within the software. They only need to access to literal data including identifiers provided by the different organizations (e.g. national number), with alignment of their own perception of the reality and how it is reflected at user interface. The identifiers related to organizations are usually used as external keys, which are independent of software systems.
The internal identifiers are not significant and should not be provided to users, in particular because they are not portable between systems when willing to exchange or share information. The different possible paradigms used to describe the types should also have no impact.

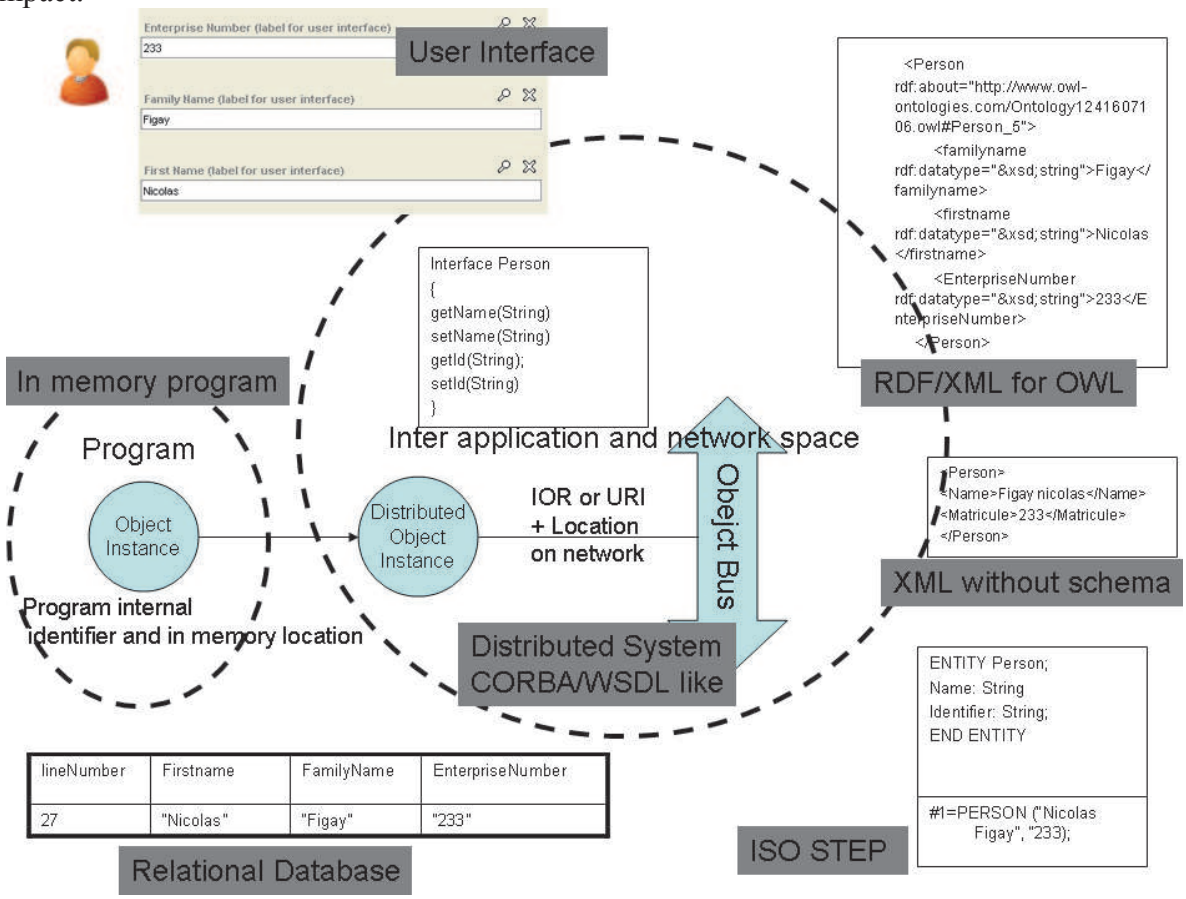


Figure 38: Business Concept all along the lifecycle of the application

As an example, for a user of an application (operator), usage during design and coding phases of UML classes (object paradigm), OWL class (classification, semantic graph or descriptive logic paradigms), EXPRESS entity (entity/relationship paradigm), relational database (tuples) or hierarchical database (RDF graphs) should be transparent when using an application. At the end, what the user will see is only the user interface, where used constructs and languages will not be reflected, only the value of literal data, plus eventually readable labels describing nature of printed data (on paper or on screen), as illustrated on Figure 38, where several physical representations of the same person is provided at different places within a distributed application and associated information repositories. The user (operator) only sees the user interface at the end, without knowing which data are finally used to show the result. He also

has no idea about multiple used internal identifiers such as line number within the database (27), data object number within the ISO STEP file(1), URI (Unified Resource Identifier) used for the person within the RDF/XML file (http://www.owl-ontologies.com/Ontology1241607106.owl#Person_5), IOR (Interoperable Object Reference) or UUID (Universal Unique Identifier) used for distributed systems, etc. He also has no idea if used paradigm to design the application or for its implementation is object, classification, descriptive logic, etc. For the people maintaining and connecting applications, this is not transparent as they will have to use the formal representation in order to understand how to establish interoperability (technical point of view, with syntax reconciliation) and if it is relevant within the usage context (usage point of view, with semantic reconciliation).

As illustrated in Figure 39, a same business concept is to be shared by all the actors and stakeholders involved within the different phases of the lifecycle of the application: client, domain expert, project leader, responsible of requirement gathering, designer, developer, and all the operators of the application: business user (actor of business process using application as a resource to perform his activity), business administrator (parameterizes and administrates application from business point of view) and technical administrator (parameterizes and administrates the application from Information & Communication Technology point of view). The concept is formalized using natural language, drawings, formal modeling language, formal development languages and metadata using numerous format and syntaxes within the application itself (as illustrated above).

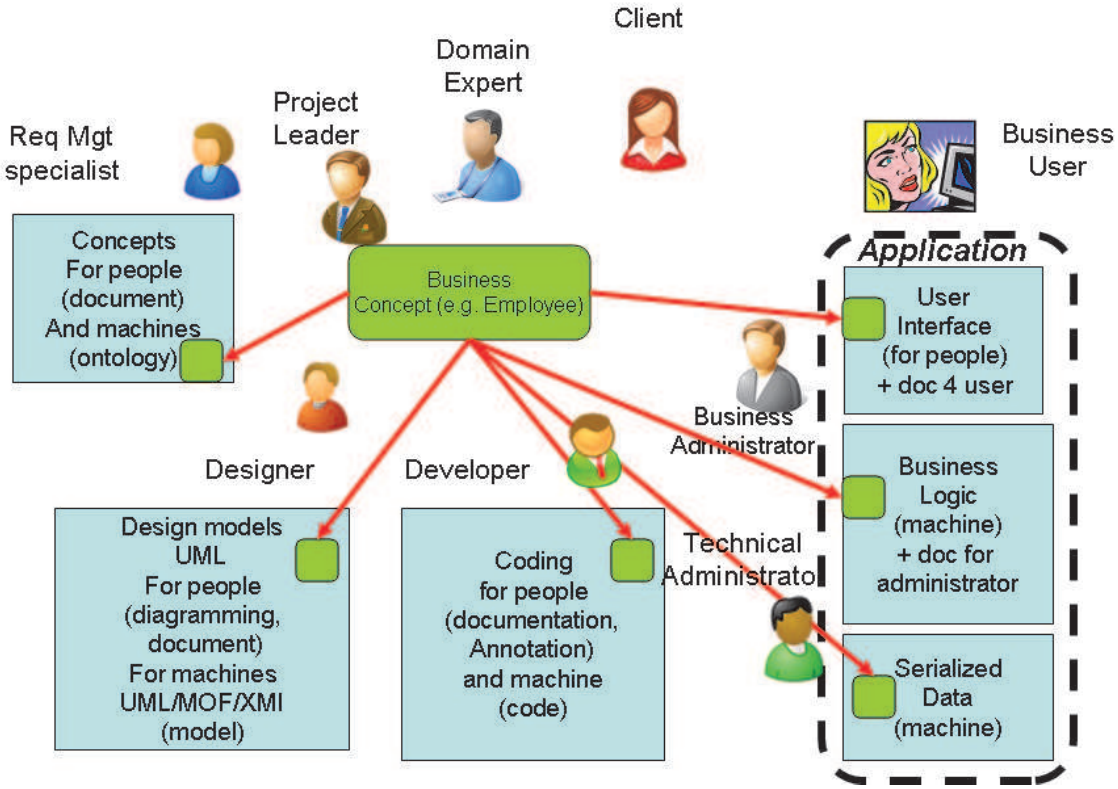


Figure 39: Used formalisms all along lifecycle

For all human readable languages, the coherency can be ensured using the same term with shared definition, and associated unique code for the different used platforms (conceptualization, design, development and execution).

It is most of the time done using a dictionary based on a shared terminology with all associated ambiguity, in particular when we are aggregating already existing conceptual models (ontology), design models, code or application which have been developed separately.

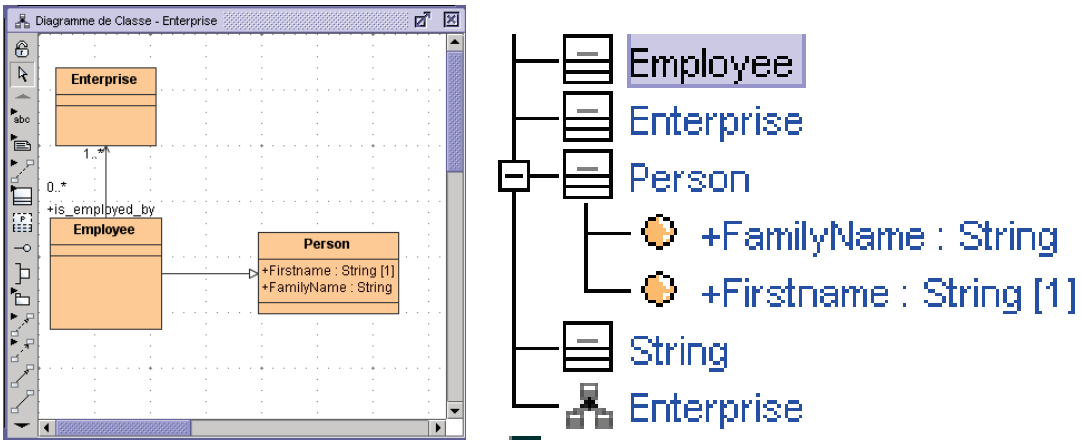
The same term/code can be used for different concepts, or several terms can be used for the same concept, if the process does not allow ensuring alignment. With have in such a case semantic mismatch. Let's be more precise about the example. For requirement gathering, the concept "Employee" will be identified as a business concept that will be linked to the application. A definition will be given in natural language, for example:

> Enterprise: an enterprise is an organization …
> Employee: an employee is a person working for an enterprise, with establishment of a contract. One person can work for 0 or n enterprises. An enterprise has at least one employee.
> Person: a human being, characterized by a family name and first name.

Eventually ontological model will be provided, for example using OWL with RDF-XML syntax:

```
<Enterprise rdf:about="http://www.owl-ontologies.com/Ontology1241607106.owl#EADS"/>
<owl:Class rdf:about="http://www.owl-ontologies.com/Ontology1241607106.owl#Employee">
    <owl:equivalentClass>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Restriction>
                    <owl:onProperty rdf:resource="http://www.owl-ontologies.com/Ontology1241607106.owl#is_Employed_by"/>
                    <owl:someValuesFrom rdf:resource="http://www.owl-ontologies.com/Ontology1241607106.owl#Enterprise"/>
                </owl:Restriction>
                <owl:Class rdf:about="http://www.owl-ontologies.com/Ontology1241607106.owl#Person"/>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
    <rdfs:comment rdf:datatype="&xsd;string"
        >An employee is a person, which is employed by an enterprise</rdfs:comment>
</owl:Class>
<owl:Class rdf:about="http://www.owl-ontologies.com/Ontology1241607106.owl#Enterprise"/>
<owl:FunctionalProperty rdf:about="http://www.owl-ontologies.com/Ontology1241607106.owl#EnterpriseNumber">
    <rdf:type rdf:resource="&owl;DatatypeProperty"/>
    <rdfs:domain rdf:resource="http://www.owl-ontologies.com/Ontology1241607106.owl#Person"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="http://www.owl-ontologies.com/Ontology1241607106.owl#familyname">
    <rdf:type rdf:resource="&owl;DatatypeProperty"/>
    <rdfs:domain rdf:resource="http://www.owl-ontologies.com/Ontology1241607106.owl#Person"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:FunctionalProperty>
<Person rdf:about="http://www.owl-ontologies.com/Ontology1241607106.owl#FigayNicolas">
    <familyname rdf:datatype="&xsd;string">Figay</familyname>
    <firstname rdf:datatype="&xsd;string">Nicolas</firstname>
    <EnterpriseNumber rdf:datatype="&xsd;string">233</EnterpriseNumber>
    <is_Employed_by rdf:resource="http://www.owl-ontologies.com/Ontology1241607106.owl#EADS"/>
</Person>
```

In this file, several classes are declared, such as "Person". Properties such as "firstname" is also declared. The designer will use UML (Unified Modeling Language), using diagramming (class diagram) and modeling (tree and XML Model Interchange):



The model serialized using XMI is the following:

```
    <Foundation.Core.ModelElement.name>Enterprise</Foundation.Core.ModelElement.name>
    <Foundation.Core.ModelElement.visibility xmi.value="public"/>
    <Foundation.Core.GeneralizableElement.isRoot xmi.value="false"/>
    <Foundation.Core.GeneralizableElement.isLeaf xmi.value="false"/>
    <Foundation.Core.GeneralizableElement.isAbstract xmi.value="false"/>
    <Foundation.Core.Class.isActive xmi.value="false"/>
  </Foundation.Core.Class>
- <Foundation.Core.Class xmi.id="_9_5_1_74f019b_1241627768734_778936_17" xmi.uuid="_9_5_1_74f019b_1241627768734_778936_17">
    <Foundation.Core.ModelElement.name>Person</Foundation.Core.ModelElement.name>
    <Foundation.Core.ModelElement.visibility xmi.value="public"/>
    <Foundation.Core.GeneralizableElement.isRoot xmi.value="false"/>
    <Foundation.Core.GeneralizableElement.isLeaf xmi.value="false"/>
    <Foundation.Core.GeneralizableElement.isAbstract xmi.value="false"/>
    <Foundation.Core.Class.isActive xmi.value="false"/>
  - <Foundation.Core.GeneralizableElement.specialization>
      <Foundation.Core.Generalization xmi.idref="_9_5_1_74f019b_1241627914156_293768_76"/>
    </Foundation.Core.GeneralizableElement.specialization>
  - <Foundation.Core.Classifier.feature>
    - <Foundation.Core.Attribute xmi.id="_9_5_1_74f019b_1241627844250_813065_61" xmi.uuid="_9_5_1_74f019b_1241627844250_813065_61">
        <Foundation.Core.ModelElement.name>Firstname</Foundation.Core.ModelElement.name>
        <Foundation.Core.ModelElement.visibility xmi.value="public"/>
        <Foundation.Core.Feature.ownerScope xmi.value="instance"/>
      - <Foundation.Core.StructuralFeature.multiplicity>
        - <Foundation.Data_Types.Multiplicity xmi.id="_9_5_1_74f019b_1241628428140_256239_147" xmi.uuid="_9_5_1_74f019b_1241628428140_256239_147">
          - <Foundation.Data_Types.Multiplicity.range>
            - <Foundation.Data_Types.MultiplicityRange xmi.id="_9_5_1_74f019b_1241628428140_302799_148" xmi.uuid="_9_5_1_74f019b_1241628428140_302799_148">
                <Foundation.Data_Types.MultiplicityRange.lower>1</Foundation.Data_Types.MultiplicityRange.lower>
```

We retrieve again a “Person” class, with attributes such as “Firstname”.

The developer will use Java as programming language:

```
/**
 * @(#) Employee.java
 */

public class Employee extends Person
{

}
```

```
/**
 * @(#) Enterprise.java
 */

public class Enterprise
{

}
```

```
/**
 * @(#) Person.java
 */

public class Person
{
        public String Firstname;
        public String FamilyName;

}
```

Here again we can find a class “Person”, with as attribute “firstname”.

For all used language, some coherency is insured within the example through usage of some common terms as codes, associated to a same concept, with efficient management of potential synonymy.
For people, dictionaries are produced, with explanation of the concept(s) behind the term. For computers, the term is used as unique coding sequence in a given namespace. It is important to consider that computers are not impacted by the meaning of the term.

If an obfuscator[23] is used to automatically rename class names, method names and attribute names, there is no impact on the generated code. Only literal values that are dedicated to interpretation by users should be unchanged. So usage of terms for code used by computer is only used for people: mainly developers, designers and modelers. The exchange of semantic is insured by the process consisting in establishing a well managed reverse functional relationship between codes in one or several naming spaces, and terms corresponding to defined concepts, known by the different persons concerned by the application.
A partition can be established between concepts belonging to the usage domain space, the development and the modeling (for system design or for ontology). All the terms used for a language, being language constructs, name of variable and data literal values, when using software platform, have also to establish relationships between codes (for machines) and terms corresponding to a well know concept. With a grammar describing a language, it is easy to decide to change a keyword by another. It should only be reflected on parsers and compilers.
One issue with programming language comes from the fact that tools related to the programming language are not only characterized by the meaning, but also by the expected behavior of programs obtained with set of tools using the language to produce binary code associated to such or such processor. The "real" world concerned is the world of machines and processors. The designers make the link between the natural language and formal language, allowing easily translating the business logic to program.It is particularly important for the designers to provide all the information related to usage and expected behavior for the users, not on the way to develop it.
Or, when reusing ontology, models, code libraries or binary components developed independently, obtaining semantic coherency is more difficult to obtain than when mastering all the phases of application lifecycle and all the produced artifacts. The shared semantic is managed outside of the components, and is the "glue" between the components. It can be managed by publication and sharing of interfaces or protocols, as consensual formal models with associated definitions with are defined independently from components. It is the case for data exchange (e.g. XML vocabulary, STEP application protocol, CORBA interface specifications, WSDL services definitions, Process models or ontologies).
However, such interfaces do not allow most of the time to precisely defined behavior of a system nor to insure interoperability between several systems.
It is the reason why the only case justifying code decompiling is the interoperability! It allows clearly understanding of differences between published public interface and real internal model, which can be very different. It is the case in particular whit wrappings of components or implementations of data mappings. It also allows understanding why expected behavior is not the real behavior, or some impedance mismatch due to internal data structure.
Finally, the way the real thing is formalized by such or such domain may also lead to distinct decomposition, within the usage context, or within the implementation context. For example, considering industrial product, different breakdowns are used which are very different according it is made for functional decomposition, for digital mock-ups or for assembly lines. For a given breakdown, the granularity of the decomposition is usually different according the fact we are using a management tool, such as a Product Data Management systems, or an authoring tool such a Computer Aided Design tools.
A more precise example, for geometric definition of mechanical part, is related to geometric constructs such as points, lines, surfaces, etc… which are managed by Computer Aided

[23] An obfuscator is a tool allowing transforming code in order to change names of classes, attributes, functions, etc… For example, person will be replaced by A1, employee by A2, firstname by B1, family name by B2, etc… The goal is to make it very difficult to understand the code obtain if decompiling a program. It avoids illegal retro-engineering in order to capture the know-how of the software provider.

Design tools, but not by Product Management Systems. When dealing with part geometry, several representation and decomposition may exist if dealing with exact geometry or meshing. The decompositions can be different not only in term of granularity but also in term of considered components. Several parts with homogeneous properties can be aggregated in a single part for calculation.

As a conclusion, I showed that ensuring coherency of numerous used identifiers, types and decompositions is important to ensure semantic preservation and interoperability. Or most of the time, when aggregating preexisting information, models and software components, which have been developed and managed independently within different boundaries, alignment of identification, decomposition and typing between systems reflected within applications is not ensured. It is true for product models, organization models or Information & Communication infrastructure models. As a result, it is difficult to ensure coherency of distributed information managed by several applications. Consequently, it is also difficult to ensure efficiently data interchange and sharing, and finally interoperability of applications.

5.4 -Application Development logic

When developing an application, different logics exist. “In house” applications are supporting operations and solve some problems enterprise is facing. Commercial software products are targeting a market and providing a generic solution. For “in house” application, the development is directly related to users of the developed software, which is strongly aligned with targeted usage in specific context of the client. The conceptual model of the application is aligned to specific vocabulary of users. Openness and usage of external standards are usually not considered. For commercial software solution, the client is the marketing department. The idea is to produce a generic product that can be used by the larger market on numerous platforms. Consequently, the solutions should allow parameterization in order to adapt the product to the operational context of the client. The internal model of the product must be specialized in order to be aligned with specific internal model of the organization using the software product as an application. For commercial product, the provider will generally target his product to be compatible and useable with his portfolio of solutions, but not necessarily with other products provided by other providers. It will depend on his position on the market (leader or not). It will also depend on his strategy. These two logics have an important impact on interoperability of applications. As they prevent using open standards, they are going against easy integration and federation of applications with other applications.

5.5 - Sliding of formalism usage

From conceptualization phase to runtime and exploitation phase, a temptation exists to reuse formalism of a given phase for the next phases, in particular for execution.
For example, UML (Unified Modeling Language) can be used not only for application design, but also for Computer Independent Models allowing to conceptualize a domain, or to describe an organization from a business analyst point of view. UML can also be used for Platform Dependant Model taking into account the programming language used (Model of Implementation). Some efforts exist to develop executable UML. Finally, when adopting Model Driven Architecture, model artifacts could be used as software components of an application in operation (in particular for transformation of data with heterogeneous models).
This trend does not exist only for UML, but also for numerous other languages. It is the case for example for the Business Process Modeling Notation (BPMN), ISO STEP EXPRESS language or Web Ontology Language. BPMN was designed having in mind the model

produced by the business analyst will be the one use by process execution engine. Reality is different, as models produced by business analysts are most of the time not useable by developers. The EXPRESS language designed for ISO STEP, in order to have a dedicated language, independent from the technologies used for software implementation, can be considered like a language to be used at the conceptualization phase: creation of a computer independent model to describe a domain. As STEP technologies are used to implement data interchange or sharing between application, the language is used also by developer (production phase) and during runtime (exploitation phase). But what about the design phase? How to exploit an EXPRESS model (application protocol) when UML is used to design an application? A temptation here is simply not to use UML at design phase to formalize the information model, in particular because of semantic mismatch, and to directly use a java binding for development.
For Web Ontology Language, the produced model can be used at the conceptualization phase (Computer Independent Model describing a domain of knowledge), but also at runtime, so in development and exploitation phase. So same question than EXPRESS: what about the design phase?
How these different languages can be used in conjunction all along the lifecycle of a project illustrates what I call "sliding of formalism usage". As soon as activities within a phase of the lifecycle is computer aided, it will lead to establishment of a formal language and associated technologies that can be used at execution phase. Sliding of formalism usage is an interoperability issue. If using one technology and associated language all along the lifecycle of a project solve interoperability issues for this project, it don't prevail silos between applications. Such approach adds on more formalism to deal with for software components.

5.6 - Impact of COTS parameterization

Most of the time, the impact of Commercial Off-The-Shelves (COTS) parameterization during deployment and exploitation phases are not considered. As Commercial Off-The-Shelves software products are targeting a market, i.e. several clients, software providers are providing generic solutions which have to be adapted to the operational context, organizational (specific work organization) and technical (specific hardware and software environment). Usually, it can be done several ways, or through parameterization, which does not imply to produce any code, or by allowing code extension. Ways to perform both are usually described within the product documentation, and is most of the time required. Very often, buying COTS, it is more and more considered that it should be used as it is with minimum customization, in particular because maintenance of customized application is to be performed by the enterprise (and not by the provider) and because it may lead to rewrite the specific code when updating software product.
In reality, any adaptation, being parameterization or customization, is always mandatory with COTS, and have the same result, i.e. modifying behavior of the application. The difference is that customization requires having developer knowing a programming language, while parameterizations don't. But customization is more robust and scalable. The more important problem is coming from extension which is implementing already existing services by ignorance, or misusing existing services. It is often the case due to the too numerous functions enabled by modern COTS. Consequences are that each application is different, and requires adaptation for any standardized interface. One belief is that applications will interoperate if the same software product is used by the partners, with the same customization. It is absolutely false, in particular for enterprise applications, as the applications are not supporting the same activities within a collaboration process. Finally an issue exists for customization to

ensure coherent change propagation for all the interfaces of an application (user interface, software interface) and for all its components (database, business logic implementation, etc).

5.7 - Legacy component not considered by conceptual frameworks

Most of the proposed conceptual frameworks does not address properly reuse of legacy systems. Legacy systems can be already developed software components or already deployed applicative components. The exception concerns frameworks dealing with integration. But integration is most of the time addressing a posteriori needs for interconnection of applications, which were not initially design to interoperate. Consequently, using integration frameworks, in order to federate enterprise applications, remains very expensive and complicated.

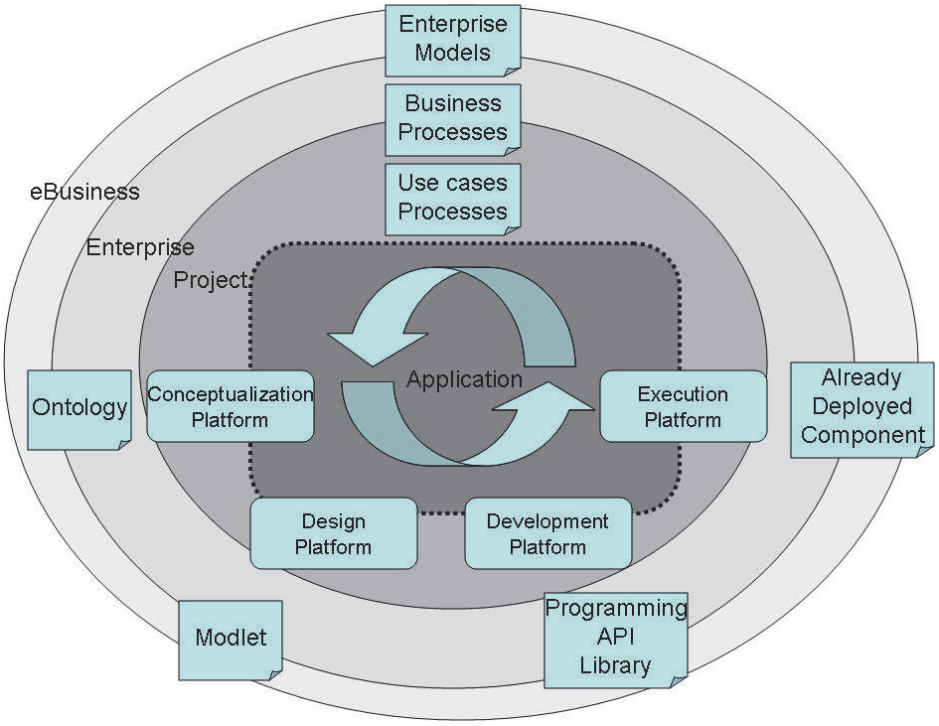


Figure 40: Reuse of legacy for enterprise applications

When integration is done, a very difficult issue is then the management of the configured components all along the lifetime of applications and of the information systems. Within the federation of application, evolution and changes will imply to deal with the pre-existing already deployed components, pre-existing components on the shelves (similar to equipment) or to-be developed component. The components can be digital artifacts related to conceptualization time (e.g. ontology), at design time (e.g. model component or "modlet"), at development time (e.g. Application Programming Interface and associated library) or at execution time (e.g. already available services, enacted capabilities or online database)

Figure 40 illustrates how these different components, which can encompass eBusiness or enterprise standards, are related to usage of specific platforms used by the different actors involved all along the life cycle of an application. It includes conceptualization platform, design platform, development platform and execution platform. The pre-existing components can be reference at project level (project reference), enterprise level (enterprise reference, standards referenced by standardization policies) or e-Business Level (eBusiness Standard).

For the components which are already deployed or implemented, their associated models are most of the time not available. Solution vendors will not provide them with the product, as it is internal know-how. For in house applications, documentations and design artifacts are

always lost or not well managed. In addition, duration of life of design platforms and related technologies is shorter than duration of life of the produced software itself, leading to important migration issues. Finally, the different produced models can rely on heterogeneous and incompatible design technologies. It is consequently difficult designers to reuse components, as they have to perform retro-design of the components they have to integrate.
When provided, the models should be made available through different coherent representations which can be used on the platforms used during the different phases of the lifecycle of applicative components (e.g. component libraries for designers – models -, developers – code or integrator – published interfaces, etc.). Existing frameworks are most of the time not considering impact of management and availability of the design and development artifacts related to reused components.
Finally, as the innovation process and research encourage development of disruptive solutions and new paradigm, integration of legacy systems is not considered. The new paradigms have important impact on the used formalism for conceptualization or design, leading to fast deprecation and obsolescence of the platforms and formalisms used to produce model components. Continuous migrations and evolutions are consequently to be considered. If not related to efficient governance at enterprise level (controlled urbanism) at community level (Governance of eBusiness for a community), it leads to coexisting sets of heterogeneous and incompatible legacy artifacts.

5.8 - Graphical and textual languages

Important issue exists due to insufficient consideration concerning association of graphical formal languages and associated textual formal languages when dealing with interoperability.

5.8.1 - Formal textual and graphical languages

Numerous formal graphical languages have been developed in order to be able to easily describe complex systems dedicated to human being: conceptual graphs, Unified Modeling Language, EXPRESS-G…They were initially computer usage independent, it means dedicated to drawing using paper.
First automation was done by providing drawing aided tools, allowing replacing the paper and the pen. But when willing to use produced drawing for other automation as validity checking or code generation, it appears that graphical languages are not computable. It was the motivation, in the 80/90's, for ISO STEP technology, to select EXPRESS, a textual and computable language, as the way to produce normative specification, while the graphical language, EXPRESS-G, was provided for informative parts of application protocols, and aiming to help human being to navigate visually within the model.
Unified Modeling Language was initially a diagramming language. The issue of Design Model interchange came later for Software engineering domain compare to manufacturing domain. XML Interchange Model was produced only in June 2000 and was based on XML (with DTD). But this standardized serialization is only concerning the model, and not the graphical representation of the diagrams. It was corrected through the Diagram Interchange specification, finally released (V1.0) in April 2006.
Process Modeling Notation is a graphical language, and it can consequently be exchanged between modelers. OMG provides only a mapping to the Business Process Execution Language, which is a programming language. Workflow Management Coalition aligned the second version of XML Process Definition Language (exchange format for workflow models) with BPMN, mixing drawing concepts with modeling concepts, leading to some confusion concerning the way to use or to interpret a pool, a lane or an actor.

This three examples are showing that the way to deal simultaneously with textual (serialization) and graphical (notation) representation is not obvious, and may lead to complex and inefficient standards when willing to exchange information within eBusiness, that can be shared by people and by automate, in particular when the conceptual models are not aligned.

When dealing with interoperability, important work was done by the ISO STEP community in order to build the STEP technical framework. (Schenck 1994) already characterized differences between graphical representations and lexical representations for interoperability purpose, justifying the choice of EXPRESS for normative specification, while EXPRESS-G was only used for information. Considering the extensive usage of graphical representation within some standardization communities, without alignment with a lexical representation of reference, we can consider that we are facing a problem of maturity of these communities concerning interoperability of applications.

5.8.2 - Physical representation of a document

A similar issue exists with physical representation of a document dedicated to people, as the precise way it is printed on a physical support, being paper or screen, should be the same. Standards such as PDF or XML-FO are dealing with such physical representation, dedicated to human being. When the way elements of a document are position in space produce information, it is absolutely required. WEB navigators or different versions of office suite are most of the time not able to ensure to produce the same representation, which consequently lost information.
As it is mandatory, within the context of eBusiness collaboration implying enterprise application, to support exchange of information between software but also between people, clear coverage of coherent exchange of serialized information and their physical representation is absolutely required.

5.8.3 – Maturity of standardization community and enterprises

Nor the standardization communities nor the users are still really mature on this topic. The late release of Diagram Interchange specification or the mixing of drawing and modeling within the last version of XML Process Description Language demonstrate it. For the user, an important problem comes from the fact he just can't now what is exactly the software internal representation of what is displayed on his screen. So for him, a bitmap picture, a vector graphic or a drawn model have exactly the same value and are interpreted the same way. But automation capabilities are highly different. A picture or a vector graphic can't be validated by a computer from a semantic point of view. All the semantic computable information are using formalism dedicated to drawing automates. Within enterprise, due to this confusion, drawing (e.g. Microsoft Powerpoint) or diagramming tools (e.g. Microsoft VISIO) are widely used to produce only contemplative models. Contemplative models are not computer-understandable models (Bezivin 2002). They can't be used by modeling software product, being for validation, checking, simulation or code generation. Even for advanced research on enterprise modeling language, such as POEM[24] or ArchiMate[25], graphical representations, such as Microsoft Visio stencils or UML diagrams, are provided but not computable

[24] POEM: stands for Process Oriented Enterprise Modeling. It is a descriptive process oriented organizational/enterprise modeling methodology, based on reuse of the Unified Modeling Language and the Business Process Modeling Notation. It is developed by the Institute of Informatics of the University of Maribor. Methodology and Visio stencils can be found at http://bpmnpop.sourceforge.net/

[25] ArchiMate: a project and a language standardized by Open Group. ArchiMate stands for "An integrated architectural approach for the description and visualization of different business domains". Language specification for version 1.0 can be accessed at http://www.opengroup.org/bookstore/catalog/c091.htm . Information related to the project are available at http://www.archimate.org

representations. So contemplative models are still widely used and developed. They are preferred to computer-understandable models. As a result, an important number of models will be produced which will not be useable for automated checking, validation, simulation, code generation or model driven engineering.

5.8.4 - Growing interest and importance of visualization techniques

Investigating on how to solve such issue led to collect and study different approaches related to interoperability, which are considering graphical representation. (Ray 2006) makes a clear distinction between languages for people and languages for machines. He identified several standardized graphical languages which should be mapped to Common Logic. (Bezivin 2002) makes distinction between human readable models (contemplation) and computer-understandable models (code production) models. Such approach could be extended to executable models, which are closer to development languages. For code production and execution, the models must rely on textual formalism, but advanced visualization techniques can be used to support developers involved in complex applications, as illustrated by application of advanced visualization techniques developed by Chisel group[26] to java programming (Creole[27]).
Such visual exploration becomes important due to growing complexity of information to share and exchange within the extended and virtual enterprise. It is illustrated by work performed by (Tricot, 2006) around the "semantic cartography" emerging domain. In addition, with modeling tools and standardized serialization, it is possible now for software product to produce interactive graphical representations, which are no more static, and which provide an important added value when willing to navigate and manipulate visual and graphical representation of complex information or systems.
Some freely available tools illustrate it, such as Freemind[28], Shrimp[29], Jambalaya[30] or Yed[31], allowing new cognitive way to explore complex information. (Mueller, 2009) is studying interactive visualization of semantic graphs.

5.8.5 - Identifying difference between graphs and models for end user

The frontier between drawing tools which are not capturing the semantic of the drawing constructs and those which are associating semantic which can be interpreted by the machine is clear, but not always identified by the user. For example, Freemind only allows drawing mindmaps, but no semantic is attached to the nodes and to the arrows. Protégé Jambalaya plug-in is drawing graphs from OWL models, i.e. all the nodes and arrows are typed before to create a graphical representation of the semantic graph. Ambiguity here comes from the fact that representations for human can't be used without a computer, when willing to take advantage of interactions and dynamic aspects of the representation. Such interactions and dynamic aspects are even providing more information to the users than classical frozen static

[26] Chisel Group: Computer Human Interaction & Software Engineering Lab, http://www.thechiselgroup.com
[27] Creole: Eclipse plug-in allowing to explore Java code visually allowing to see its structure and thelinks (calls, access, etc) between its different pieces. Available at http://www.thechiselgroup.com/creole
[28] Freemind: open source mindmap editor available at http://freemind.sourceforge.net
[29] Shrimp: an application and a technique developed by the Chisel group, for visualizing and exploiring software architecture and any other information space. SHriMP stands for Simple Hierarchical Multi-Perspective. It is available at http://www.thechiselgroup.com/shrimp
[30] Jambalaya: Shrimp plug-in for Protégé, available with Protégé download at http://www.protege.stanford.edu
[31] yED: graph editor that can be used to quickly andeffecively generatedrawings and to apply automatic layouts to a range of different diagrams and networks. Edited by yWorks, it is available as a free download with unrestricted functionality at http://www.yworks.com

representation when printing on paper. But does used application interpret literal information or not?

5.8.6 - Conclusion

Alignment of textual and graphical representation for a same set of standardized modeling concepts is today an issue when willing to insure exchange of information and interoperability. As conceptual alignment is most of the time not insured and as drawing constructs and modeling constructs are not always separated, it leads to some confusion and to real difficulties to understand how to use formal models. Finally, it is not always identified that graphical languages, even if formal, are not accurate when willing interpretation by a machine.

5.9 - Federation of systems

A complex problem is the federation of managed resources being controlled and filtered by several systems. When willing to interconnect several systems managing their own set of resources, which are to be interrelated, numerous issues exist. It is for example the case of Product Management Systems federation. In such a case, we don't have a single server insuring logic of cooperation of the different partners. We have several local private systems, which are implied in collaboration transversal processes, but with change managed and under the control of the systems. How to ensure transfer or sharing between these systems, knowing that we are not only transferring data, but also control and property on the data?
In addition, the viewpoint on the data is changed during the transfer, implying some transformations. Other issue is segregation of data: how to ensure data privacy without keeping it inside our own private systems? Such consideration is very important within industry, when willing to protect intellectual property.
As a consequence, poor information exchange and sharing is done between partners managing their information with such systems. When performed, it is done by hand. It is consequently error prone and resource consuming. Efficient change percolation is very difficult to ensure coherently between the different concerned systems.

5.10 - Data exchange: not only translation but transposition

Most of the time, the work related on data exchange is focused on establishing mappings and associated translations, working on the concept of "equivalence" between classes. But when exchanging data between management systems, some change of context may occur between the source system and target system leading to a transposition of the same concerned thing.
For example, an internal employee within the enterprise A will be transposed as an external employee within the enterprise B. It will always refer the same person, but we can't say that class of internal employee and class of external employee are two equivalent classes.
So a transposition is to be performed, leading to manage links between the information representations of the same think within the different systems. Note that if the two applications are independent and don't know each other, this kind of transposition relationship between classes is to be managed outside the two applications. Targeted class will not be formalized inside the source system.
Such transposition may become very complex when dealing with coherency between several internal management systems of collaborating enterprises.
It leads to establish coherent links between multiple identifiers, types and decompositions which are managed independently for same things which are represented in different contexts.

For a system, according the considered viewpoints and phases of the lifecycle, the different considered and use breakdowns, for decomposition or for composition, are also different.
None well established transpositions are leading to non interoperability of used applications.

5.11 - Conclusion: statement of the problem

Willing to address effective interoperability of technical enterprise applications, I firstly characterized the level of interoperability to achieve for the different concerned systems, relying on existing relevant models of reference used by the research and standardization communities. I then characterized the different kind of interoperability, with the associated more relevant standards and interoperability frameworks.
As a starting point for the thesis, I considered that a holistic federated approach combining different domains and their standards is an absolute prerequisite to achieve effective interoperability of technical enterprise applications supporting eBusiness collaboration within extended and virtual enterprises. This starting point is the same than for the research community dealing with Interoperability of Enterprise Application.
Such approaches have been studied within the research and standardization community. Within the scope of the thesis, these approaches have been studied, analyzed, with in particular comparison with several practical experimentation and usage.
As a result, identification of needs, brakes and issues that are not addressed was performed, allowing producing a list[32] of twenty one interoperability brakes and issues (summarized by mindmap on Figure 41).
Currently proposed frameworks are not accurate to achieve pragmatic interoperability with current business context. Considering the listing of interoperability brakes, reasons for non interoperability are numerous. They may come from paradigms and methods used to specified, design, develop, support, change and use the applications.

In one hand, it is difficult to consider that interoperability of enterprise applications can be easily obtained and continuously maintained within always changing environments, without an important consumption of resources. It is also difficult to believe that new emerging paradigm or technologies will resolve the identified issues, without creating their own boundaries and excluding important part of the legacy applications.
In the other hand, systematic usage of computers and digital artifacts within the enterprise creates a difficult situation: if bringing a lot of advantages allowing managing more complex systems, faster and with a better quality, it also creates a border effect. Amount of heterogeneous and incompatible existing informational and computational resources is growing and growing, leading to difficulties when willing to aggregate (information) or interconnect (applications) these resources when willing to support efficiently cross-organizational cooperation, taking still advantage of computers.
The main objective of my thesis is to define a set of principles allowing building pragmatic interoperability framework. The approach is holistic and based on federated usage of the different identified relevant domains and disciplines and their associated reference models and standards. It aims to master the identified border effect. From the analysis, it is consider that interoperability of enterprise applications within the multiple environments they are belonging to is a quality which is to be maintained over the time, and which require important resources to be maintained. Pragmatic interoperability means that cost of interoperability maintenance should be reasonable with regards to expected return on investment (if it can be calculated), industrial risks limitation (durability of computational resources and produced knowledge,

[32] as required by NEHTA Interoperability Maturity Model to reach level three of maturity for a community

over complexity of the information system to maintain, etc) and business opportunities taking into account globalization and required competitiveness of enterprises in such a context.

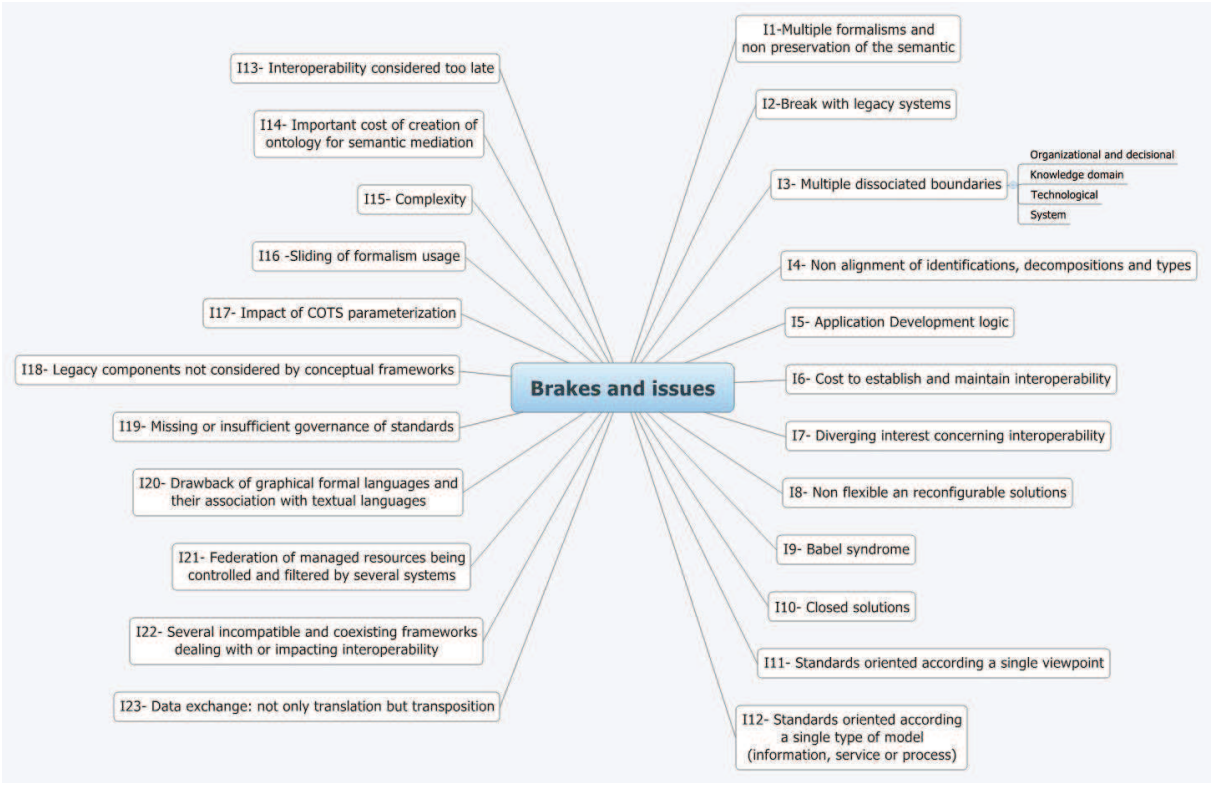


Figure 41: Brakes and Issues Mindmap

Part2 – Establishment of pragmatic Framework for interoperability of enterprise applications

In this part, I firstly describe the set of principles to be applied by a community willing to build an interoperability framework, which address the interoperability issues identified. The target is achievement of pragmatic interoperability when dealing with enterprise applications within a given networked organizations.
The principles are built from analysis of the state of the art, but also from analysis of piloting and demonstration activities. The principles are concerning systematic usage of relevant open standards, consideration of legacy, multi-disciplinary and holistic approaches, generic architectures of reference for eBusiness applications, federation of models defined using different formalisms, languages and paradigms, semantic preservation for all the data shared and exchange between all the actors implied within the lifecycle of an application and associated environments, and finally governance of standards.

I secondly develop an innovative concept; the "extended hyper model", in order to address semantic preservation between information systems and applications implied in collaboration. Based on the consideration that the different implied applications and information systems are in fact places with multiple representations of same things, using heterogeneous languages, paradigms and formalisms, "extended hypermodels" aims to address unresolved interoperability issues described in Chapter 5, in particular the semantic and data lost when applying current approaches for interoperability.

Chapter 6: Principles for interoperability

6.1 - Introduction

After describing within the general introduction the emerging and growing needs concerning interoperability of enterprise applications, I made within the first part of the thesis a complete state of the art allowing referencing main interoperability models and classifying the different types of interoperability. Relying on NEHTA Interoperability Maturity Model, I also referenced the different relevant legacy standards and interoperability frameworks related to industrial sharing, taking advantage of the numerous research projects I was involved in the previous twelve years. It allowed me to identify a set of interoperability issues and brakes which are not currently addressed by research community, if willing to achieve pragmatic interoperability of technical enterprise applications within a networked organization. Considering it is useless to propose one more framework, I firstly produce a set of eleven principles to apply if willing to insure such interoperability. In order to respond to the set of identified issues which explain difficulties to obtain an effective interoperability of technical enterprise applications within an eBusiness environment, I defined a set of principles which could support establishment of a framework to support holistic model driven approach based on "projection" of enterprise models on service oriented execution platforms.
One of the most important principles is to avoid any disruptive innovation, without considering reuse of legacy applications, technologies, standards, etc. It leads to systematic reuse of standards based software components available as commodities on the web, and systematic reuse of formal business models of reference, and to include them systematically in a model driven engineering approach leading to project models on execution platform.
With iterative projects, bidirectional model transformations are required due to continuous evolution of application environments. Different relevant architectures for reference are to be used and aggregated for execution platforms interconnected with other modeling and development environments. As we are dealing with complex systems of systems, graphical representations are mandatory, but should be aligned conceptually with the formal language dedicated to automation. As it is a holistic approach taking into account lifecycle of the application and of its components, interoperability is to be managed according interoperability actors and stakeholders, which are numerous and should be identified.
When possible, the reference models should be based on generic standardized meta-models. Appropriate generic process for establishment of collaborative infrastructures should be established and shared. Finally, as standards are key components, their lifecycle and evolutions are to be mastered by mean of appropriate governance.
The next chapter is numbering and detailing these different principles, establishing how their usage can partially respond to some of the interoperability issues and brakes identified within Part 2.

6.2 –Definition of the principles

N°	Designation
P1	No disruptive innovation and reuse of legacy coming from various communities

It is identified within that disruptive innovation leads to non interoperability. Disruptive innovation and new paradigms lead to break with legacy systems (Table11-I2), by providing new incompatible applicative systems and continuous replacement of legacy systems components. Implied migrations are most of the time very expensive and imply to stop the operations.
Disruptive innovation, but also the associated belief that one single discipline or community will bring a complete solution, should be avoided. It is important, in particular due to the fact that communities are most of the time focusing on a single viewpoint (Table11-I11) or on one particular type of model (Table11-I12). Single viewpoint focus is illustrated by the relevant standards and frameworks described within the state of the art, which are addressing a particular viewpoint or a particular type of interoperability.
In addition, disruptive innovation is going against the idea to federate composite set of legacy partial solutions (c.f. P1).
Finally, application of this principle gives the reason why the state of the art is detailing a set of standards and frameworks, which will be considered as mandatory components of global interoperability solution.

N°	Designation
P2	Systematic reuse of software components based on open standards and available as commodities on the web

This principle concerns systematic reuse and joint usage of components of the shelves based on standardized open formal specifications. It derives from principle P1, with special focus on technologies and components with are the less difficult to share and to access to.
The standardized open formal specifications are defining complementary expected functions and associated formal services, when willing to implement enterprise applications. Each specification provides a relevant model of reference for each function, and should be associated with a set of existing industrial and robust implementations, being commercial or commodities on the web (i.e. robust and standard base, freely available on the WEB, as for example relational database systems based on relational database standards (SQL and ODBC) or web server application implementing standardized protocols (such as HTTP, FTP, etc.).
A set of generic architecture of reference, based on expected functions and architectural pattern of reference, should provide a generic framework. This generic framework can be validated at specification level. Then, each generic component should be realized by a concrete robust component, implementing the specification, and it should be ensure that integration of concrete components have the expected behavior.
Applying such principle should solve issue related to usage of close solutions (Table11-I10) and of interoperability considered too late (Table11-I13), as selected components are design phase are already based on open standard and providing access to the source through appropriate licenses.

N°	Designation
P3	Systematic reuse of formal business model of reference

This principle concerns systematic reuse and joint usage of relevant formal business models of reference when existing (e.g. STEP application protocols), despite they can be developed using heterogeneous languages, for different purposes and according to different paradigms. It will prevail to development of one specific and proprietary formal business application model per closed software product (Table11-I5). It will also avoid the multiplication of formal model for a same domain. It will allow protecting previous investment made by industry and communities in order to formalize their business, and to avoid to involve again experts to provide new and redundant formalizations, and to obtain a consensus. So it should partially address ontology development cost issue (Table11-I14).
Nevertheless, due to Babel syndrome (Table11-I9), several redundant specific models of reference will necessarily be produced and will co-exist, introducing some incoherencies.
This incoherency will have to be managed. It is one of the goals of "extended hypermodel" I developed and which is defined within Chapter 7.

N°	Designation
P4	Systematic usage of model driven engineering to project business logic on execution platform

This principle concerns the application of approach promoted by ATHENA program, and which constitutes a starting point of my thesis. A flexible enterprise application aiming to be easily federated within a networked organization should be obtained through joint usage of enterprise modeling and model driven engineering. Model driven engineering allows generation of business components to be deployed on service oriented execution platforms.
I consider it is a key enabling principle, as applying it should allow, through an industrial approach, to reduce the cost when establishing and maintaining continuous interoperability (Table11-I6). It should make it possible to deal with federation of multiple formalisms, when combining different complex and huge models (Table11-I1). It will also imply usage of formal models, which are formalizing a way allowing processing and interpretation by computers with the accurate level of details, allowing computer aided interconnection.

N°	Designation
P5	Bidirectional model transformations to support iterative and continuous evolution

It should be possible for any iterative process allowing to aggregate legacy components at different stage of their lifecycle, and considering that we can or not use the intermediate artifacts the models are, to establish bidirectional transformation allowing using different available legacy component of the shelves (Table11-I18). Usage of a set of models and formalism allowing bidirectional transformation should address the issue of semantic preservation and data lost.
An example of such transformation could be for example, from interface description of a deployed component in Web Servicer Description Language, the production of an annotated Platform Independent Model describing business services, and the reverse.

N°	Designation
P6	Generic Architecture of reference for interconnected execution platforms

Generic architectures of reference should be established in order to contextualize and to provide the execution platforms where the business logic will be projected from modeling and development platforms.
These architectures should be defined considering the different important viewpoints and scales. I consider it should include at least personal centric viewpoint (user or actor), team centric viewpoint, enterprise centric viewpoint and network of enterprises centric viewpoint.
The generic architectures of reference should also federate frameworks used for single terminal objects, components (composition of objects with on program), applicative components (set of components of a unit of management from user point of view), software product components (set of components of a unit of management from vendor point of view), enterprise information system (set of applicative components with an enterprise) and network of enterprise information system (set of interconnected web application).
In order to illustrate and to validate the approach I am developing within the context Product Lifecycle Management, such set of generic architectures of reference are proposed in the third part of the thesis report. Derived from existing reference models and standard based interoperability frameworks described and studied within the state of the art, this set is also aiming to respond to business objectives identified within the introduction.
The set of generic architectures of reference will allow identifying the applicable formalisms to use, for which preservation of the semantic (Table11-I1) will have to be addressed, with regards of the concrete implementations of the interconnected execution platforms. It will allow to formalized required interfaces between existing boundaries (Table11-I3) and provide methods adapted according the application development logic (Table11-I5). It will also allow to master cost to establish interoperability (Table11-I6) with industrialization of open (Table11-I10) flexible and reconfigurable (Table11-I8) application development, with easier management of the complexity. It will also allow to addressing impact of Commercial Off-The-Shelves parameterization (Table11-I17).
Finally, it will structure the environment where “extended hypermodels” will be created and used.

N°	Designation
P7	Establishment of clear distinction and complementarities for used graphical and textual formalizations.

Clear distinction and complementarities should be established for used modeling graphical formalisms (dedicated to human) and textual formalisms (dedicated to machines). It is mandatory when willing to deal with semantic preservation and coherency between the different used representations.
It will allow solving limitation of graphical language for usage by computers what described within previous part (Table11-I20).
In addition, it is consider that modern modeling language should be associated to visual representation(s), in order to deal with complex models involved in interoperability of enterprise applications (object of semantic cartography). However it should be based on aligned conceptual modeling constructs

N°	Designation
P8	Interoperability to be managed according to interoperability actors and stakeholders all along application lifecycle management

This principle concerns systematic establishment of cartography of actors of interoperability (and non interoperability) all along the lifecycle of an application, in order to be able to propose a holistic approach and an accurate methodology to ensure interoperability.
Let's consider the following: we can consider that operational applications don't interoperate when there is a failure of communication. Such failure can comes from any decision or action occurring during the different phases of the life cycle of the involved applications and their communication infrastructures. Here are several examples:

- If interoperability is not elicited or identified as a prior requirement at the specification phase (as for security) of an application, the final solution will probably not be able to interoperate with other applications.
- The applications are deployed within the enterprise information system. If the enterprise information system is not well structured and documented according controlled urbanism rules, it is very difficult to defined constraints in term of communication protocols that will have to be supported, being technical or business.
- If the designer does not provide a flexible and open architecture for a solution, the resulting application is not be agile and managing change of environment of the application will cost.
- Development and testing is particularly important to ensure good software component with compliance with the standards.

These different examples illustrate different kind of actors and stakeholders of interoperability, which have to be identified and contextualized according concerned processes and systems.
Listing interoperability actors and stakeholders all along the lifecycle of enterprise applications should allow to identify and manage risks related to break with legacy systems (Table11-I2), the different decisional and activity boundaries (Table11-I3), the diverging interest concerning interoperability (Table11-I9), to put in evidence the Babel syndrome and to try to solve it, and finally to put in evidence danger of close solutions (Table11-I10). It finally allows considering interoperability at the earlier stage (Table11-I13).
Projected on graphical representation of the different generic models of reference, including processes maps or representations of architectures, it will provide cartographies which will be efficient tools for communication and knowledge sharing concerning a complex system.

N°	Designation
P9	Generic standard based meta-model of reference

It should be clearly established which are the different concerned systems, and some generic business meta-models of reference should be provided to describe and rely them, based on existing models of reference (enterprise internal), standards (eBusiness) and best practices.
The meta-models of reference and the standards are identified generic models, which are available within a mature organization or community, which have a real added value and are reflecting reach enough generic or specific consensual models which can be used for different applications.
Here meta-model is understood as a M2 level model within the MDA architecture. For example, it corresponds to a STEP application protocol, the internal model of a software product, a vocabulary for electronic document, etc.

The state of the art within this report reflects the work done to identify and assess standards, models of reference and methods for the different concerned communities: information and communication technologies (ICT) providers, being products, components, standards, methods or innovative paradigms. In order to be useful, vertical domain(s) and discipline(s) should also be chosen as they are defining the business semantic. It is the reason why interoperability of enterprise application is studied for technical information (i.e. product development within manufacturing context).
Here numerous standards are produced. But the principle can be used for other communities such as energy, health, telecommunication, finance, etc.
Application of this principle should decrease cost for establishment and maintenance of interoperability (Table11-I6) and reduce the complexity (Table11-I15) by reducing the number of specific local model of reference.

N°	Designation
P10	Process for establishment of collaboration infrastructure

Collaboration infrastructure required to be produced according a well defined process, knowing that it is disconnected from projects which will use it.
It should be driven by a community, considering the establishment of the collaboration infrastructure and its support, but also how to join the collaborative space, leave the collaborative space and collaborating through the collaboration space.
It should allow to deal with dissociated boundaries (Table11-I3), select appropriate development logic (Table11-I5) and to work on the basis of common interest (Table11-I7)

N°	Designation
P11	Governance of relevant standards

The evolution of interrelated standards should be driven by strategy and interest of the whole community which benefits of it, not by only few stakeholders or a single stakeholder.
It is why governance of the standards of reference used for collaboration space is important.
It should contribute to address issues such as cost to establish continuous interoperability (Table11-I6), diverging interest concerning interoperability (Table11-I7), non flexible and reconfigurable solutions (Table11-I8) Babel syndrome (Table11-I9), closed solutions (Table11-I10), standards oriented according a single viewpoint (Table11-I11), interoperability considered too late (Table11-I13), important cost of creation of ontology for semantic mediation (Table11-I14) and missing or insufficient governance of standards (Table11-I19)

6.3- Recapitulation of the principles

In order to respond to the set of identified issues which explain difficulty to obtain an effective interoperability of technical enterprise applications within an eBusiness environment, I defined a set of principles which could support establishment of a framework to support holistic model driven approach based on “projection” of enterprise models on service oriented execution platforms.
One of the most important principles is to avoid any disruptive innovation, without considering reuse of legacy applications, technologies, standards, etc. It leads to systematic reuse of standards based software components available as commodities on the web, and systematic reuse of formal business models of reference, and to include them systematically in a model driven engineering approach leading to project models on execution platform. With iterative projects, bidirectional model transformations are required due to continuous

evolution of application environments. Different relevant architectures for reference are to be used and aggregated for execution platforms interconnected with other modeling and development environments. As we are dealing with complex systems of systems, graphical representations are mandatory, but should be aligned conceptually with the formal language dedicated to automation. As it is a holistic approach taking into account lifecycle of the application and of its components, interoperability is to be managed according interoperability actors and stakeholders, which are numerous and should be identified. When possible, the reference models should be based on generic standardized meta-models. Appropriate generic process for establishment of collaborative infrastructures should be established and shared. Finally, as standards are key components, their lifecycle and evolutions are to be mastered by mean of appropriate governance.

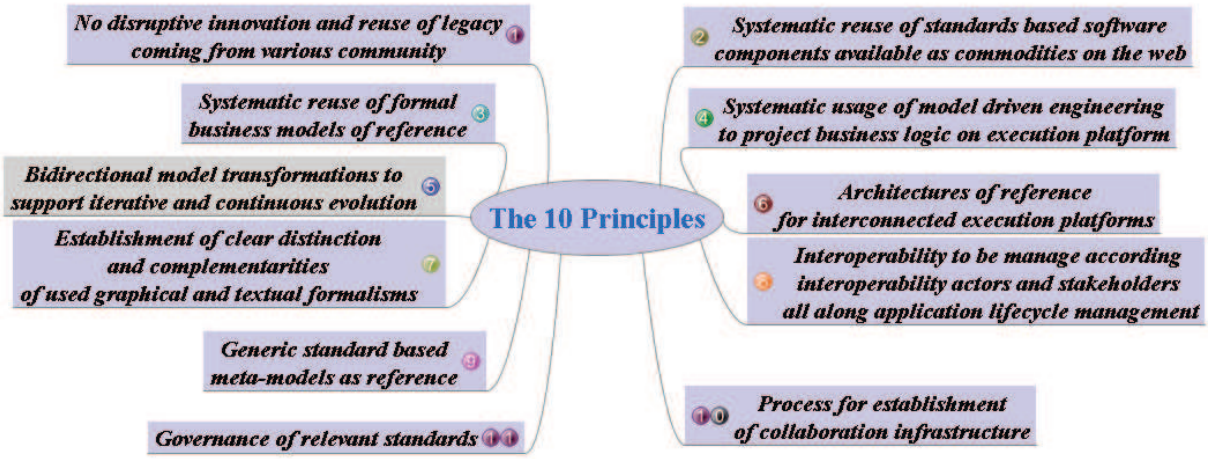


Figure 42: Interoperability principles mindmap

Applying the different principles doesn't solve all the interoperability and issues, and may lead to new problems to solve. In particular semantic preservation is becoming more important, as it is a key interoperability enabler. It is also becoming more difficult to solve, as numerous digital artifacts are considered which are subject to be produced using computer aided solutions. Finally, information which is encoded within these artifacts should be exchangeable and shareable by numerous persons and systems. So we can say that applying the proposed principles, semantic preservation must be ensured between used environments (conceptualization, design, development, and execution), produced artifacts and information repositories.

It is today very difficult to ensure with existing standards and interoperability frameworks. I propose in Chapter 7 an innovative approach, related to usage of any interoperability framework defined on the basis of the previously described principle, which I called "Extended hyper-model for interoperability of enterprise applications within virtual organization".

Chapter 7: Extended hyper-models: an innovative approach for semantic preservation

The objective is to find a way of preserving business semantic between the models used by the different platforms, being conceptualization, design, development or execution. It is also to preserve the business semantic between the different applicative components used inside a single application, such as database, application server or presentation server, or several federated applications. It is not possible to obtain pragmatic interoperability by applying previously defined principles, if the semantic preservation is not insured.

As a constraint, it should make it possible to use jointly legacy standardized languages and interoperability frameworks, being at conceptualization, modeling, design, development or execution phases. In addition, when using a model driven approach, it should be possible to trace, to link and to make coherent the different representations of used business concepts and objects, between the different artifacts produced during lifecycle of an application. Usual bindings and mappings, proposed by the different communities, are far from being sufficient: they don't avoid loss of information; they are restricted by technological silos, incompatible paradigms, organizational boundaries, etc.
This chapter presents the concept of "extended hyper model" I developed during the thesis, in order to enhance pragmatic interoperability based on joint usage of set of standards, as well as federation of applications within networked enterprises.

7.1 - Origins of the concept of extended hyper models

7.1.1 – A same concept with several representation

The starting point for extended hyper models is to consider that, for a given business concept, it does not matter for the user if it is formalized with such or such formal language. We have simply different representations of a same concept, using heterogeneous languages which are used with different intentions: data exchange, reasoning, design, programming, etc.

For example, considering the "person" business concept, it could be formalizes within different environments based on heterogeneous languages, such as UML, OWL, XML or EXPRESS. The modeling constructs will be UML class, OWL class, XML entity or EXPRESS Entity, and eventually different names and identifiers could be used. But for the users, it remains a person.

The idea is then to consider that it should be possible to capture the fact that this concept, modeled using such modeling construct within an environment, is modeled using such other modeling construct within other environment. In addition, as it still remains the same concept, a "person", used formalism should be as transparent as possible concerning the business semantic, as interpreted by people.

Referring (Schenck 1994), which provides clear distinction between conceptual information models and concrete instantiation models, the idea is to be consider that within a model driven approach, concrete models are those used for conceptualization, design, development and execution. The glue remains the conceptual model, which will ensure the semantic preservation.

7.1.2 -Hypermodel

The term "hypermodel" is used within the work described by (Carlson 2006), and which lead to a software freely available on the web, "Hypermodel"[33]. This tool allows working on a model formalized as an XML schema, within an UML environment, and possible annotation using OWL. A same concept is "multi – represented": it can be consider as an XML entity or a UML class. It is represented as an UML class within dynamically created UML diagrams.

The term "hypermodel" is also used in research related to model based learning (GOBERT, 2005) (Horwitz 1995) and is defined as combination of aspects of open-ended exploratory tools with content-specific text and multimedia. "Hypermodel" tool is mixing heterogeneous interactive representations both textual and visual of a same subject. So a hyper-model can be considered as a multi-representation of the same model, using different formalism and medias, but preserving the semantic.

"Hypermodel" as defined by (Carlson 2006) is limited to only three languages, which are all dealing with entities and informational viewpoint. It was not developed in order to address interoperability of enterprise applications. As my intend was to extend it to other languages and modeling constructs (services, processes), in order to differentiate the targeted usage, I called it "Extended Hypermodel" for interoperability.

7.2 – "Extended" hypermodel for interoperability

7.2.1 – The basic concepts and principles

A **ground** is a concrete environment where a model is formalized. It is constituted by a set of configured software components, which ensure conformance with a given standard (versioned) providing one formalism we want to consider within the hypermodel.

A **landscape** is a set of grounds. The landscape of a given standard is the set of available grounds enabling usage of this standard (versioned), and with appropriate conformance. The landscape of a hypermodel is the set of environments, related to the standardized languages considered by the hypermodel, which are allowing traveling of hypermodels without data loss.

A **model**[34] of a system is a description or specification of that system and its environment for a certain purpose. A model is often presented as a combination of drawing and text. The text may be in a modeling language or in a natural language.

A **hyper-graph**[35] is "a generalization of a graph, where edges can connect any number of vertices. Formally, a hyper-graph H is a pair $H = (X,E)$ where X is a set of elements, called *nodes* or *vertices*, and E is a set of non-empty subsets of X called *hyper-edges* or *links*…
While graph edges are pairs of nodes, hyper-edges are arbitrary sets of nodes, and can therefore contain an arbitrary number of nodes. However, it is often useful to study hyper-graphs where all hyper-edges have the same cardinality: a k-*uniform hyper-graph* is a

[33] Available at http://xmlmodeling.com/hypermodel
[34] The definition is the one provided within the MDA Guide V1.0.1
[35] The hypergarph definition is extracted from Wikipedia

hyper-graph such that all its hyper-edges have size k. (In other words, it is a collection of sets of size k.) So a 2-uniform hyper-graph is a graph, a 3-uniform hyper-graph is a collection of triples, and so on…
Unlike graphs, hyper-graphs are difficult to draw on paper, so they tend to be studied using the nomenclature of set theory rather than the more pictorial descriptions (like 'trees','forests' and 'cycles') of graph theory."

A **hypermodel** (H_M) is a hyper-graph, providing a multi-representation of a model using the different standardized modeling language implied within a community governing interoperability of its applications.

X_M is the set of concrete constructs, specific to the formalisms aggregated by the hypermodel, allowing to model a concept or to represent a real thing.
Each business concepts and each real thing is represented by a subset of X_M, which is a hyper-edge of the hypermodel. The subset contains all the nodes representing the concept or the real thing.

Figure 43 is describing different hyper-edges based on generic descriptive business concepts (associated to a color), derived from ontology modeling concepts: class, property and individual. Each hyper-edge is grouping nodes which are used to represent the business concepts or real thing for a given formalism (associated to a form).

A **generic descriptive business concept** is a concept that is suited for simple description of the real world by persons which are not computer scientists.

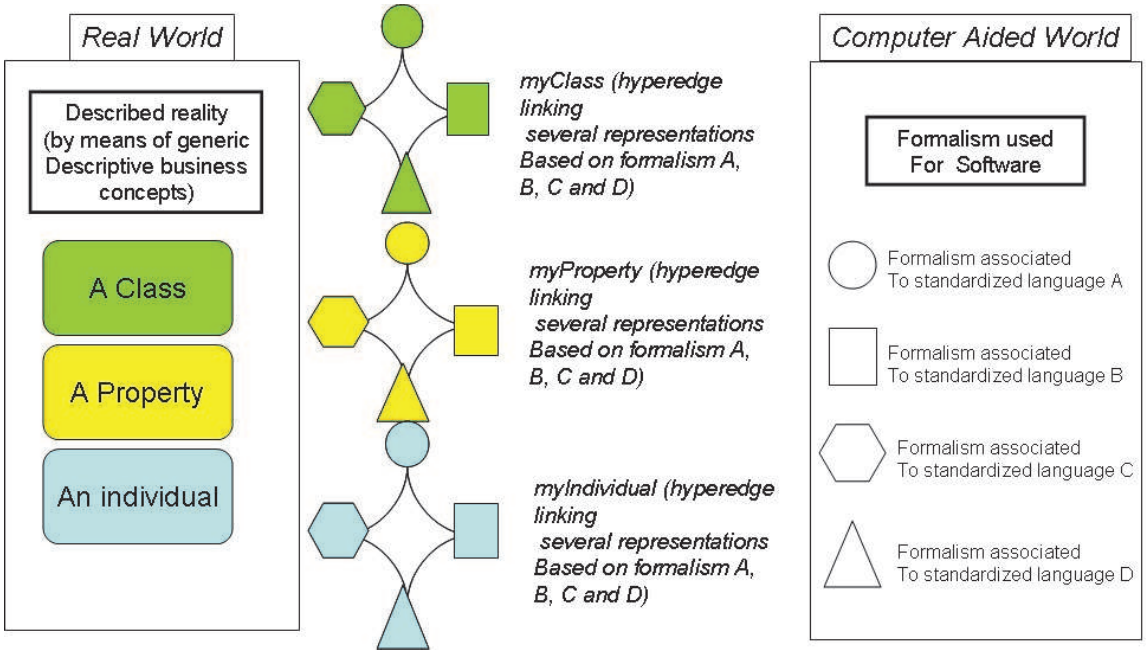


Figure 43: basic hyper-edge of an extended hypermodel

It is expected that it will be possible to formalize concretely an extended hypermodel within several grounds, and to transfer an extended hypermodel from one ground to another without information loss, through accurate set of transformation based on standardized data exchange and transformation. It is what I call hypermodel "**travelling**", i.e. the ability to change place remaining the same, like illustrated by Figure 44.

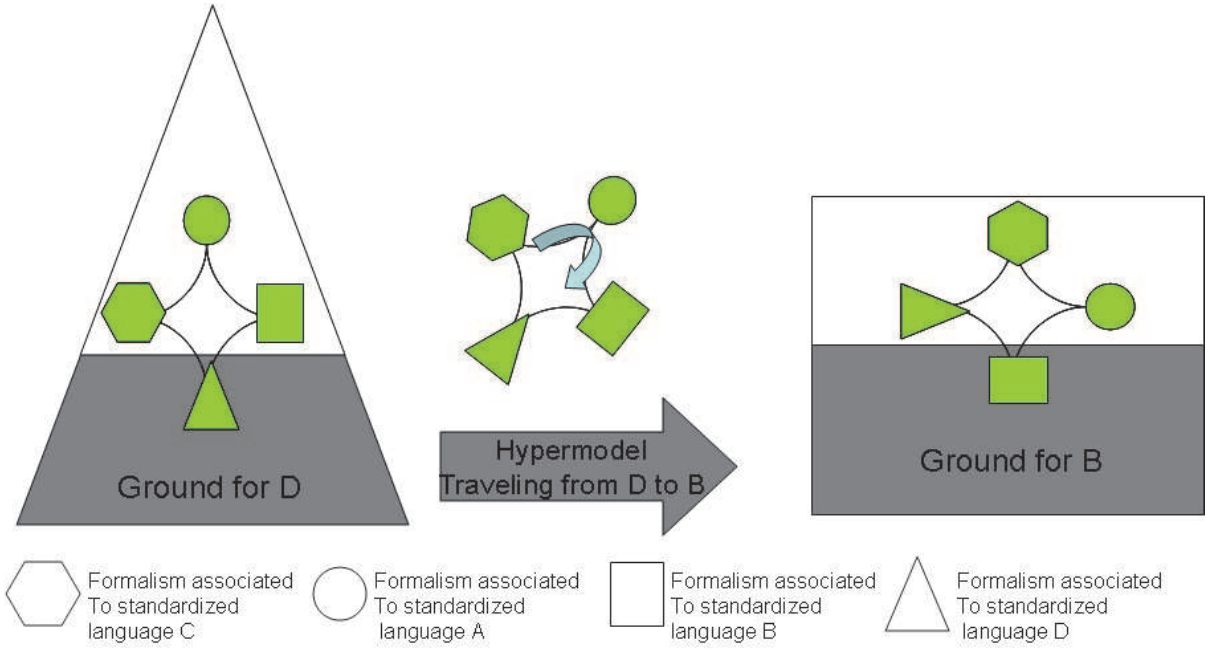


Figure 44: Hypermodel traveling between grounds

As willing to avoid development of a new modeling language for extended hypermodels and associated grounds, legacy platforms belonging to the several federated grounds should be able to process them, being for edition, validation, completion or other automated usage. In other words, it should be possible for an extended hypermodel to be multi-represented within the grounds constituting the landscape of the hypermodel.

7.2.2 -Chosen generic descriptive business concepts

A default set of generic abstract modeling constructs is proposed, based on core upper concepts defined by the semantic web community and the Web Ontology Language. Several reasons exist for such a choice. Firstly, we may have to aggregate a posteriori heterogeneous resources, and it is addressed by the semantic community. Secondly, we need a formal language allowing describing what exists. It is the goal of ontology. Third we need to rely on an open standard that is widely used, with available commodities on the web. Web Ontology Language being based on the stack of W3C recommendations, and numerous commodities on the web being available, it responds to our criteria. In addition, the Web Ontology Language ground is particularly appropriate as annotation is a basic construct and as reasoning engine are available for free, making it possible to automatically complete an hypermodel using inference engines. Consequently classes, properties and individual are the chosen generic descriptive business concepts

Considering hypermodel only dedicated to information exchange, this set of generic descriptive concepts will be sufficient. When willing to deal with other kind of interoperability, related to services, processes or systems, they will have to be extended. Given the set of descriptive business concepts, the different relevant hyper-edges will have to be defined. A set of hyper-edges will be dedicated to hypermoding of business classes. A set of hyper-edges will be dedicated to properties. A set of hyper-edges will be dedicated to individuals.

Hyper-edges for classes

Each generic abstract modeling concept is associated with a set of concrete modeling constructs, one per language and formalism used, for which we can establish a semantic equivalence with the generic abstract modeling concept. Such equivalence will be established through study and comparison of the different languages and of the available mappings.
Equivalence is to be understood as defined by the semantic web community, working on Web Ontology Language, i.e. the meaning is the same, but the intent and usage can be different.
For most of the conceptualization, design and development language, we will have terms such as “class” or “entity”. It will be associated to persistence of data, inside the memory or on a physical media. The differences will come from the intention behind the languages: system engineering based on object paradigm, data exchange, description of an ontology, etc. In some case, modeling intention will be structuring information for efficient processing and automation, in other case it will be flexible description of real things subject to changes. But we can consider that most of the time, we have very equivalent meta-concepts, which are used to model the same business concepts.

Hyper-edges for properties

Properties are usually allowing establishing a typed oriented link between too individuals or between an individual and a value.
The usual terms used will be “property” or “attribute”. In several languages, the properties are defined independently from classes, as they can be applied to individual who are not classified. It is nevertheless possible to indicate that a class of individual is belonging to the range or to the domain of a property. In more structuring language aiming to efficient data structuring, the value of properties are attached to the classes in order to insure data proximity, fast access and concision. It leads to systematically define a property within the scope of a single class, which is the single element of the domain. But if having a property with a domain containing several classes, we have to define the property for each class. For example, “id”, “name” and “description” are properties which are systematically defined for each class of a language using attributes and not properties as a construct. The range of the property can also contain a single class, or several classes. Numerous mechanisms are proposed by the different existing language: usage of specific construct such as “selectType” for the EXPRESS language, generic abstract class of object oriented modeling, etc. Usually, a language using only “attribute” modeling construct and not “property” modeling construct will have to find some work around in order to formalize properties with domain containing more than one class. For example, an entity will be created with a name indicating it is a relationship, and attributes indicating the domain (relating) and the range (related). Nevertheless, semantic equivalence can be established. In some case, as for the given example, language mapping will not be sufficient and we will have to go through study and interpretation of the semantic of the model, i.e. used names.

Hyper-edges for individuals

An individual is thing that can be identified and have an identity. It can be any real thing which is represented within an information system, or any concrete representation of an abstract thing, e.g. a class or a property.
According the level of abstraction (e.g. layer of the model for Model Driven Architecture), a same thing will be formalized as an individual or a modeling construct.
In some case, it will be possible to mix the layer of abstraction in a model. Semantic graphs are an example. In some other case, it will be forbidden, as for example with descriptive logic.

It is important to realize that any modeling construct, in a language, is identified by mean of a unique identifier for any language. It will be the name of a class in a given name space, or the name of an attribute prefixed by the name of the entity.
Different kind of identification and identifier exist, which can be used to identify the same thing within different contexts (c.f. real person example).
According the phase of the lifecycle of an application, we will or not deal with individuals. For example, designing a database, we will define the conceptual model during analysis, the logical model and the physical model during design, and we will then produce the implementation. Data will come later, during the exploitation phase, and will be produced by application users. So the population of individuals is described during the last phase by users. Creating an ontological model, we can as well defined classes, properties and individual in any order. Individuals can be represented first, and then categorized.

Linking individuals, properties and classes

Some relationships exist between classes, properties and individuals which can be formalized with the language. Categorization or typing consists in indicating that an individual is an element of a class. A "class" is related to a "property" by putting the class within the domain or the range of the property. Hyper-edges will also be created for these each element of the graph of these relations.
If it is easy to represent hyper-edge for a class, an individual or even a property, it is a little bit more difficult for hyper-deges representing elements of a graph of a relation between classes, properties and individual. Figure 45 illustrates it by representing typing links between an individual and a class.

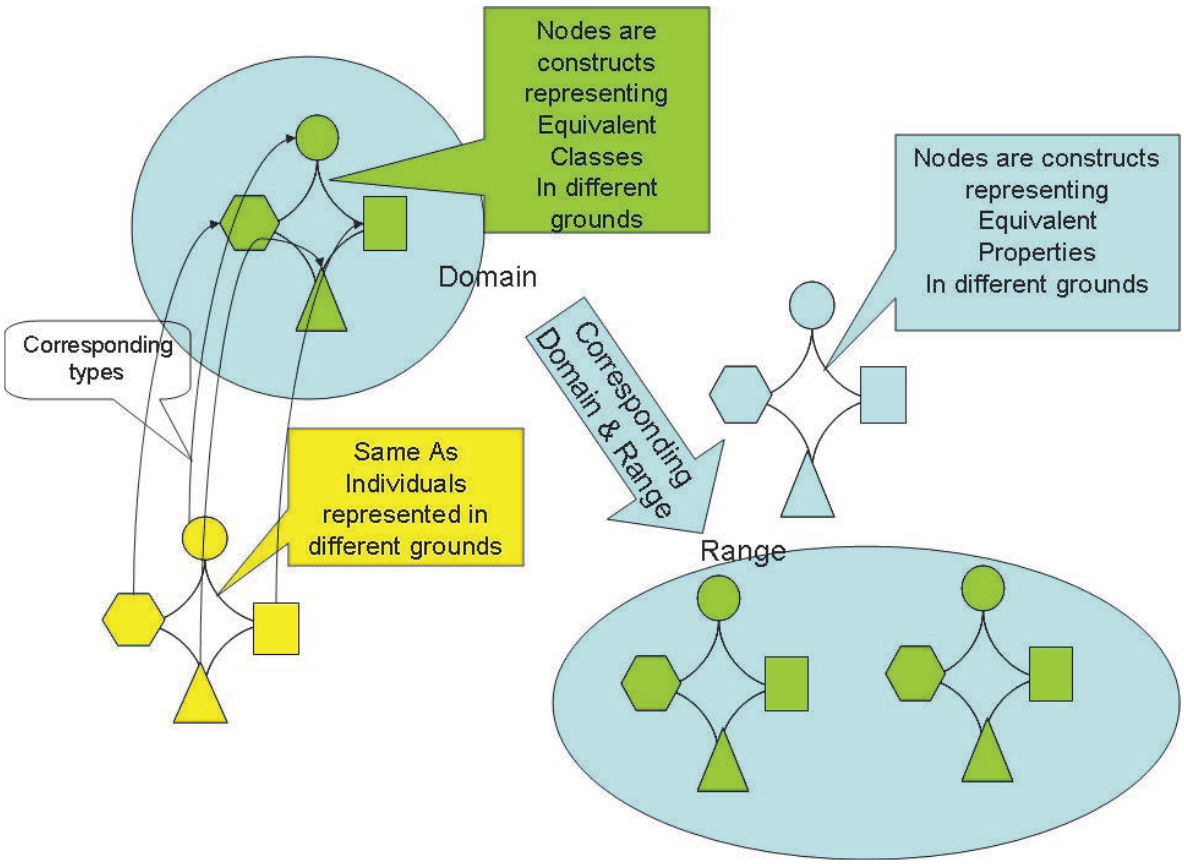


Figure 45: graphical representation of hyper-edges representing relationships between modeling concepts

7.2.3 – Operations related to hypermodels

Once the different generic constructs have been defined, the next step will be to establish the equivalence between the different languages of the considered landscape in order to support the model driven engineering process. Once validated on several ground, it will be possible to defined different operation in order to ensure the travelling between environments.
From an existing model formalized with one language, we can realize the completion of the hypermodel by generating representation for the other languages.
From a hypermodel, we can validate that all asserted representations are correct and complete.
To make a model travel, it should be exported, transposed (file syntax and formalism changed) and then imported. For most of legacy tools, the export, transposition and translation will have to be developed.
In some cases, grounds will not provide way to capture how the model is represented within the other grounds. In such a case, traveling will be performed by mean of bidirectional translations. In such a case, it should be done on models using only the constructs for which equivalence exist, in order to avoid loss of data.

Chapter 8- An example illustrating extended hypermodels for manufacturing ecosystem

8.1-Identification of the standards of the PLM ecosystem

Within the manufacturing ecosystem, it was identified that the relevant set of model of reference to enable interoperability are STEP application protocols, PLCS Reference Data Libraries and PLM services. The relevant standardized languages to consider are EXPRESS, for product data exchange and sharing (ISO STEP), Web Ontology Language for data integration and semantic web (PLCS RDL), XML schema for service oriented execution platforms and Unified Modeling Language for Model Driven Engineering applied to applications.
For services, we identified Unified Modeling Language and Web Service Definition Language which are used to define PLM services. But some others languages can also be considered, such as Interface Definition Language used for PDM enablers specification, or OWL-S which extend OWL for service definition.
For processes, we identified Business Process Modeling Language, Unified Modeling languages, XML Process Definition Language (referenced by PDM Enablers through Workflow specification) and finally Business Process Modeling Notation and Business Process Execution Language which are emerging standards for composition and orchestration of services.
For relevant service oriented execution platforms, we identified EJB3 specifications (JSR181), Workflow Management Coalition specification and WEB specifications for service oriented architecture and semantic web.
For transformation, we have numerous standardized language we can use, that are related to the different frameworks provided for software engineering, service oriented architecture, data exchange, etc.

These languages are distributed between the different environments of a collaborative product development framework enabling model driven approach, which is illustrated by Figure 47.

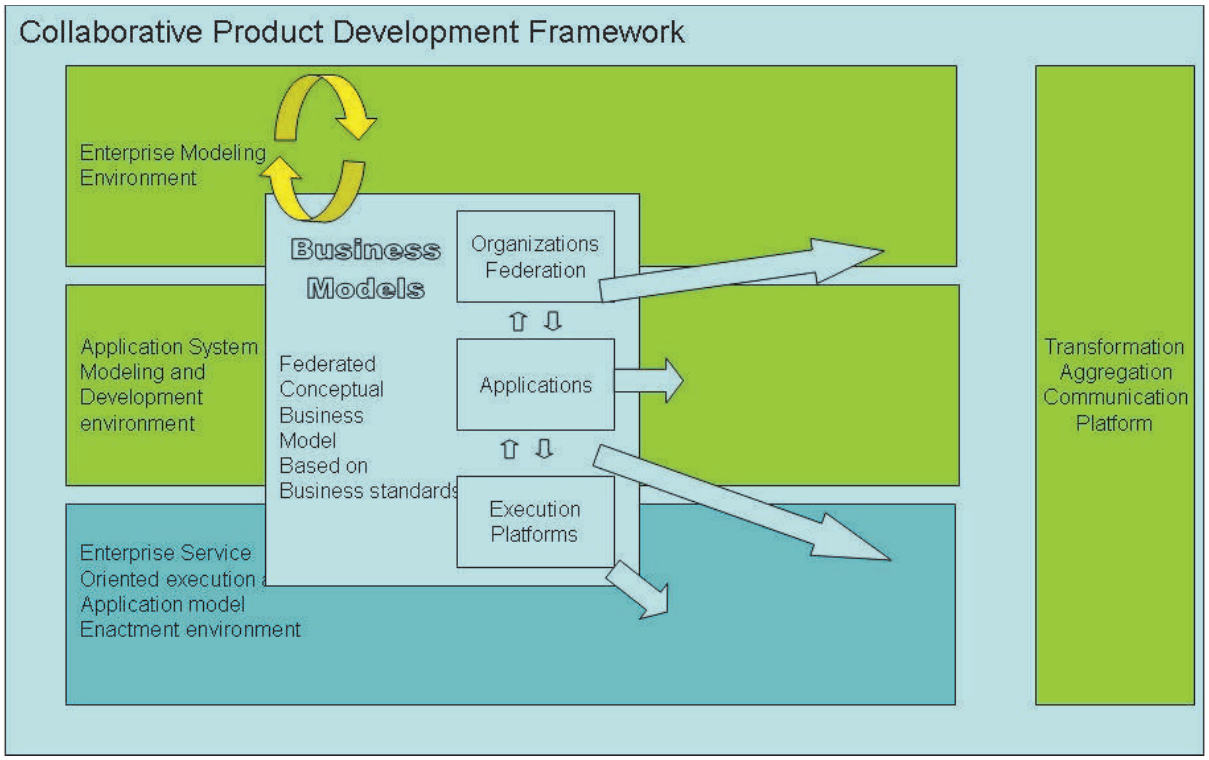


Figure 46: Extended hypermodel usage context

It includes enterprise modeling environment with modeling tools, application modeling and development with design tools and programming tools and environment and execution environment. In addition, we consider communication enabling environment which provide services for transformation, aggregation and transportation of hypermodels between these environments. These different environments are populated by commodities on the web implementing the identified standards.

8.2- Definition of the landscape

The landscape definition is initiated by a model driven process we want to implement. For our case, the legacy business models of reference are the STEP application protocol. We consider that we want to produce ontology in OWL from the application protocols, in order to obtain model for our conceptualization environment. When validated logically, we will generate UML computer independent model, then UML platform independent model, then generate Java code with EJB3 annotation in order to product EJB entities which will be deployed within an application server (projection of the business logic).

Three grounds were identified. The first ground is part of the Enterprise modeling environment. It is related to OWL 1.1. It includes Protégé 3.4 and Protégé 4.0 associated with Pellet reasoning engine and Jambalaya visualization plug-in. This environment is compliant with the specification of OWL1.1, and allows serialization in RDF/XML but also using other formats. Other grounds can be built belonging to OWL 1.1 landscape, as for example using Swoop, Jena2, etc.
The second ground is related to UML 2.0. It is related to design and development environment. It includes Papyrus UML 1.11, which is based on Eclipse Europa, with appropriate Eclipse Modeling Framework plug-in and model transformation tools (Action Transformation Language plugin). Other grounds can be built belonging to UML 2.0 landscape, as for example Rational Rose Sotware, Magicdraw 12, etc.

The third ground is JBOSS 4.05GA, which support EJB3, and annotated Plain Old Java Object annotated according JSR181 specification. It is related to execution environment.

The two first environments are allowing performing meta-modeling and transformation. They can be consequently considered as meta-workshop. Let's focus on them, and let's consider a very simple hypermodel, concerning a single hyperedge: "Person" business concept. This hyperedge will be constituted of nodes UML:Class, EXPRESS:Entity, OWL:Class and XSD: Class. This hypermodel can be described within the two previously defined grounds, as illustrated by figure Figure 47.

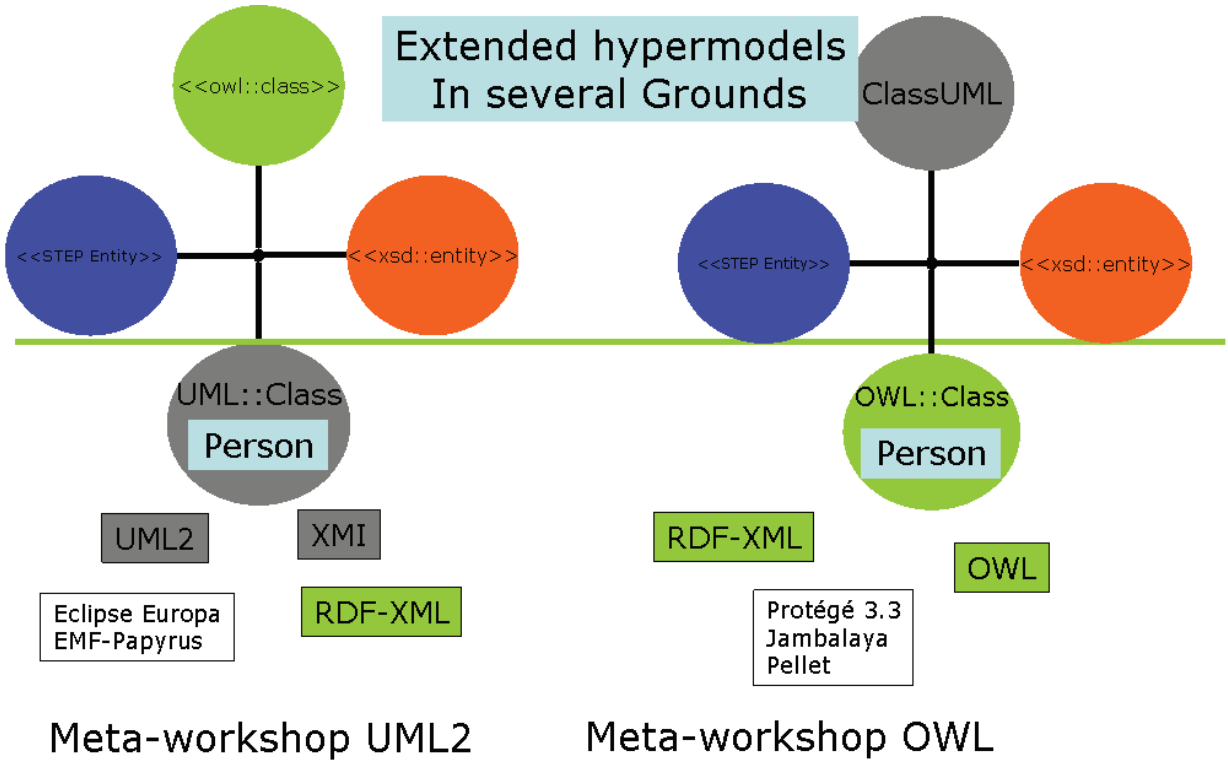


Figure 47: A simple hypermodel within two grounds

8.3 - Hypermodel formalization

Using OWL 1.1, it is possible to create a class "HyperModelNode" and to populate it with UMLClass individual. This individual can be attached as a comment to the class "Person", as illustrated within Figure 48.

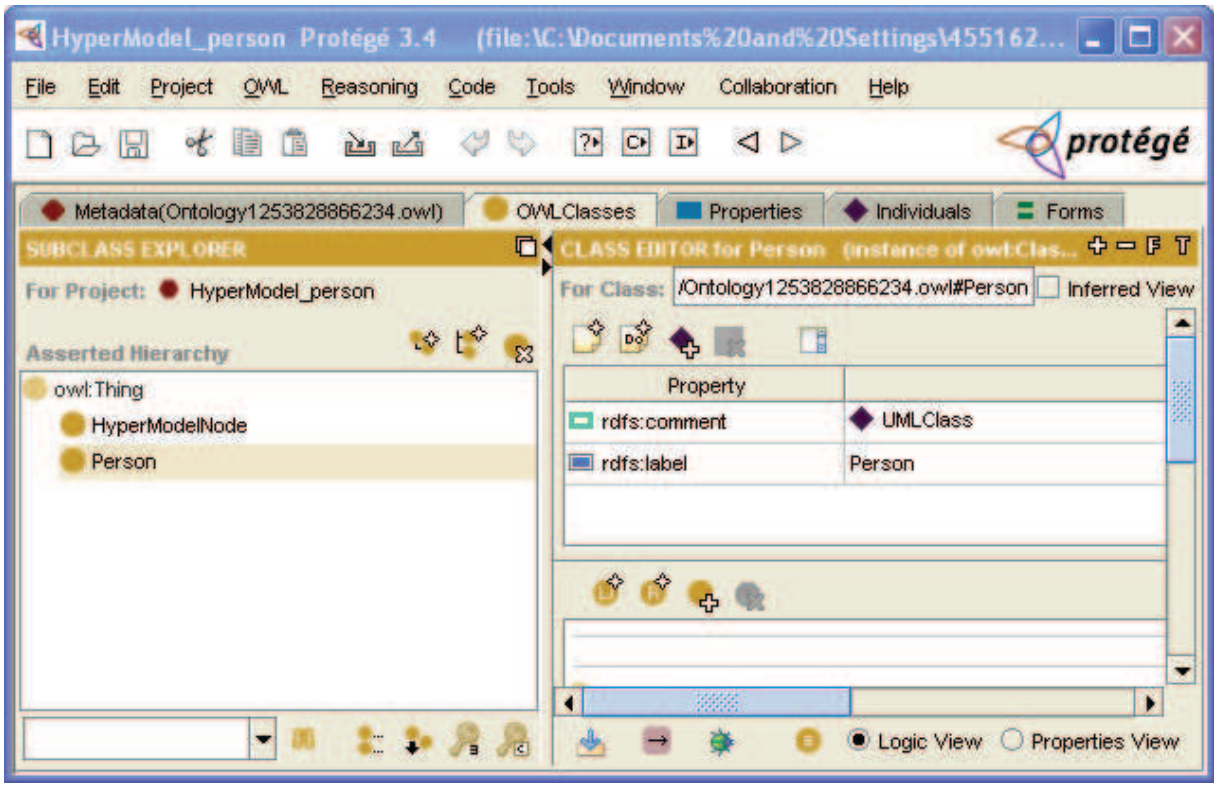


Figure 48: Formalising an extended hypermodel with Protégé 3.4

As the Protégé 3.4 does not allow creating correctly all the annotations of OWL, a similar way could be used for annotation, using an annotation property in place of rdfs:comment, as illustrated by Figure 49.

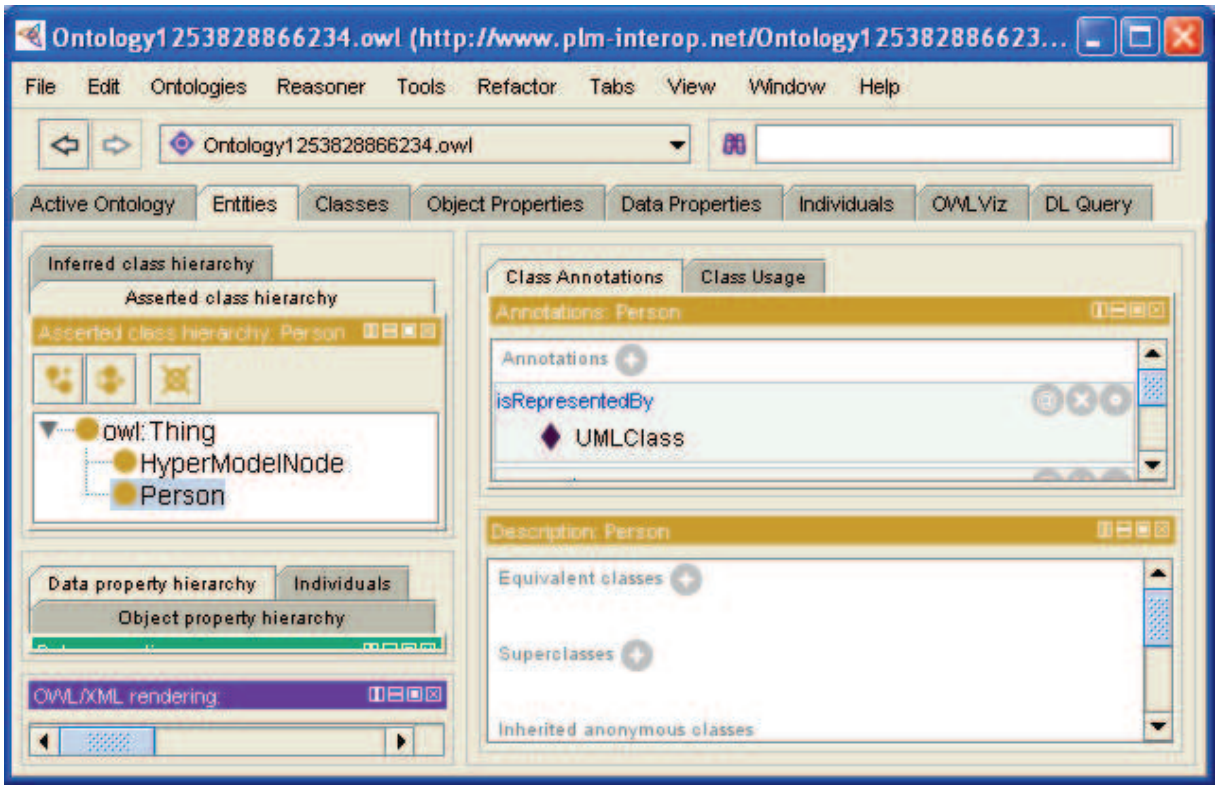


Figure 49: Formalizing an extended hypermodel with Protégé 4

A single way to formalize the extended hypermodel should be defined, according OWL language and the capabilities of the used grounds. The example here illustrate importance of tools really compliant with the standards and allowing to support efficiently all the construct of the language.
Within Papyrus, the hypermodel is represented by means of stereotypes, extending the Person class, as illustrated by Figure 50.

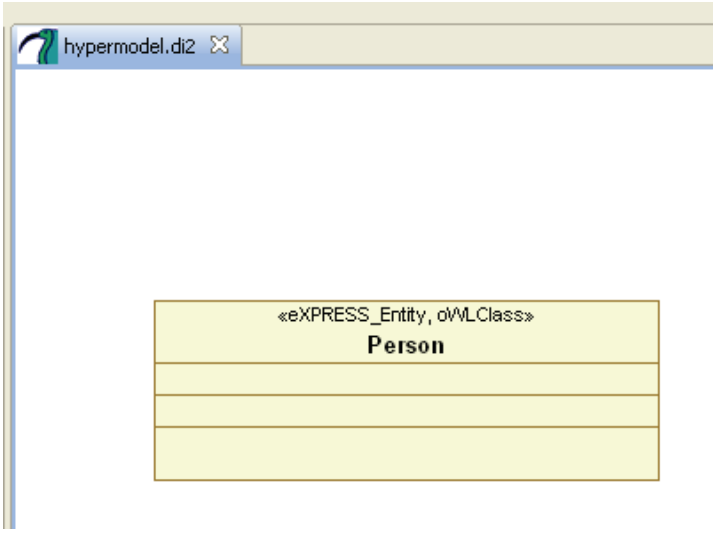


Figure 50: Formalizing an extended hypermodel with Papyrus 1.11

8.4- What exactly a model travelling within hypermodel landscape is

For the given example, it will be to extract the hypermodel from Protégé, and to transpose it using annotation related to UML constructs, completed by stereotype concerning EXPRESS and OWL constructs. In addition, the file format of export, RDF/XML, should be transformed in XMI.

8.5- Contextualization of hypermodel within a model driven approach

Having identified all the standardized formalisms and standardized models of references we have to use within the PLM community, we will have to establish the complete list of hyper-edges we want to consider and represent, and also the way to formalize it in used ground with existing capabilities when possible. As several solutions exist, they have to be assessed in order to define a single one. Figure 51 illustrate we need multi-directional mappings between the different used languages.
We also need, for a model driven approach, to clearly establish if the languages are used for producing computer independent models, platform independent models, platform specific models, code or operational artifacts. It is illustrated by Figure 52.

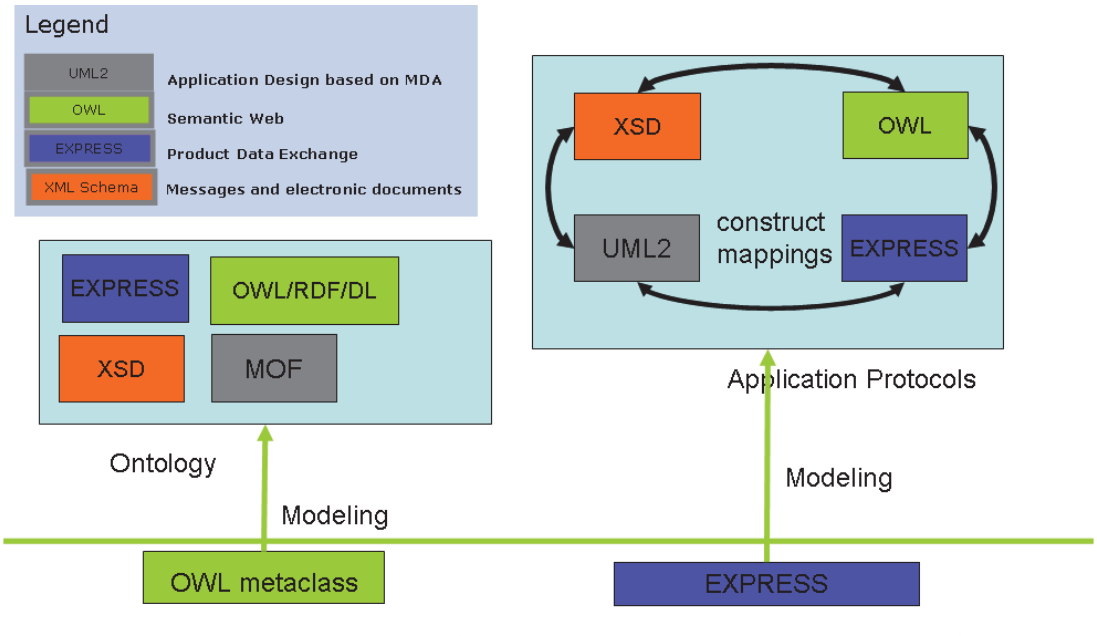


Figure 51: Different mapping formalization environment for different communities

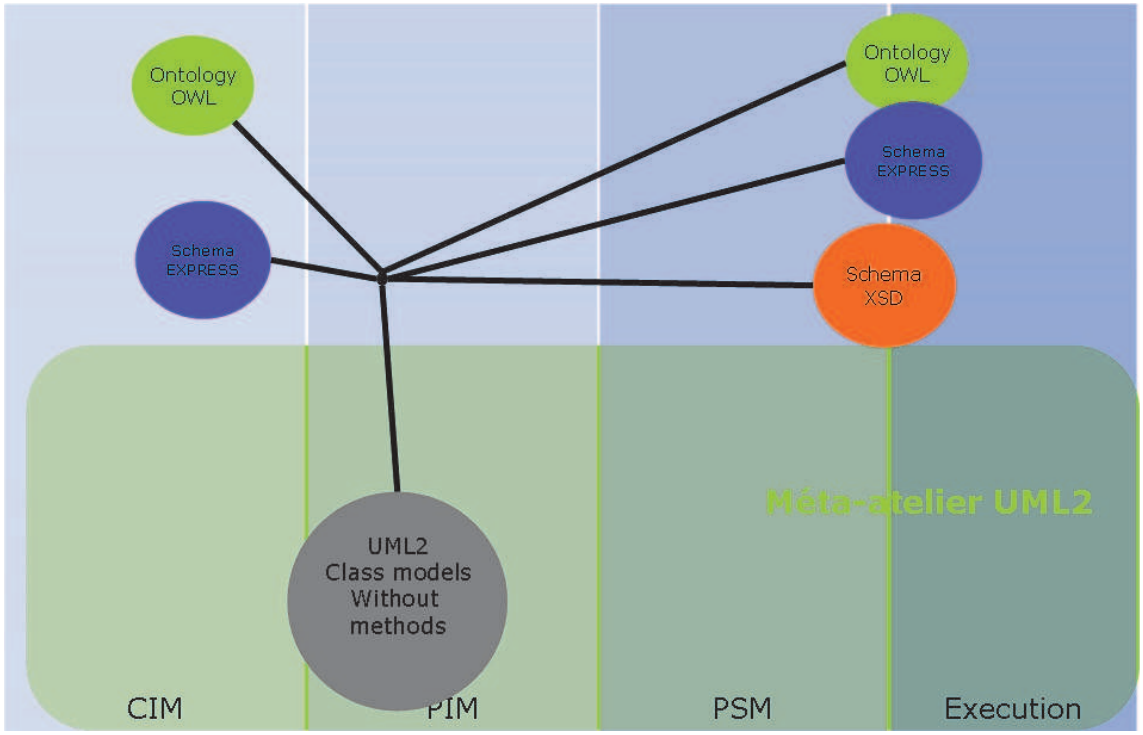


Figure 52: Categorization of informational grounds according MDA models typology

Such categorization of the languages and consequently associated grounds is arbitrary, and will depend on the set of standards chosen by the considered community defining the framework.

8.6 - Extending the landscape for business services and processes

The idea is not only to consider informational viewpoint, but also services and processes. For modeling of services and processes numerous standards also exist with can be selected and categorized the same way.
For example, I selected for my use cases OWL-S for ontologies (definition of services and their composition in OWL), the Interface Definition Language of CORBA (which is used for PDM Enablers) and Web Service Definition Language (which is used for PLM Services). It is illustrated in the following figure, with UML used for PIM and PSM models, and WSDL, IDL and OWL-S used at execution. Here again, the choice is related to the global context and the considered community.

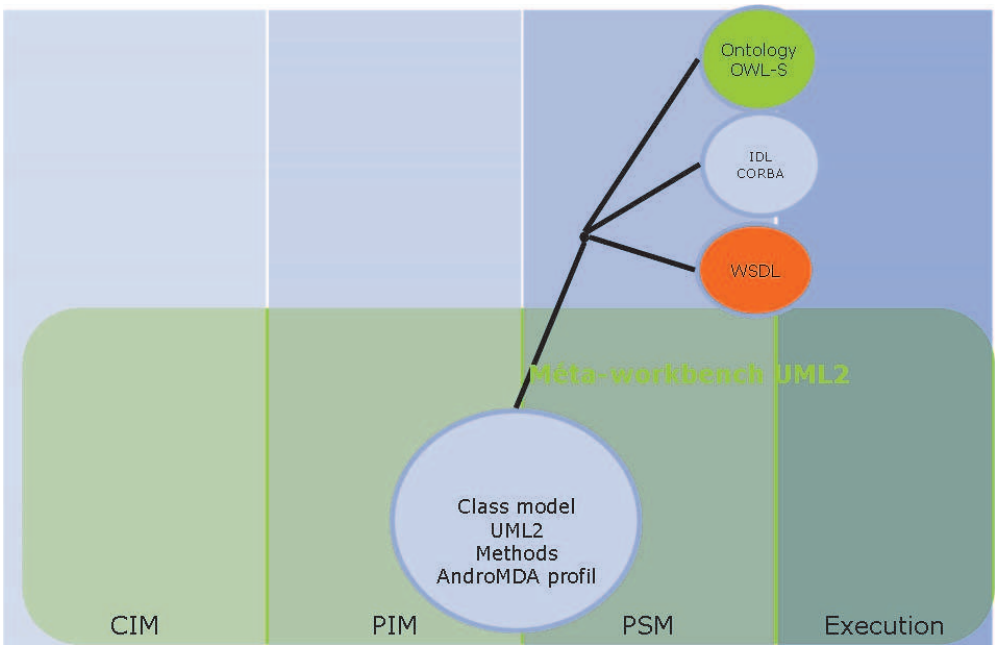


Figure 53: Categorization of service related grounds according MDA models typology

Considering process modeling language, the same categorization can be performed on relevant standards. The relevant languages, and most of the time standards, I identified which are relevant within the PLM context are SPEM (modeling of a methodology), BPMN (graphical notation, dedicated to eBusiness and service oriented infrastructure), BPM4STRUTS (for use cases and human application interactions), OWL-S (for semantic combination of services), BPEL (for orchestration of services) and XPDL (for workflow systems and choreography of services provided using heterogeneous technologies). UML 2 is of course used at design time with activity models. But most of the time these UML should be specialized with profiles related to the other identified languages, as too generic.

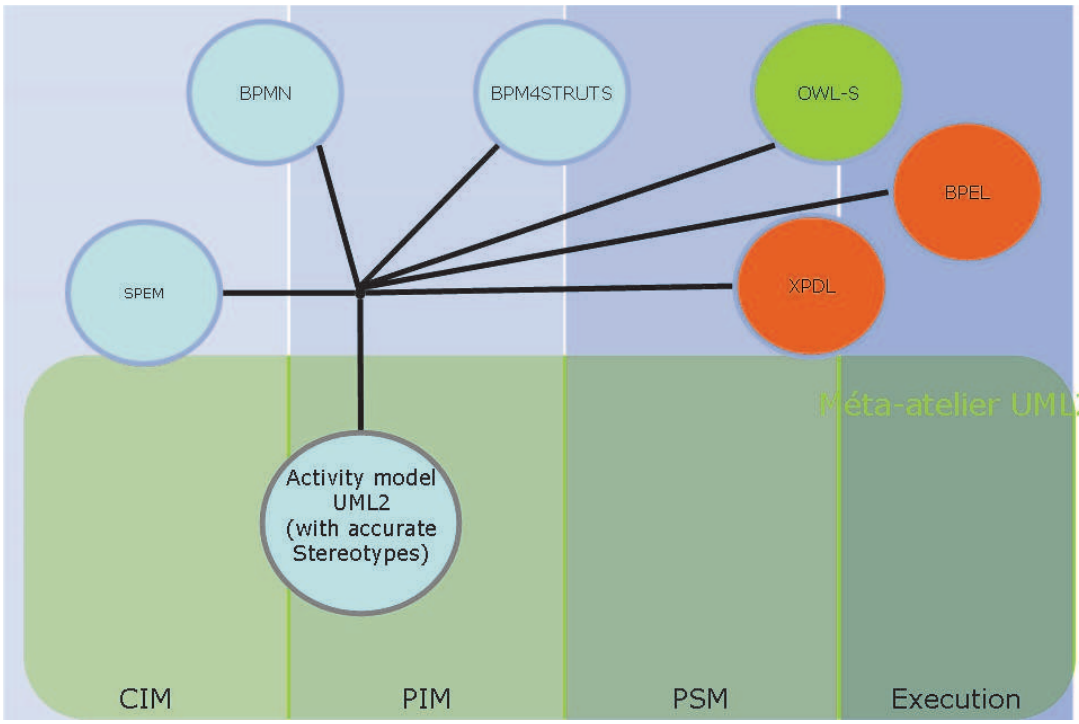


Figure 54: Categorization of process related grounds according MDA models typology

Considering information, services and processes models, it is also important to be able to aggregate these models in order to obtain coherency. Here the principle 5 is to be applied, in particular for linking entities, services, messages and processes. For an extended hypermodel, this idea is to ensure that all the more used modeling constructs for open de-facto standards selected as reference within the collaborative network can be aggregated through a hypermodel with well established conceptual equivalence and coherent representations within the different grounds. Finally, it is important to be able to interconnect the different kind of models we want to federated, being information, service or process model. I studied the different languages in order to extract the main generic descriptive business concepts that should be considered to build hypermodels, and how they can be interrelated in order to be used together.

Generic descriptive business concepts to consider are service, operation, typed input and output, messages, processes and activities.

Figure 55 provides an illustration of roof mapping of these descriptive concepts used for entities representations (with XML, EXPRESS, EJB Entities, XML, etc), services/operations/messages (WSDL, SOA, IDL) and processes (UML, IDEF0, BPMN, BPEL, XPDL, etc).

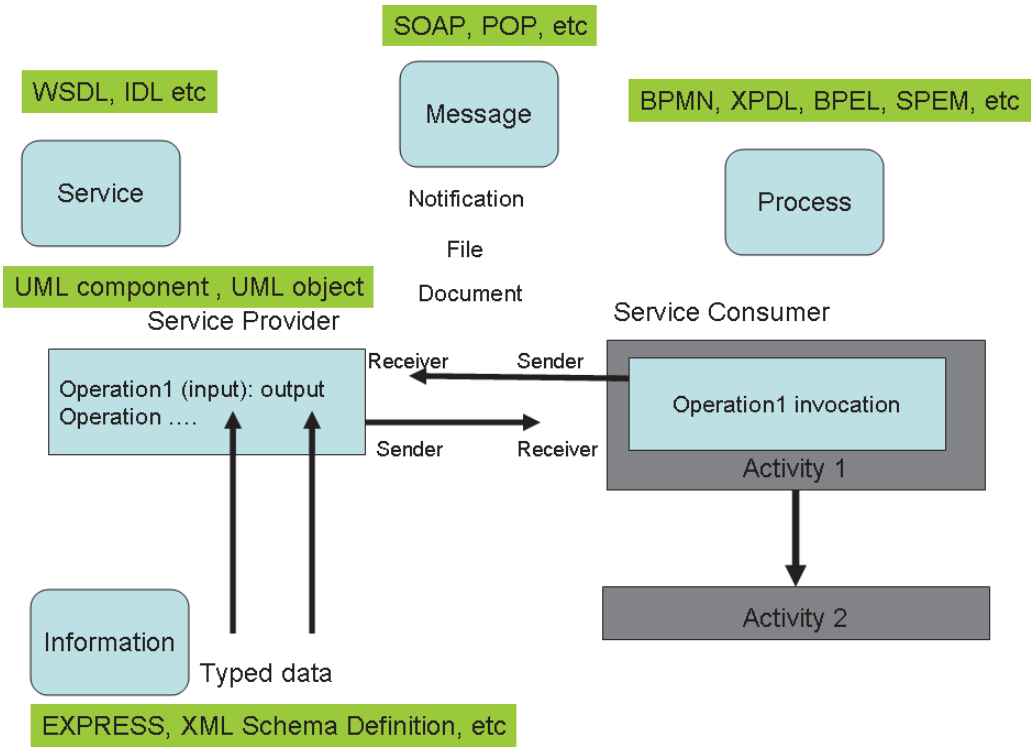


Figure 55: Linking entities, services, messages and processes

8.7- Impact on standardization

I have provided within this chapter illustration of concrete usage of extended hypermodel for interoperability. It was done within the context of PLM eBusiness community willing to established interoperability frame, federating legacy standards and frameworks of interest, in order to support improvement of their interoperability maturity level.

It is important to point out that using extended hypermodel for interoperability is not only a tool for application interconnection. It is also a tool allowing to assess joint usage of standards and to support governance of standards for a community by allowing to select and to invest only on those which are really relevant. From technical point of view, it is a way to assess legacy standards, and identified issues which are requiring changes and evolutions. It can allow pointing out some defaults of the specifications, and initiating a quality process with continuous improvement of standards for enterprise interoperability purpose.

It also allows validating implementation of standard and their conformance. Applying the approach, I identified numerous drawbacks related to lack of conformance of tools with the standards. So performing the exercise is also a way to insure quality of implementations of references and of tools. Finally, the assessment can be done at community level, with tracking of issues and knowledge sharing between different specialist, allowing growing maturity of all the stakeholders, adopting a global and systemic view.

For example, it was identified that XPDL 2.0 is mixing definition of drawing and business concepts, with unclear rules on how to use them simultaneously.

It was also identified that some tools with are based on open standards don't cover the full specifications. As an example, ArgoUML does not provide user interface to create efficiently annotations, despite the fact annotation are part of UML 1.4. Protégé 3.4 does not support properly creation of annotation through the user interface too, despite the fact it is part of the OWL 1.1 standard. Usage of hypermodel is a tool allowing improving global quality of standards that are to be used jointly in order to insure pragmatic interoperability.

Chapter 9- Generic process for building collaborative space

9.1-The main stakeholders

The proposed approach is not universal, and will support community willing to achieve upper level of interoperability maturity to build their referential and methods in term of interoperability standards, architectures and frameworks. Standards becoming components on the shelves, as the tools implementing them, supporting their selection and integration is absolutely required.
The proposed approaches don't solve all the issues. As I consider we are within an evolutionary environment, what is important is more to be able to identify the issues and to help to deal with them, knowing that evolution of context and evolution of needs with always be source of new issues and non interoperability, which will have to be solved or avoided by efficient knowledge sharing about complex environments of enterprise applications.
Consequently, I consider that proposing a detailed process or a methodology to deal with interoperability of enterprise application is useless, as probably too rigid and subject to continuous changes and adaptation.
I consider that, having a clear picture of the kind of interoperability a community want to achieve, and a legacy to deal with, it will be up to each community to put in place their frame, but taking into account interoperability standards and frameworks which are out of their domain, in particular for information and communication technologies.
I also consider that some other new kind of actors are to be considered, which will be the operator of the collaboration space, and that they will have, in collaboration with the community, to establish the collaboration infrastructure and to support the infrastructure of the collaboration.
The enterprise will be able to join and to leave the community, and then to connect their applications to the collaborative platform in order to support set of collaboration processes implying digital resource sharing and computer aided collaboration. A set of simple processes dedicated to set up, to support and to operate a collaborative space is represented in Figure 56.
Once the applications connected to the collaborative space, it will then be possible to launch collaboration with other member of the collaboration space.
The proposed approach is to be applied when defining the collaboration space, the ICT (Information and Collaboration Technologies) infrastructure, developing the business collaboration platform and supporting its evolution.
It will include identification of the relevant standards of the community, establishment of the landscape of the coloration, selection and assessment of tools in order to support definition of the model driven method to choose.

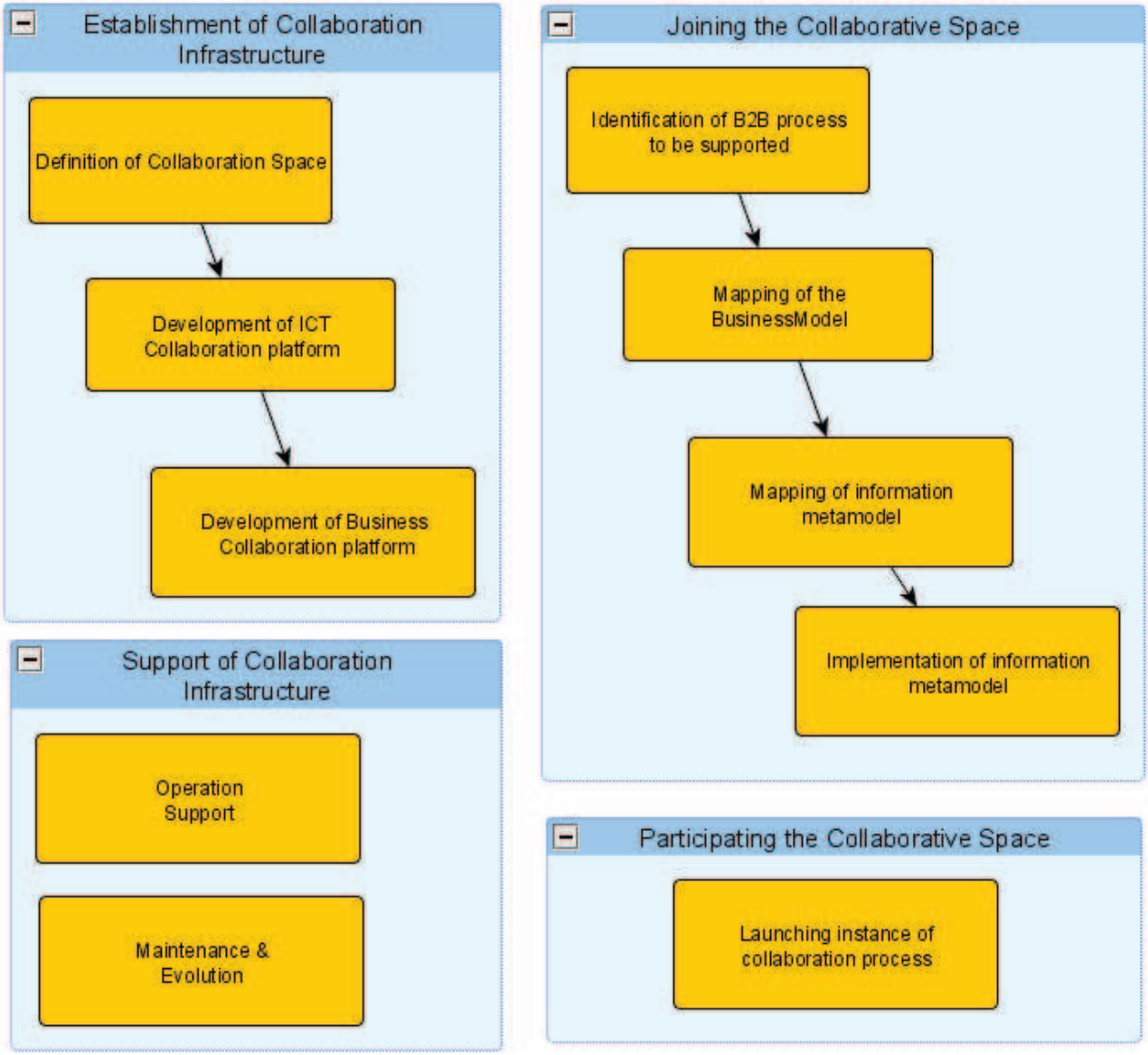


Figure 56: Processes associated to networked collaboration environment

As a principle is to select already available components on the shelves, it is strongly suggested to adopt System Engineering practices, with clear identification of actors and stakeholders. It will be particularly important for governance of standards, which will be enabled by clear involvement of all the stakeholders and actors which are concerned. It will in particular clarify who is doing what, and why. It will also allow identifying easily impact of the changes. It is illustrated by Figure 57 which provides an example. It describes how, when willing to develop an applicative system (enterprise application) which will be operated within a targeted environment, we can picture the different actors which will have to select component on the shelves, or develop their own, in order to finally design and produce a complete system which will be deployed, operated and supported within an operational environment.

In addition, it could be a high added value to formalize such System Engineering Process using formalism such as Software Process Engineering Meta model, which will allow to characterized the different produced artifacts

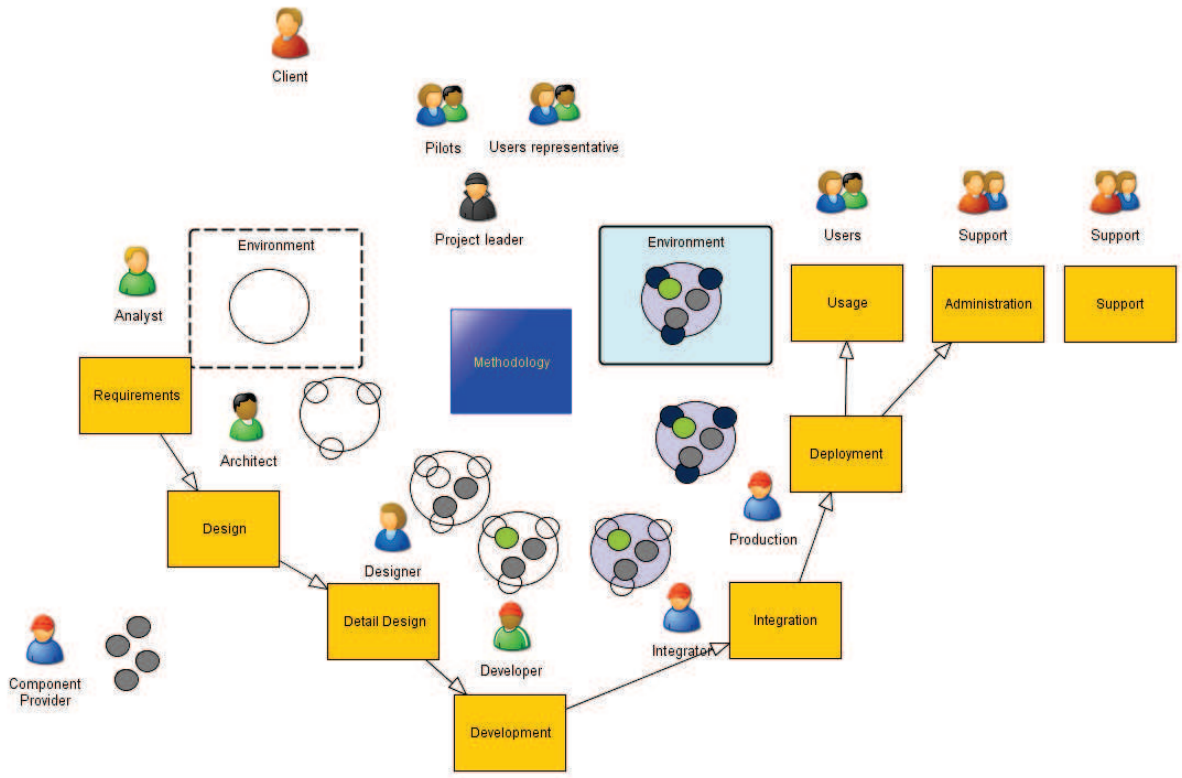


Figure 57: Actors and stakeholders implied within phases of a application lifecycle

Finally, a clear distinction should be done between processes implying a single person, the final user or operator of the application, the group he belongs to, the enterprise the group belongs to and the community the enterprise belongs to. Depending on the granularity we are considering, all can be consider as end users. Most of the time, the client is not the operator of the application, but the group he is belonging to. It is illustrated in Figure 58.

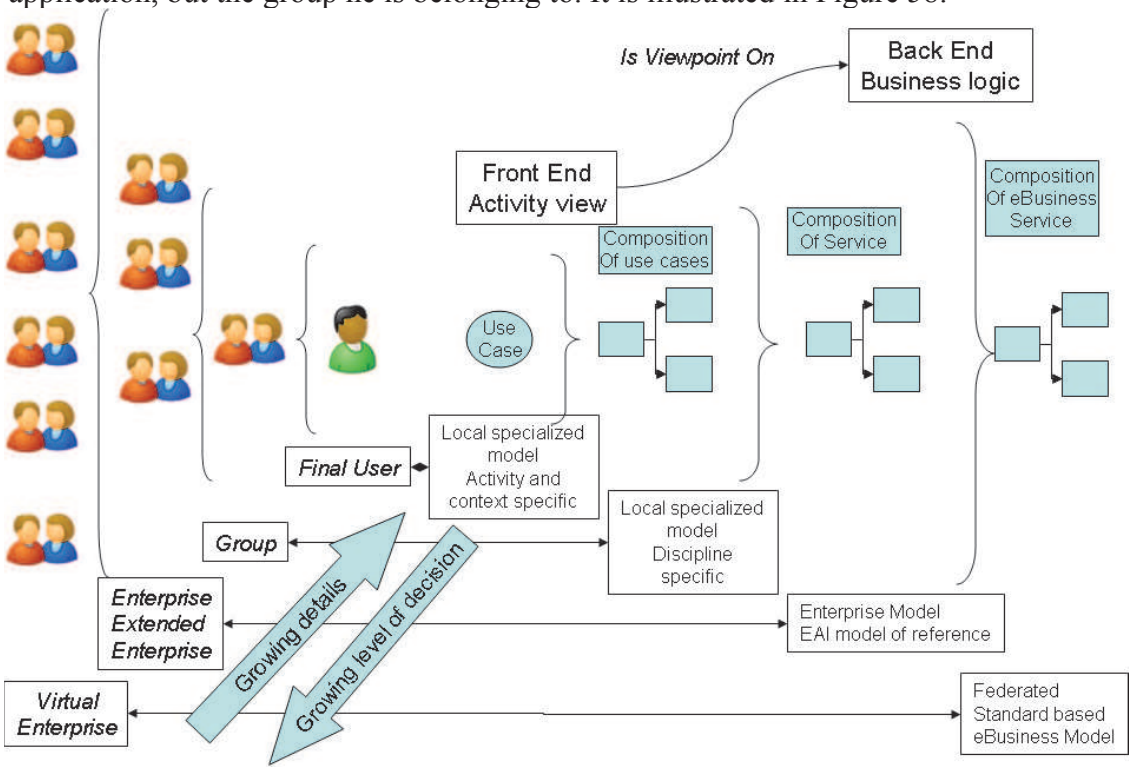


Figure 58: Service classified from user centric prospective to eBusiness Community prospective

9.2- A generic scenario implying operator of the collaboration platform

At the very earlier stage of the application development, the standardized models can be reused in order to formalize the requirements, providing some computer independent model.
These models can also be available as component on the shelves, provided by operator of collaboration infrastructure which is used by the enterprise where the application will be deployed.
For the application architect, he will be able to rely on model of reference and standards of the community when willing to interoperate, but also to optimize architecture in order to have open, flexible and agile application. He will also be able to select execution components which are validated by the community and compliant with standardization policy for continuous interoperability. Architect and designer will be able to provide specification to the developer and set of test being supported by the operator of the collaborative space.
The production will collaborate with the operator of the collaboration platform, in order to connect the application to the collaborative hub, and will be able to qualify all the collaborative processes that can be supported with the outside.
Then it can be used by the designer to generate platform independent models.
In the reverse, the organization and the different level of services will have to be reflected within parameterization of the collaborative platform, in order to insure proper support for the collaboration.

9.3- Efficient governance for high quality set of standards

It is absolutely required to organize governance of standards in alignment with a process to continuously identify and address the issues for standards, frameworks and software components, in order to insure quality of the infrastructure and continuous interoperability. It will have to involve enterprises of the network and the operator of the collaborative platform.
The different standardization organizations are too numerous and different to propose a generic process. The referential of community standard and knowledge about issues and principles should be freely available and shared by all the members of the community.

Part3 – Concrete experimentations and implementations

In this part, I describe concrete experimentations and implementations of the principles defined in part 2 for setting up collaborative space enabling interoperability of technical enterprise applications

They were realized through different and successive projects, with iterative resolution and discovery of interoperability issues (Part 1), and continuous assessment and improvement of the proposed principles (Part 2) to apply when willing to set up environment with collaborative interoperable technical applications.

The need of extended hypermodel emerged from this different experimentation.

A first experimentation (chapter 10) is the Networked Collaborative Product Development platforms I developed during the ATHENA program. This platform is dedicated to the PLM domain.

A second experimentation (chapter 11) is the set-up of a meta-modeling platform, Papyrus, to support interoperability between System Engineering platform and Simulation Lifecycle Management enterprise application. This platform is based on open source Eclipse framework. It implements and is compliant with Object Management Group's Model Driven Architecture and Unified Modeling Language standardized specifications. Different enterprise modeling languages were imported as Unified Modeling Language profiles, in order to evaluate Papyrus as a meta-modeling platform. The idea of the extended hypermodel concept comes from the experimentation of combining different modeling language on a same platform.

The SEINE project (chapter 12) was the opportunity to apply the principles within an industrial context, in order to define a collaborative hub for the European Aerospace Industry, based on usage of manufacturing standards. In order to prove accuracy of STEP standards and validate interoperability requirements, I developed in one month a full demonstrator from scratch.

Chapter 10- ATHENA Networked Collaborative Product Development

As described within the preamble, I was involved within the ATHENA program. I provided business scenarios and needs specific to aerospace ecosystem. I was leading dynamic requirements engineering project. I was involved as a contributor for the ATHENA interoperability framework. I was finally involved in piloting activity for assessment of integrated ATHENA solutions for coverage of full spectrum of needs.

10.1 - Introduction

Networked Collaborative Product Development is an Aerospace pilot I defined to cover needs for advanced emerging collaboration needs according identified trends, aligned with industrial context description as described within the introduction of the thesis: virtualisation of the enterprise, virtual aircraft for early involvement of downstream activities, virtualisation of the Product and emerging Product Lifecycle Management approach from requirements to recycling.

The identified interoperability challenges were the followings:

1- How to ensure efficient global Configuration Management and Product Information/data Coherency between different PDM Systems?
2- How to ensure efficient Product Data Exchange, Sharing and Long Term Retention supporting Business Processes such as Exploitation, Maintenance, Support, Change and Configuration Management for Aircrafts in operation, Traceability for Legal information?
3- How to ensure seamless collaboration of designers and technicians despite heterogeneous environment (process, methods, applications and software products)?

A very strong requirement is integration of open standards for product data exchange and sharing based on STEP technologies. These technologies and covered domain (Product Data Management) was not in the domain of competency of the involved partners, which were promoting other technologies. It was nevertheless required to be able to take advantage of the STEP application protocols as de facto standards. Strong discussion occurred related to the definition of ontology.
From the definition provided by [Gruber93], “an explicit specification of a conceptualisation”, an application protocol can be considered as an ontological model. But experts and researchers of the ontology domain consider that it is not the case as it does not support reasoning engines. In addition, it was also required to be able to use application protocols as Computer Independent Models, and to be able to projects them at the appropriate place of a service oriented execution platform.

10.2 - The process

It was done according the Principle 9, "Process for establishment of collaboration infrastructure". The defined and followed process was the following:

Establishment of the Collaborative space

1- *Definition and design*
   - *Actors : Networked Organization Collaboration Space Architect and Consortium behind the network*
   - *Tools : Enterprise Modeling tools*

2- *Development*
   - *Actors: ICT people*
   - *Tools : standards, software engineering tools, Commercial Off-The-Shelves software*

3- *Exploitation, maintenance and evolution of the collaborative space*
   - *Actors: Networked Organization Managers, ICT people*

Joining the networked organization

1- *Identification of B2B processes to be plugged on the collaboration space*
   - *Actors: Business people*

2- *Mapping (process, roles, actors, tasks, stakeholders, objectives…)*
   - *Actors: Business+disciplines experts*
   - *Tools: modeling-mapping tools at enterprise/business level. Includes enterprise modeling tools and software engineering tools*

3- *Implementation of the mapping*
   - *Actors: Developers (not traditional, those using, parametering and controlling execution of services provided by ATHENA for transformation, negotiation…)*

Participating the networked organization

1. *Launching instance of collaboration process*
2. *External Change Management Process*
   - *Actors*
     Virtual SpaceCraft Program
     Integrator (EADS), Design Office, Engineer E
     Engine Provider, Design Office
     Landing Gear Provider, Design Office, Engineer F
   - *Tools*
     local PDM system- as workflow enactment service, connected to the Collaboration Space tool

Leaving the networked organization

1. *Disconnecting the organization from the collaborative space*
2. *To update information concerning the leaving partner in the networked space to indicate that he is no more part of the network*

Underlying cartography of actors was defined in order to define user profiles for the applications.

10.3 - The collaborative space

I defined a Networked Collaborative Product Development Organization and infrastructure in order to provide a collaboration space for a network of enterprises involved in an industrial program to develop the virtual spacecraft. It was used to run different pilots which were an opportunity to set-up such an environment in order to use and evaluate ATHENA results, and to integrate ATHENA solutions together with specific solutions coming from the outside. These solutions were open source and standard based solutions.

Functional needs to be covered by the targeted collaboration platform were modeled through formalization of use cases and of a composite system integrating pre-existing execution components. In order to obtain a flexible, agile and open solution, the system was design in order to use components based on open standards and commodities on the web (manufacturing and ICT, i.e. principles 1 and 2)

Figure 59 illustrates the different functional components of the proposed collaborative space, with associated standards for which implementations exist as commodities on the WEB, with their distribution on modeling environment, execution environment and communication environment. I consider the standards as artifacts produced by a standardization activities.

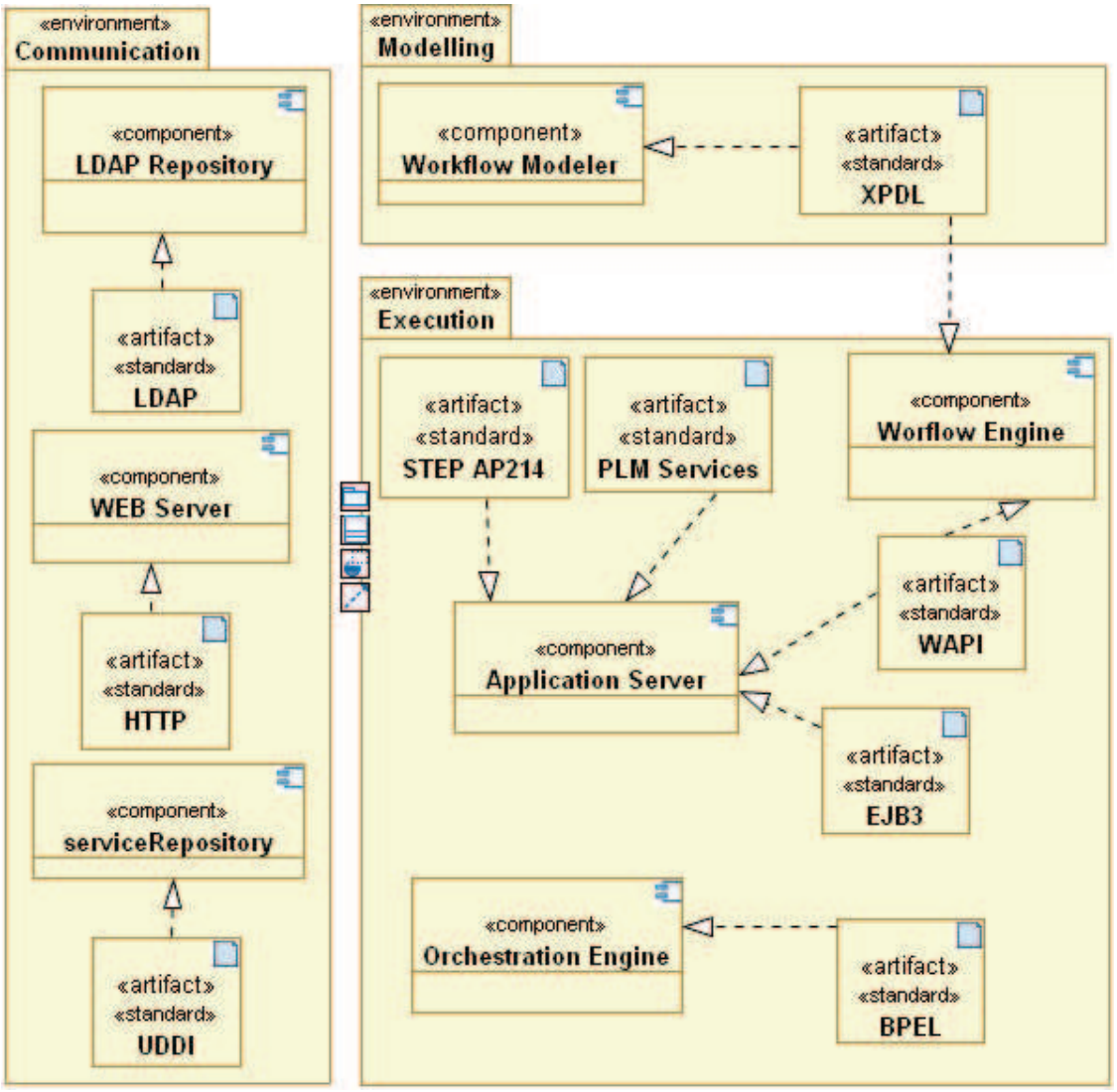


Figure 59: Application of interoperability principles within ATHENA

In order to select relevant standards, the approach was to consider open standards defining conceptual models, logical models and physical models for processes, services and information, as illustrated by Figure 60.
The more suitable standards were those already used by the eBusiness and Manufacturing communities, for which some available open source implementation were available. Different implementations were assessed, and then combined in order to provide concrete implementations of each component of the different environments.

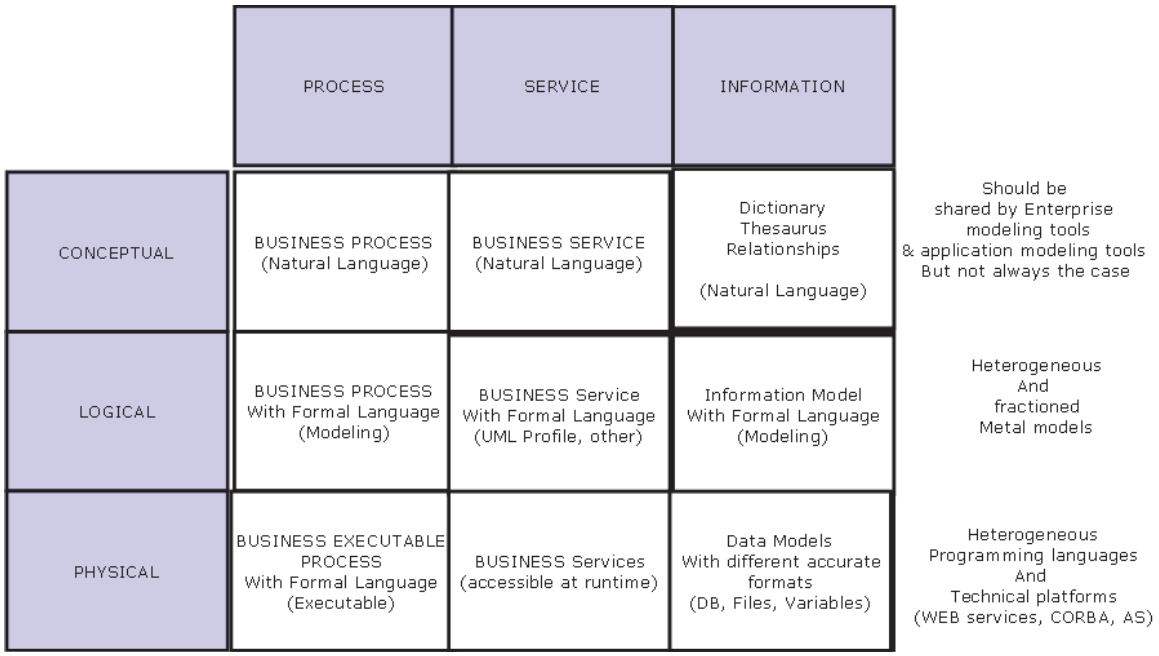


Figure 60: Framework for selection of standards

Figure 61 illustrates the different software components which were used for the piloting activity. The business logic, formalized by mean of business models such as STEP application protocols for the business information, PLM services for the services and Configuration Management standard (CMII) for the business process, was projected on the execution platform.

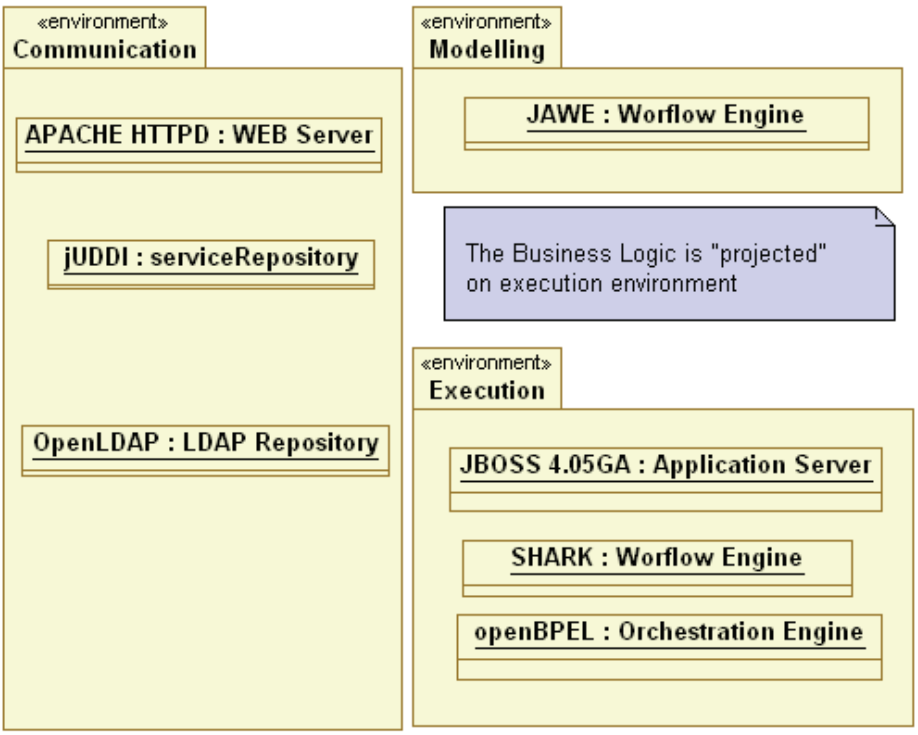


Figure 61: Concrete environments of the collaborative process demonstrator

10.4 - The different pilots

Different pilots were run in order to assess solution components provided by ATHENA partners. The strategy was to be able, for each covered function, to find equivalent open source component based on open standards, in order to identify added value but also to be able to have a fully functional environment.

Semantic mediation

This pilot was aiming to assess solutions for semantic mediation. The technologies coming from the "ontology" domain were compared to those provided by the STEP community.
The ATHENA project dealing with ontologies was aiming to address semantic mediation proposing a simple ontology for Organizations, People and Activities (OPAL), formalized with OWL, with associated semantic service implementation in order to build a semantic Hub. Different components were respectively developed by the project allowing formalizing reference ontology (RO), to formalize the semantic correspondence and to perform the transformation, as illustrated by Figure 62.

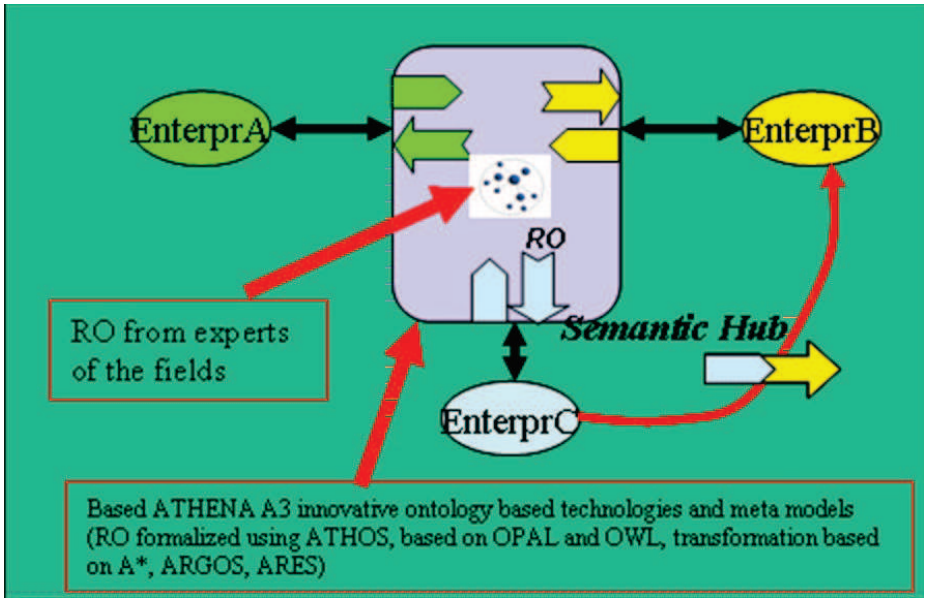


Figure 62: Semantic Mediation by means of ATHENA A3 project

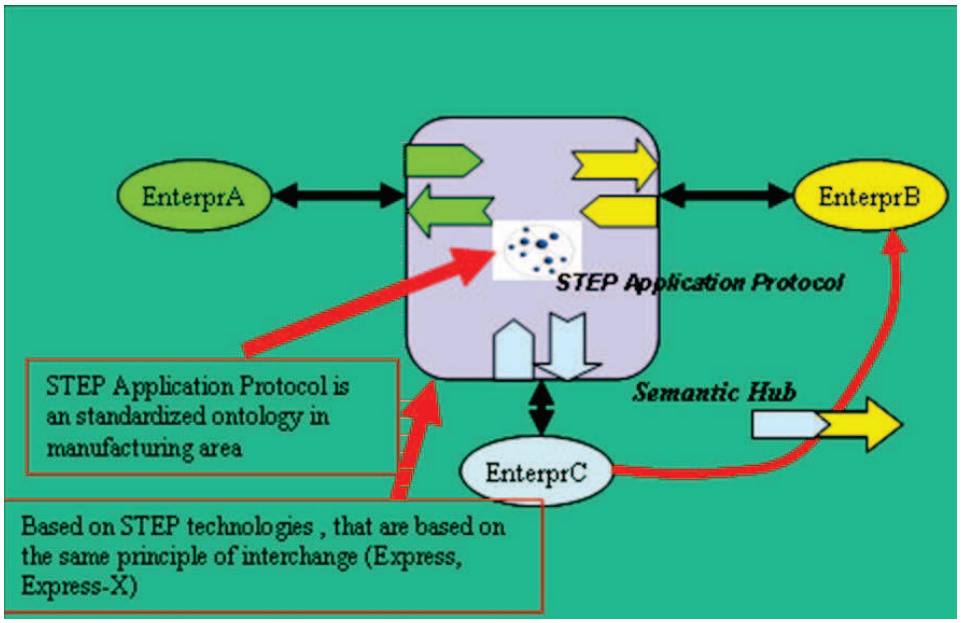


Figure 63: Semantic mediation by means of STEP technologies

To compare the technologies, it was necessary to ensure that it was possible to translate an application protocol such as AP214 as ontology in OWL (Web Ontology Language) without loosing any information. Functionally, the semantic hub proposed by ATHENA partner is closed to what is already covered by STEP technologies and standards, which is illustrated by Figure 63.
As STEP technologies are mature technologies, it was necessary to ensure that proposed formalisms to describe business semantic will not lead to loss of information, but also to assess robustness of proposed technologies when defining huge ontologies such as application protocols, when transforming these ontologies and when transporting these ontologies.
As partners developing solutions had no competence in STEP, and in order to be able to validate emerging semantic web technologies in advance, before availability of ATHENA's demonstrators, I decided to define a mapping from EXPRESS to OWL, and to develop a tool implementing the transformation, the STEP mapper using PERL language. It was a "textual" transformation based on regular forms, from EXPRESS (file format defined by STEP for schemas) and Part 21 (file format defined by STEP for data) to produce OWL ontologies using RDF-XML file format. Doing so, relevant test data sets were prepared for manufacturing business cases to assess ATHENA solutions. In addition, it was the opportunity to measure impact of emerging semantic web standardized technologies. Figure 64 described the targeted demonstrator.

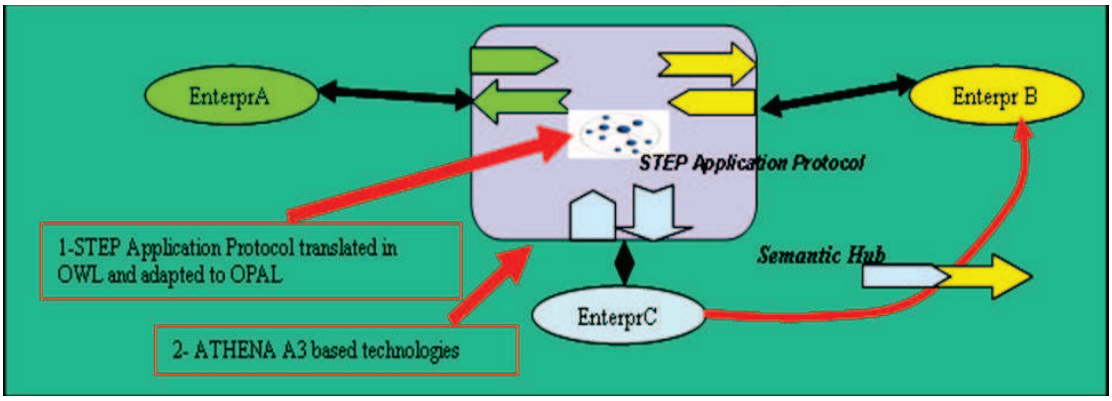


Figure 64: ATHENA semantic mediation for product data exchange

In addition, the same STEP mapper was used in order to define transformation in stereotyped UML model (defined by AndroMDA for Enterprise Java Beans Entities generation) using XMI file format. Figure 65 illustrates the different mappings I defined and transformations I realized in order to support model driven engineering of application reusing an application protocol, as well as semantic mediation.

The target was ability to project business logic on different execution platforms, based on heterogeneous technologies, being application servers, service oriented execution platforms or semantic web. From an application protocol, it was the opportunity to create a knowledge base in OWL, a computer independent model in UML, a subset of platform independent model for designing application, platform specific model for code generation on targeted execution platform, and finally executable components of the demonstrators which can be further reuse.

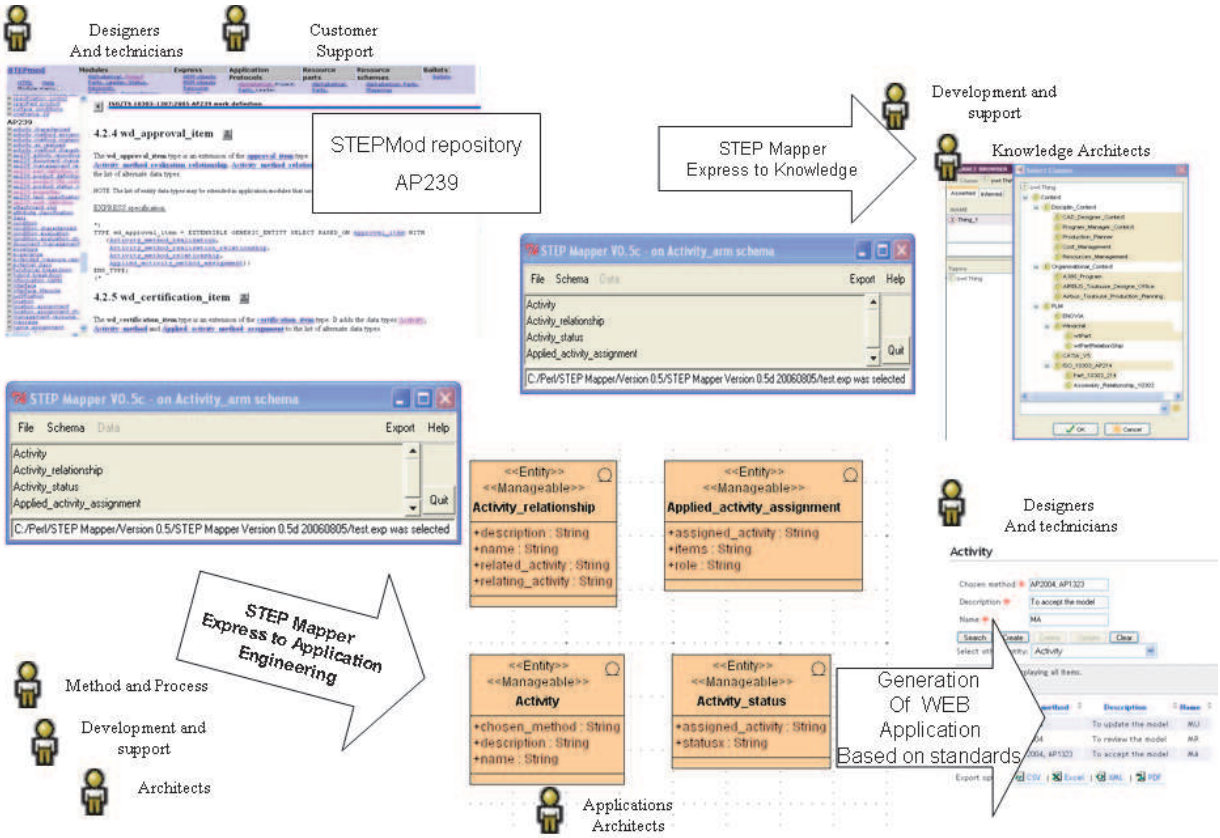


Figure 65: STEP Mapper

Cross organizational change management

This pilot was aiming to assess solutions for cross-organizational change management workflow, with coordinated usage of internal private workflow engines inside each concerned enterprises, with a shared view of the collaboration. This is illustrated by Figure 66.
The different workflow engines used are:

- Enhydra Shark, based on Wfmc Standards, within the collaborative space,
- Internal workflow engines of the Enterprise Resource Planning tools (BAAN, SAP) or Product Data Management tools (DS Enovia) fot the two collaborating enterprises.

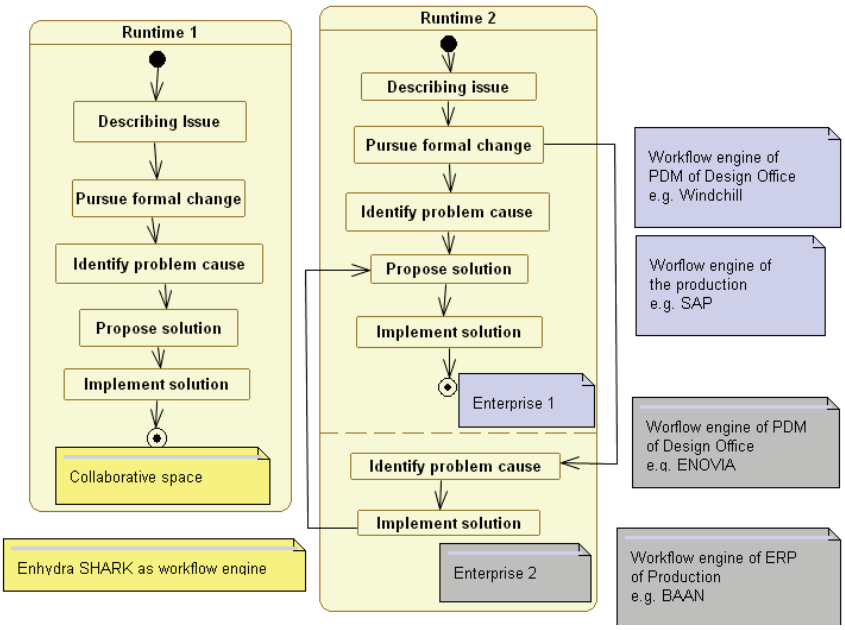


Figure 66: Distribution of one Collaborative Process Instantiation at Runtime 2

In addition, the escalation process for change management within a collaborative networked organization was defined considering two partners (EADS as enterprise 1 and LG as enterprise 2) and an industrial program (Virtual Space Bird program), as illustrated by Figure 67. When an issue occurs, it can impact only the configuration item managed by one partner, or it can impact some others. In such a case, the change process is not internal to one company, but internal to the program. So we have to deal with some escalation. The concerned workflow is no more internal to one enterprise, but external. It is then managed within the collaboration space, with dedicated and standard based workflow system.

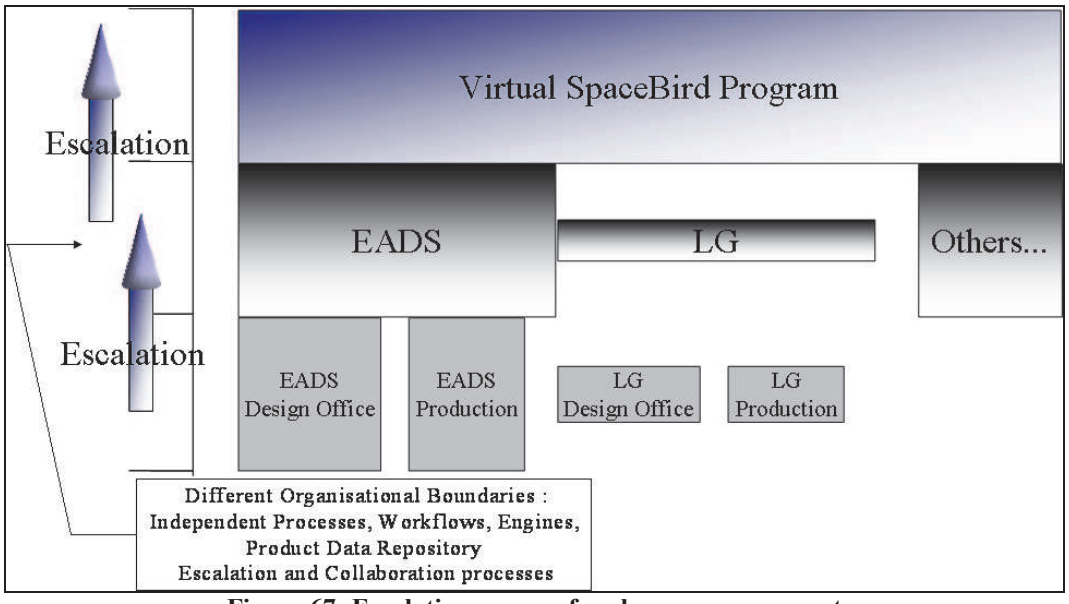


Figure 67: Escalation process for change management

An innovative solution component was provided by ATHENA, allowing defining a collaboration public view combining local public views of private workflow processes. It was based on usage of Business Process Execution Language (from OASIS), but not related to existing workflow standards despite my recommendations. If very innovative, the solution was planned to be made available neither to the partners nor at open source solution. It was

consequently difficult to rely on it for future solutions. It is the reason I also investigated an alternative scenario based on usage of XML Process Definition Language (from Wfmc) was also investigated, based on usage of open source implementations, Jawe (workflow editor) and Shark (Workflow Engine). The test scenario was collaboration between two persons within different enterprises, working on an instance of a change process. It is illustrated in Figure 68. Dominique and Jane have to deal with a change request following the standard process, supported by two collaborating workflow systems.

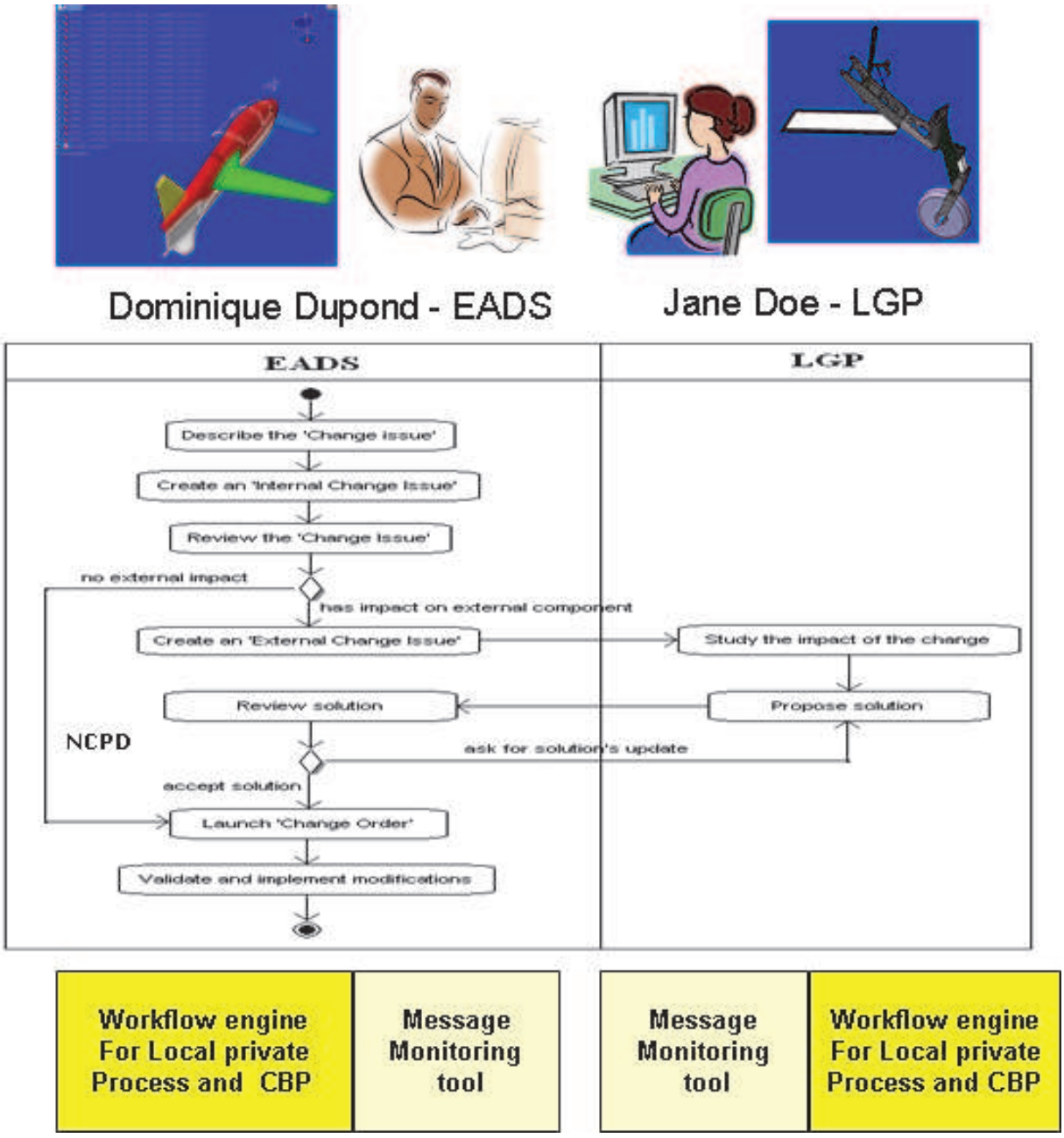


Figure 68: Change process with Executable collaboration process

Networked Collaborative Product Development

This pilot was aiming to assess solutions for Networked Collaborative Product Development, including specification of the collaborative infrastructure, and allowing assessing different transformation and communication services between the different platforms.
The STEP mapper was addressing need to reuse application protocols as what is considered as Computer Independent Model, and to produce then subset of Platform Independent Model which will be used.
Trying to use ATHENA results, I had a lot of problem due to used versions of UML (Unified Modeling Language) and XMI (XML Model Interchange), which were different between the partners and not reusable by open source UML modeler. The UML modelers used by ATHENA partners were software products providing not correct XMI files.
Finally, I used an open source MDA case (Model Driven Architecture based case tool) called AndroMDA, in order to "project" application protocols on an execution platform including an application server based on Enterprise Java Beans standards version 3. All the attributes were converted as strings in this first pilot, and relationships between entities were initially not covered. It was nevertheless a first opportunity to assess Enterprise Java Bean standard and available implementations.

10.5 - Analysis of ATHENA results

Numerous issued were identified concerning the way the ATHENA program addressed interoperability, which were used in when identifying interoperability brakes and issues (c.f. Chapter 5- Identified brakes and issues for Interoperability). The issues I had to face were disruption related to innovative solutions, no usage of open standards, no configuration management for the different used standards and technological silos. For this last point, an example comes from the important difficulties ATHENA faced when willing to integrate semantic demonstrators with service oriented platforms, when willing to transform XML messages formalized with XSD in RDF.
Other important difficulty came from the fact the research was more based on Information & Communication Technologies solutions for simple document exchange, but not for exchange of models and product data based on PLM business standards. As these standards are providing huge ontological models for complex model, I had to face some scalability issue with the components provided by ATHENA partners.

I consequently decided to apply the approach promoted by ATHENA in order to produce my own platforms and environments. In order assessing solutions provided by ATHENA partners, I defined a service oriented execution platform based on commodities on the web and transformation tools to be able to generate demonstrators allowing me to use Application Protocols as ontology for semantic mediation and as UML model for collaborative platform generation. Through the development of the STEP Mapper, I was able to master OWL and to have a first idea about some issues when mapping a language to another (impedance mismatch). I was also able to use open source implementation of UML 1.4 within the projects. As existing tools were not correctly supporting UML 1.4, in particular annotation which are absolutely required for Model Driven Engineering, I decided to investigate emerging version of UML, the version 2, and the associated grounds. The OpenDevFactory project was the opportunity to study the Eclipse platform with UML modeling plugins as ground for UML2 and Model Driven Engineering.

Chapter 11: UML Ground development through OpenDevFactory

OpenDevFactory is a sub-project of the "Usine Logicielle" ("Sofware Factory") French research project dealing with production of open platform based on Eclipse and on open standards to support collaborative System Engineering and Simulation Life Cycle Management.
This project was the opportunity to assess "Papyrus UML", a UML modeler based on Eclipse and integrating numerous plug-ins for model transformation (Eclipse Modeling Platform and Action Transformation Language plug-in from INRIA) as an open source Model Driven Engineering ground.
It was also the opportunity to investigate deeper Model Driven Architecture and Unified Modeling Language standards, as using an open source platform supporting fully Object Management Group's standards such as UML 2 (Unified Modeling Language), subpart of Meta Objet Facilities, model transformation, SysML and Diagram Interchange.
The approach adopted for ATHENA was extended, considering standards such as SPEM (Software Process Engineering Meta model), with investigation of Eclipse Process Framework.
Finally, as looking for a way to reuse standardized specification formalized with XML and XML schemas, it was the opportunity to discover the "hypermodel" software and to start to formalized conceptual approach based on "extended hypermodel for transportable models with semantic preservation".

11.1 - Engineering context

Engineering context was related to « System Engineering factory for the enterprise ». The idea developed in this project is to define a working environment for (industrial) systems development, from specification to development, with validation, tests and simulations. Such environment should allow an efficient collaboration for engineers and enterprises involved in aerospace products development. The types of models to manage in configuration are specifications and functional descriptions of the product, including real time and logical time models for the different concerned systems. Of course, the platform should support usage of models created using heterogeneous tools.
The simulation lifecycle management problem was related to multi-disciplinary engineering, multi-partners development and usage of multiple tools, with need to manage in configuration a set of various models and documents, and to be able to exchange, share, transform and retain them all along the lifecycle of the product.
I had to evaluate some components provided by the partners. The first component was a "process" component. This component was based on a software product built on top of Eclipse framework and supporting SPEM (Software Process Engineering Meta model). The goal was to be able to establish interchange of information between this tool and a project management tool with synchronization.
The second component was a transformation component, based on ATL plug-in (Action Transformation Language). ATL is a language developed by INRIA, which was developed before the OMG's specifications for a standardized model transformation language, the Query View Transformation language. However, the ATL philosophy is closed to QVT. In addition, no open source implementation were existing at the time of the project
The last component was the software factory integration platform based on Eclipse: the "Papyrus UML" component.

In addition to the evaluation of the components provided by project partners, I also had to develop a composite component called "interoperability", allowing evaluation of transformation component and Papyrus UML component.
Figure 69 provides an overall picture of the environment for which the interoperability is wished. The simulation engineering platforms, which are authoring tools, are to be interconnected with other technical enterprise and authoring applications, being for management of the collaborative simulation, requirement engineering, product lifecycle management, simulation lifecycle management, computer aided engineering tools (CAE), Digital Mock Up (DMU) definition tools and finally execution platforms constituting simulation workbench.

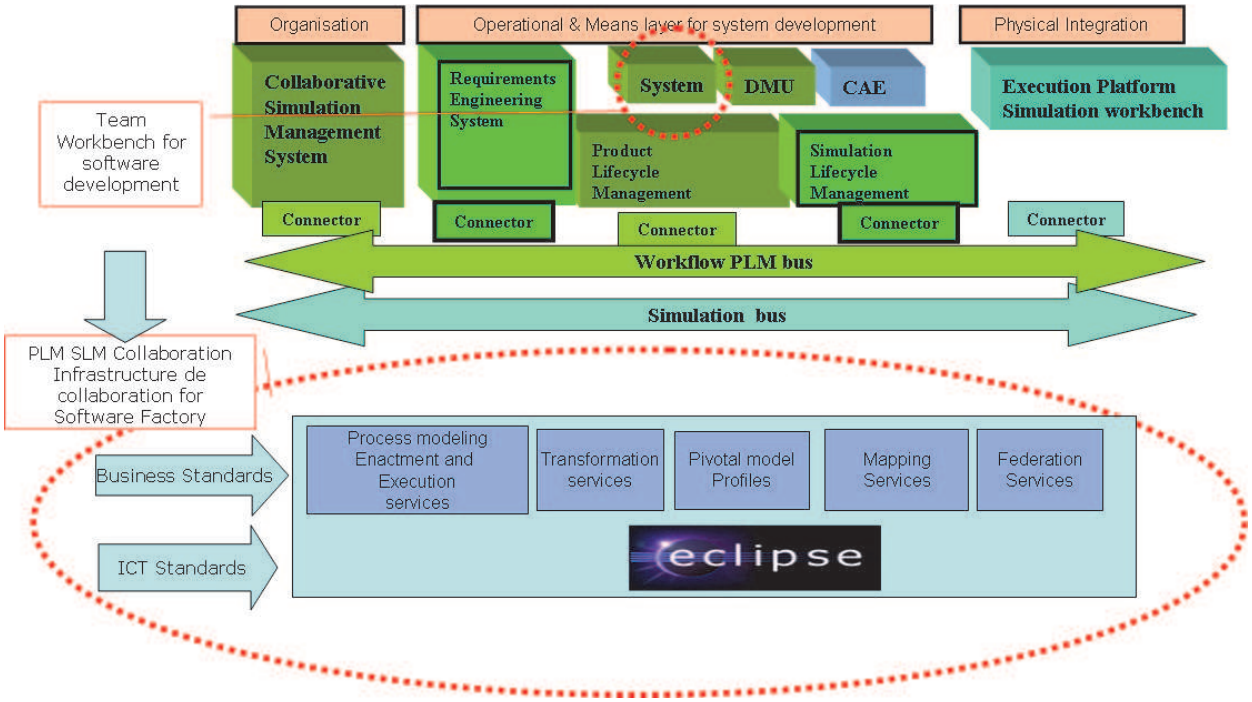


Figure 69: Engineering environment considered within OpenDevFactory

On top of Eclipse, a number of components and services were provided, based on business standards and ICT standards, supporting collaboration between teams with interconnected and interoperable platforms.

11.2 - The collaborative platform

In order to set up the collaborative space, the main parts of the process were concerning setting-up of the environment (through assessment of components) and operational usage.

The defined high level collaborative infrastructure is described by the model represented by Figure 70. The functions of the enterprise that are considered are respectively , the design, the production, the customer support and the ICT support, which are respectively attributed to the design office, the production, the customer service and the ICT department. A complementary function is proposed, the Product Lifecycle Support, which is dealing establishment of digital collaboration. The role of this function is the "controlled urbanization" of the technical information system and associated digital means. His role is to establish and to maintain a networked collaborative product development infrastructure. It is proposed to support his

activity by means of a network governance platform, an interoperability platform, an application modeling platform, an application development platform and a service oriented execution platforms. All these platforms are based on open standards.

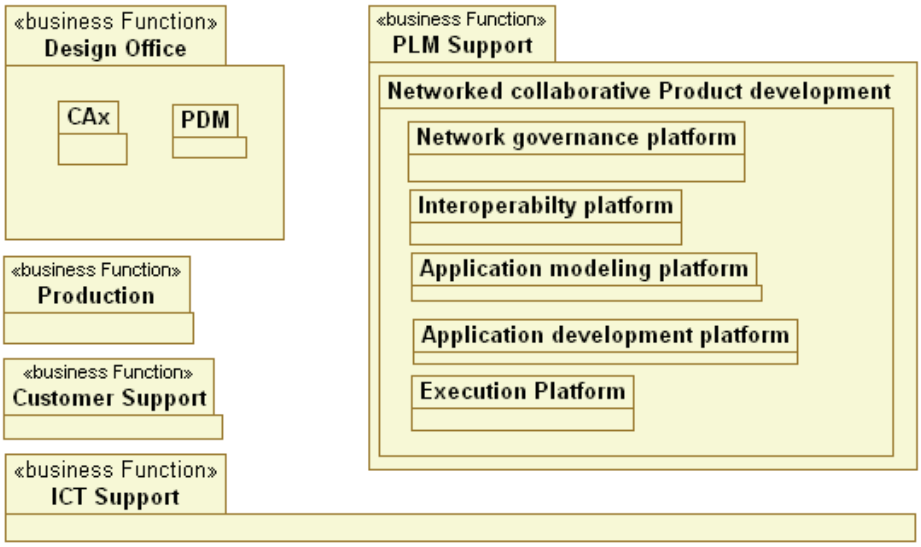


Figure 70: High level Collaboration Infrastructure considered for OpenDevFactory

The network governance platform (application of the principle 10) is based on set of UML model profiles based on enterprise modeling languages. Numerous languages and standards are proposed, but this domain is not yet very mature. In addition, it is difficult to find free resources in order to deal with them. For this reason, two freely available modeling languages were evaluated, coming from ARCHIMATE and COMET research projects. ARCHIMATE became an Open Group Standard after OpenDevFactory. Enterprise modeling aims to define as-is situation and to plan to-be situation, in order to establish a convergence plan. In addition, the urbanization rules are established in order to ensure efficient digital collaboration and interoperability of technical enterprise applications.

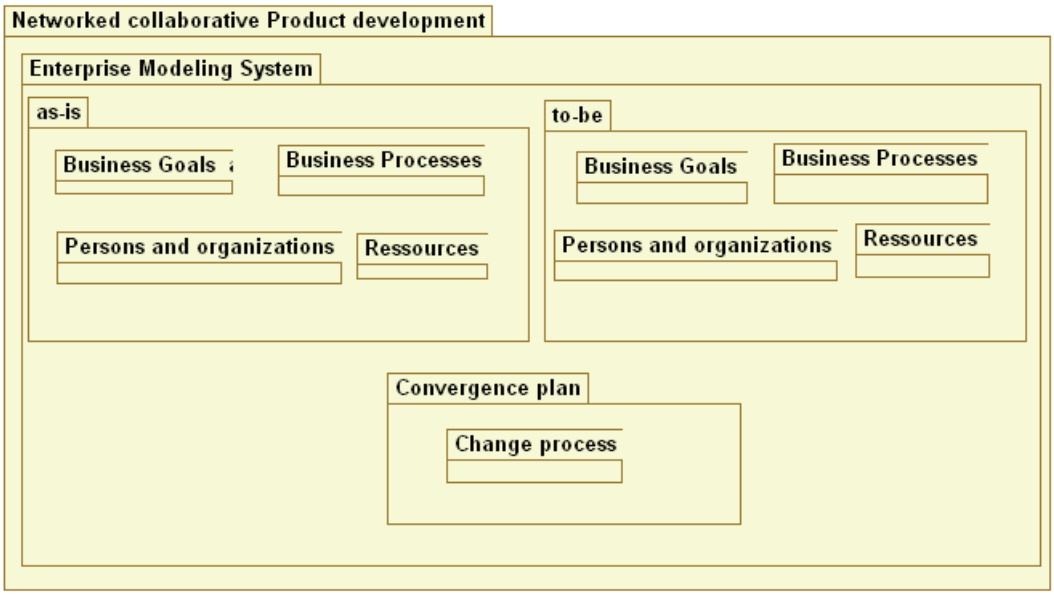


Figure 71: Governance Platform formalized for OpenDevFactory

As this platform is based on models produced with open and standardized models, these models can be reused for automated generation or validation of the parameterization of

enterprise applications. It is also possible to ensure coherency of the different models, computer independent enterprise models and platform independent models used to design and to parameter the enterprise applications.
All the platforms are based on model driven engineering, systematic usage of open standards and functional component which are available as commodities on the web implementing the identified relevant standards. Figure 72 display the model of the platform, which are composed of applications supporting implementation and modeling standards.

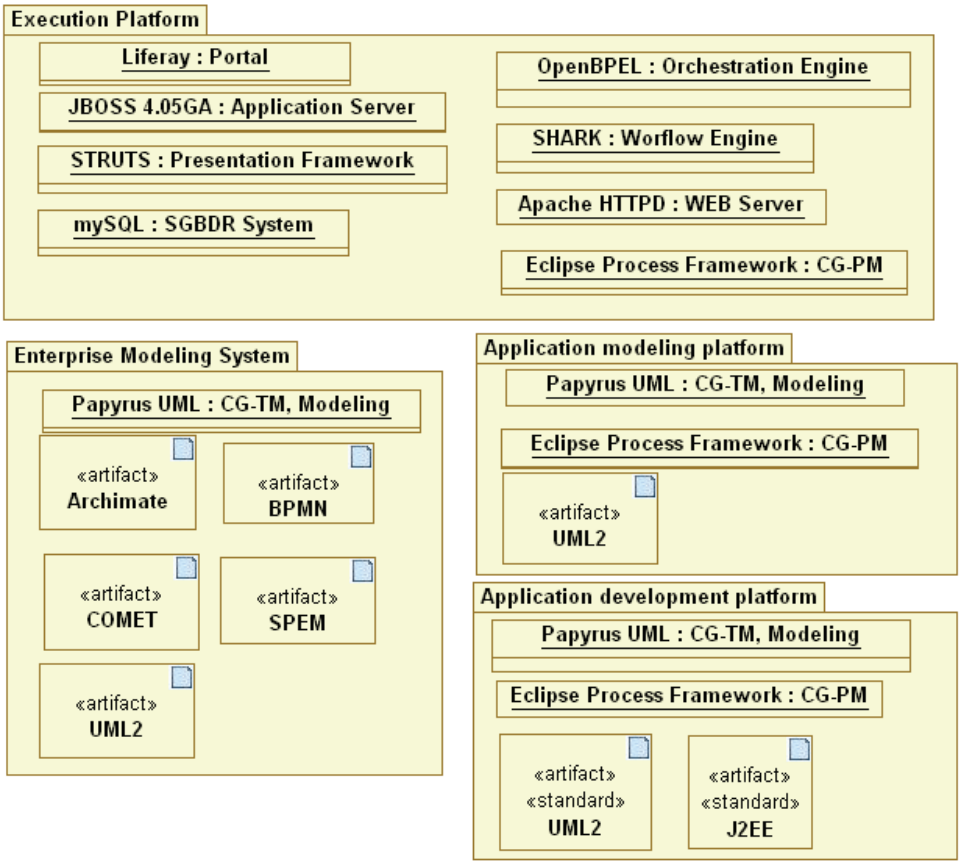


Figure 72: Concrete platforms based on open standards for OpenDevFactory

The generic model of the targeted execution platform is based on generic functions addressed by standardized specifications. Doing so, for a same platform, we can interchange implemented components and insure that the produced models and information can be easily exchanged and reused with other platforms. Figure 73 shows as an example the model of the generic execution platform used for OpenDevFactory. The execution platform is based on an application server based on Enterprise Java Bean 3, a workflow engine based on Worfklow management cohalition standards, a relational database system (RDBS) based on SQL, a portal based on Web Service Remote Portlet specification and Java Service Recommendation 168, a WEB server based on HTTP standard, etc. The architecture of the platform is made by federation of functional components with standardized specifications and available implementations.
The concrete platform is only one of the possible implementation. Any platform based on such components will be able to interchange with other similar platforms more easily and at a more reasonable cost. It includes new platform but also legacy platforms that are already in operation. Such situation is the one encountered within extended and virtual enteprises.

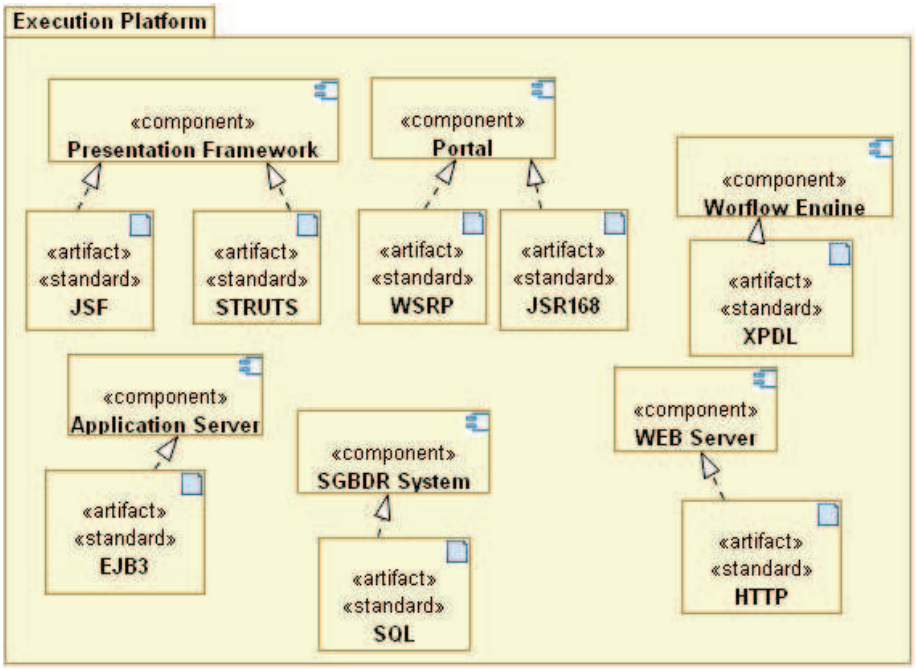


Figure 73: Standards based generic execution platform

All the platforms are based on model driven engineering. OpenDevFactory was the opportunity to assess OMG's standards allowing to deal with model driven approach and eclipse as an open environment implementing open standards allowing to support such an approach. Papyrus UML was consequently studied as a meta-modeling environment allowing federating different standards and standardized language. It was the objective of the "interoperability" platform I add to produce as an OpenDevFactory Component. This component is described in the next section.

11.3 - More on the "interoperability" component of the platform

The aim of this "interoperability" component was to be able to import and export legacy models from models repositories containing Platform Independent Models, Platform Specific Models or auto-description of a running system.
Aggregation of Computer Independent Models (CIM), Platform Independent Models (PIM) or Platform Specific Models (PSM) should be supported. Finally, it should be possible to generate code, to deploy and to enact executable model on targeted service oriented execution platforms.
From existing legacy models, formalized with dedicated specific languages, the first need is to be able to put the models and used languages on the same meta-modeling platform by mean of text to model translator tools (T2M). Then model to model transformation should be used in order to be able to support the model driven engineering strategy but also to ensure some aggregation and federation services supporting iterative development. Finally, it should be able to export the results using model to text translators in order to comply with dedicated syntax of external tools. To support such approach, constitution of a Reference Model Library is required, containing description of modeling languages as UML profile on the platform. Figure 74 illustrates the usage of the interoperability component.

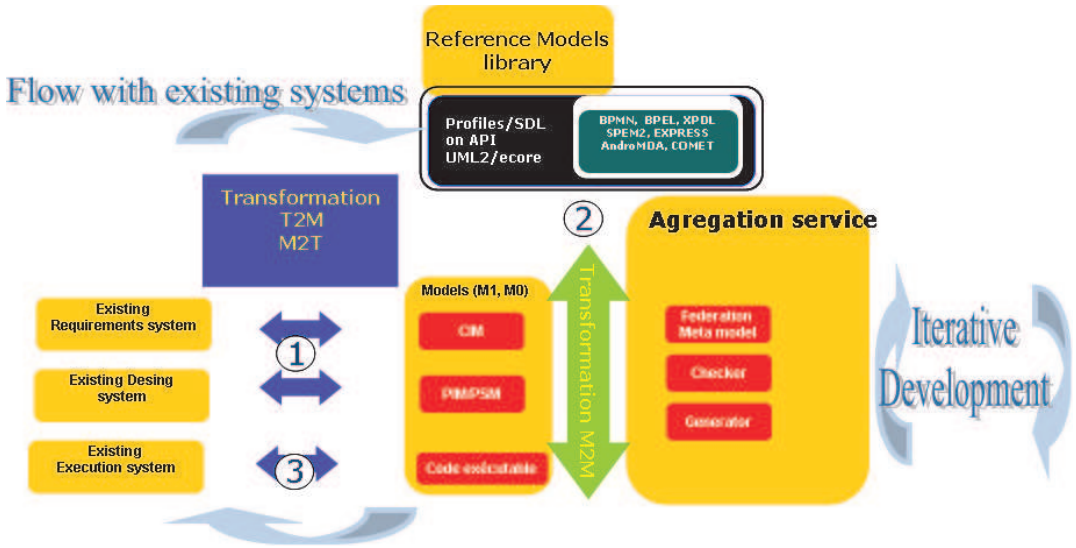


Figure 74: Interoperability component usage in OpenDevFactory

The library of models is provided through the meta-modeling environment provided by OpenDevFactory, as Unified Modeling Language profiles or as Eclipse Modeling Framework models. Gateway is established with Semantic Web through Web Ontology Language and with Product Data Exchange through EXPRESS. In addition, a library of transformation component is also required in order to be able to make translation of a model from a language to other languages, in order to validate a model or in order to complete a model.
The used language to describe model to model transformation is ATL. If some other transformation services are freely available on the WEB, they are reused. It is the case for example with "hypermodel" which allows transforming XML Schemas in UML (Unified Modeling Language) models. It was used in particular in order to reuse languages such as Web Service Description Language or XML Schema in UML, and then, through creation of ATL mapping, in a UML profile. As the idea was to reuse available components, it was required to validate the chains of transformation and to assess quality of the transformation. At the end of the project, a set of UML 2 models and profiles were available with or without associated diagrams, as illustrated by Figure 75.

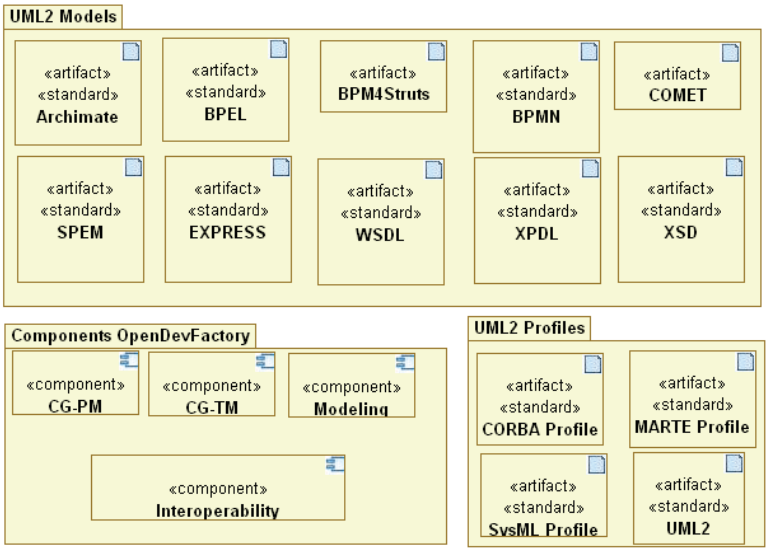


Figure 75: Model repositories produced with and for Interoperability Component

Table 12 describes what I achieved in term of profile production and evaluation of transformation chains.

		Profile UML2	DSL eCore	DSL KM3	Express	XMI
	COMET Business	x C3				
	COMET Requirements	x C3				
	COMET Service Architecture	x C3				
	AndroMDA Process					
	AndroMDA Service					
	AndroMDA Persistance					
	AndroMDA Presentation					
	AndroMDA Web Service					
	BPEL	C1				
	BPMN	C1				
	XPDL	C1				
	EXPRESS	C2, C5	x	x		x
	XML Schema	C1				
	WSDL					
	IDL					
	EJB					
	CCM					
	CMW					
	FERA					
	AP233, AP214, AP239					
	SysML	x				
	PDM Enablers					
	PLM Services (OMG Mantis –PLCS)					

Transformation chains
C1: XSD => UML 2.0 =>UML 2.1 => Profi
C2: XMI Poséidon => UML 2.1 => Profil
C3: manual modeling Papyrus 1.7
C4: Magic Draw export to EMF
C5: XMI => KM3 =>eCore => UML2 =>Profil

Legend
Model and Diagrams Available: x
Diagram available: x

Table 12: Available models and diagrams from OpenDevFactory

As an example, Figure 76 provides an extract of COMET profiles created on Papyrus in order to be used as one potential enterprise modeling language.

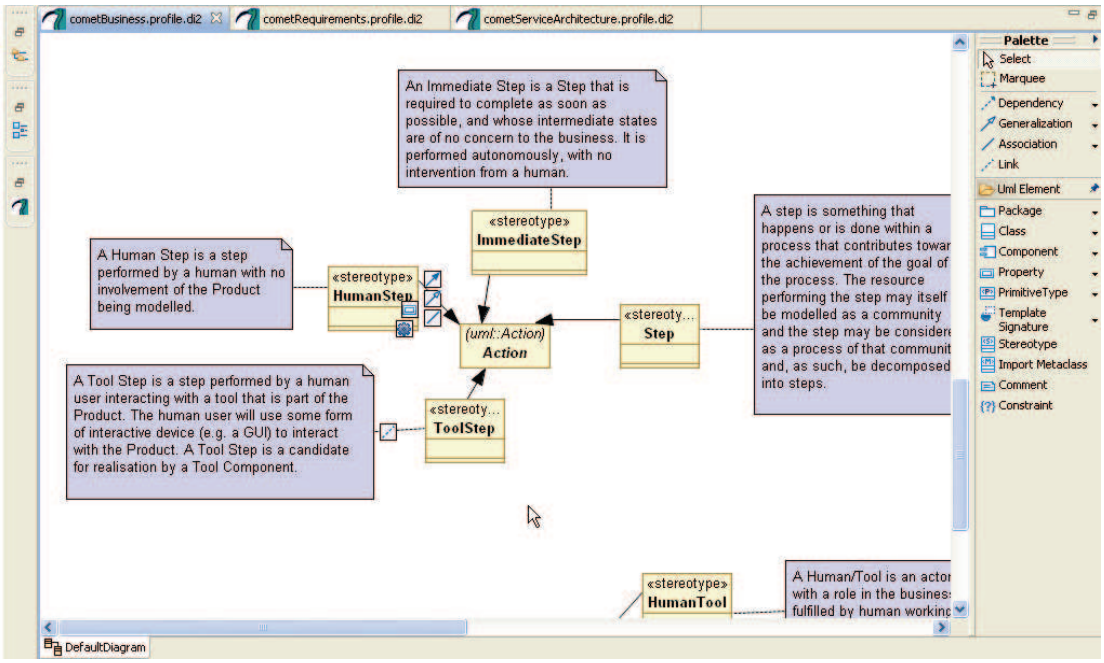


Figure 76: COMET Profile in Papyrus

11.4 - Lessons learnt from OpenDevFactory

The work undertaken for my thesis within OpenDevFactory project allows me to enrich and to improve what was achieved through ATHENA, and to apply it to another context, the Simulation Lifecycle Management.
Improvement came from usage of advance modeling tools dedicated to system designers when willing to formalize platforms and their environments. Enrichment came from deep study of UML2 ground based on Eclipse platform. This study allowed pointing out several important aspects to consider when dealing with interoperability of enterprise applications.
A first aspect is related to usage of standardized graphical languages in conjunction with textual standardized language. The fact that UML was initially a diagramming language and not a modeling language has important impact on its usage when dealing with model interchange. Visual representation of a model, which is dedicated to human, is particularly important when dealing with complex systems. The study led me to investigate impact of visual modeling on interoperability, in particular when dealing with semantic cartography, dynamic user interface generation and presentation layers. It also allowed me to discover hypermodel concept, and to develop the concept of extended hypermodels for interoperability, with deep knowledge of UML 2 ground.
The experimentation also points out importance of the qualification of transformation chains and of all the configured components which are used, in particular in term of standard compliance or support. As an example, the component I used for code generation, AndroMDA, was robust for UML 1.4 but not with UML 2. It is the reason why extended hypermodels for interoperability insists on qualification of configured set of versioned standards and implementation components.
Finally, availability of open source libraries of reference available with the different standardized languages and meta-modeling platforms changed the way to consider data transformation and information sharing. It initially the technologies used were mainly focusing on transformation of data using a neutral format and schemas, the question arose to think about development of in memory model to model transformation taking advantage of Meta Object Facilities proposed by Object Management Group. The ground for such standardized technologies is today mature.

As a perspective, the project also allowed to discover SPEM (Software Process Engineering Meta model), and associated plug-in Eclipse Process Framework, with a new kind of process model, related to methodology formalization (while XML Process Definition Language deals with workflow models, Business Process Execution Language with executable processes and BPM4STRUTS with human/application interaction). SPEM is particularly accurate when willing to support interoperability and collaboration between engineers working with authoring tools. I didn’t have the time to include SPEM within the proposed approach, but interesting perspective is to extend the approach to authoring tools which are computer aided and used to support system engineering.

The SEINE project was the opportunity to apply the ATHENA and SEINE results within an industrial environment, and to validate the approach for real business cases.

Chapter 12: PLM standards assessment for Aerospace Collaborative Hub through the SEINE Project

12.1 – The SEINE context

The SEINE project is an industrial project, where I was involved as PLM standards expert. Participating to this project was the opportunity to identify operational business needs and to promote the approach developed for interoperability.

The SEINE project was launched in order to address the digital break within the extended and virtual enterprise within the PLM area.

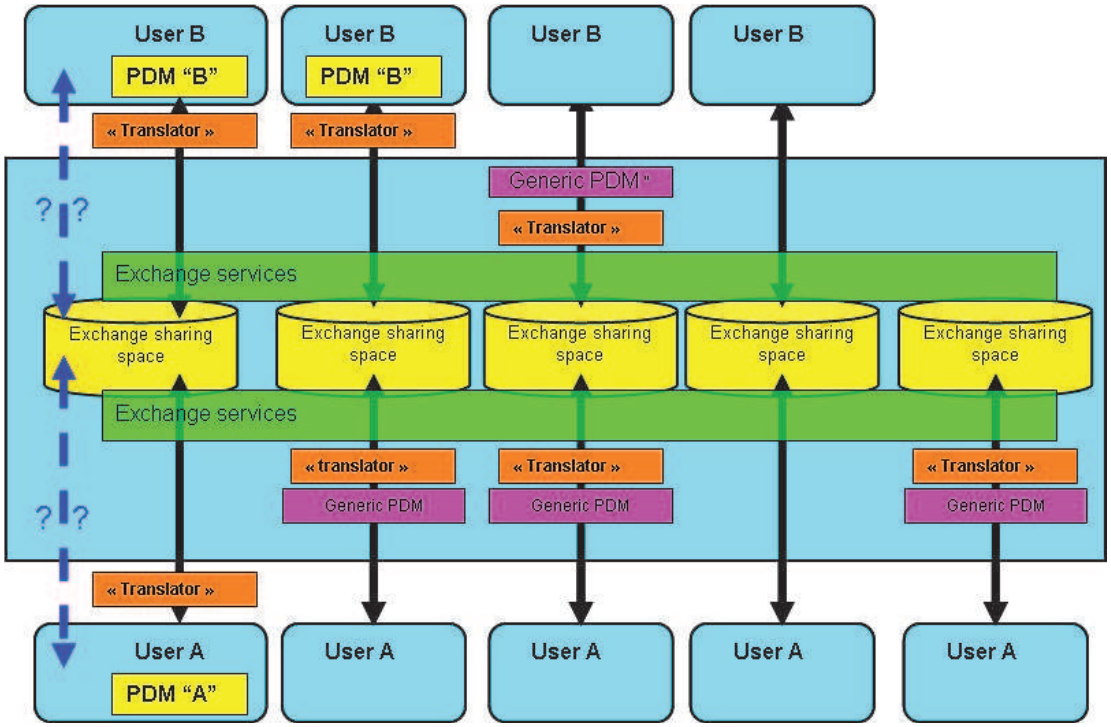


Figure 77: SEINE targeted collaborations

So the idea was to identify the relevant PLM standard for the European Aerospace community in order to ensure eBusiness collaboration for collaborative product development, based on usage of Computer Aided activities, implying partners with or without PDM systems as described in Figure 77.

For such collaboration, it was required to select a set of business needs to address with level of priority defined by industrial partners (c.f. Figure 78), and to define a cartography of collaboration use cases (c.f. Figure 79).

	UseCases / Processes	Gestion des exigences	Gestion des configuration	Concurrent engineering	Gestion des modifications	Validation et vérification	Gestion de la qualité produit	Certification numérique	Infrastructure & Architecture Technique sécurisée	
	Priorities	*6*	*5*	*1*	*2*	*4*	*3*	*8*	*7*	
1	Supplier Portal Extension to PLM								■	1
2	Context Delivery			■						1
3	Engineering Data Package (Engineering or Manufacturing)			■						1
4	Lockable Engineering Data Package (Engineering or Manufacturing)			■						1
5	Multimedia Collaboration Environment			■						1
6	Collaborative Review Environment			■						1
7	Components Catalogs			■						1
8	Collaboration on FAI (First Article Inspection)					■				1
9	Certification							■		1
10	Customer Support		■							1
11	Authoring Trade Off					■				1
12	Change Management Percolation				■					1
13	Customer Modification Impact Analysis				■					1
14	Sub-Contractor Evolution Instruction				■					1
15	States delta		■							1
16	Product analysis / Stress and weight		■							1
17	SME shared workspace offer to client			■						1
18	ASP Shared PDM for multi-site SME			■						1
19	ASP CAx for SME									0
20	Non stop engeenering 24/7			■						1
21	Product Reception Forms						■			1
22	Data protection and IT security								■	1
23	Requirements management	■								1
		22	1	3	9	3	2	1	1	2

Figure 78: SEINE Prior use cases

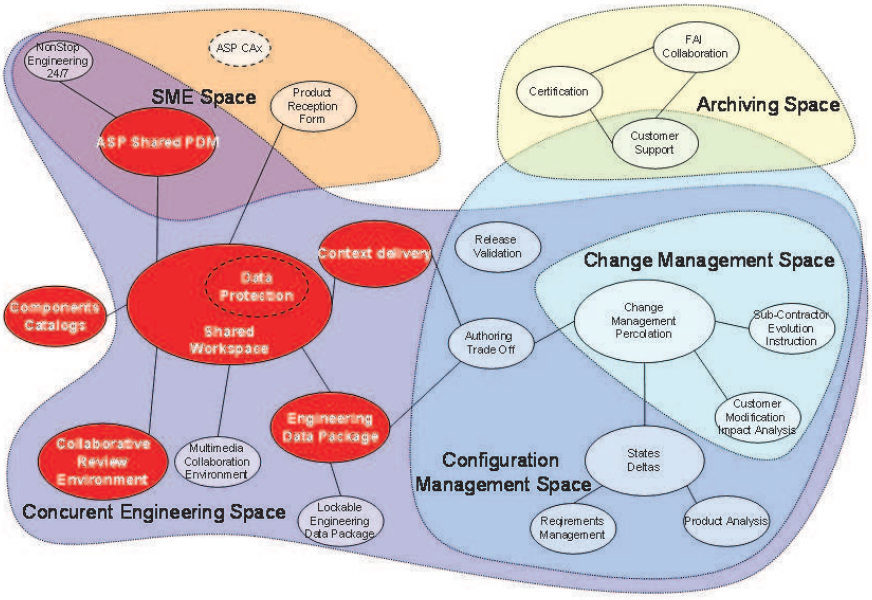


Figure 79: SEINE use cases cartography

This industrial project was the opportunity to assess the defined approach and to apply in particular the principle 10, related to concrete governance of standards of a community, with as a result launching of initiatives aiming to govern and pilots development of PLM standards: the Aeronautic, Space and Defense Standardization Strategic Group.

As standards are complex and difficult to evaluate before to ask their implementation by software provider, I used the results of my research work in order to ensure PLM standards assessment, through fast development of demonstrator using model driven engineering on top of previously identified service oriented execution platforms, particularly adapted to WEB usage and eBusines collaboration. The identified interests are the following. Firstly, it allows to produce implementations of reference which are fully compliant with the standards, even if

not of industrial quality (ergonomy, robustness, etc). Secondly, it allows creating test data sets which will allow evaluating and qualified interface of commercial solutions. Thirdly, it allows validating standard specification playing business scenarios of interchange or sharing, within a real web environment. Finally, it supports more efficient communication and training on standards and on interoperability.

12.2 - What was to be demonstrated

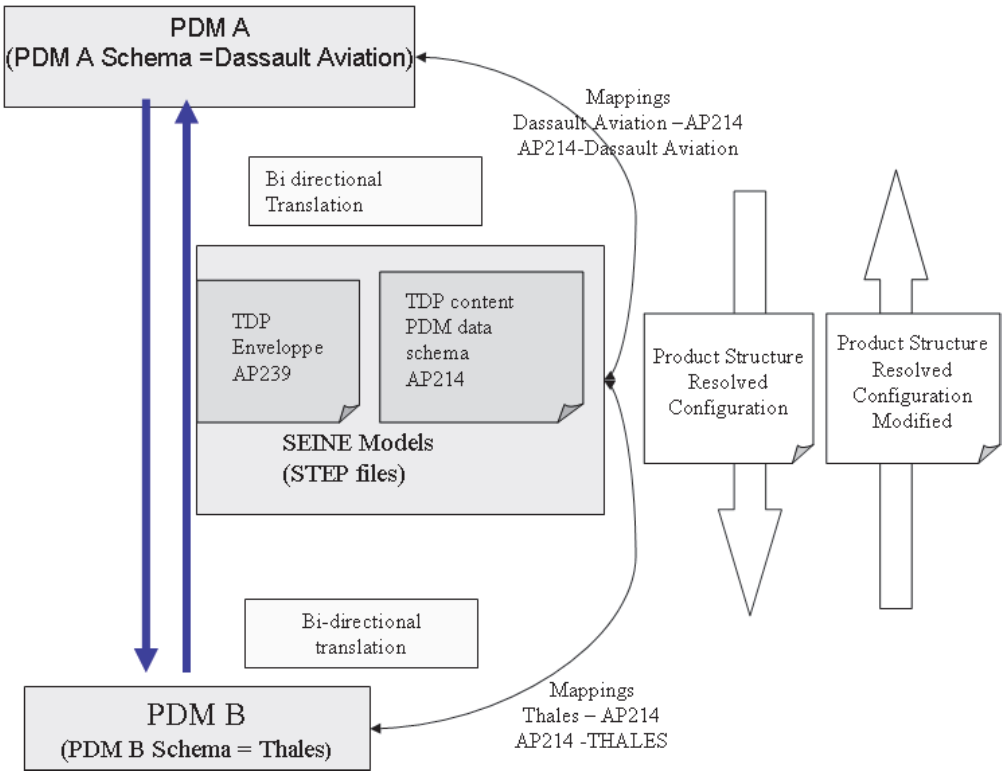


Figure 80: SEINE demonstrator on standards

In order to take a decision on feasibility of usage of standards, it was asked to demonstrate using operational PDM that interchange of technical data package, with dispatch note formalized using business vocabulary defined with STEP AP239, and Product structure with resolved configuration formalized with STEP AP214, was exchangeable between Airbus PDM system, Thales PDM system and Dassault Aviation PDM system.
Cost of such demonstration being very high, it led to some difficulties when willing to launch the development, endangering the project in particular for assessment of relevant PLM standards to use.
It is why I proposed to develop a demonstrator from scratch, allowing demonstrating feasibility of using PLM standards for interchange of Technical Data Package between Thales System and Dassault Aviation.

The time required to realize prototypes was about one month, with creation of three Product Data Servers based on the specific models coming from Thales, on the specific model coming from Dassault Aviation and on the standardized model provided by AP214.

The UML models for each specific Product Data Management system was created by hand, after interview of PDM experts of each company, and using documentation of the used information model.

It was then annotated, and servers were generated automatically using Model Driven Engineering. Access was given to users in order to enter a test data set. It allows to iteratively correcting the model when required, in order to finally obtain two PDM servers with test data sets.

Figure 81 and Figure 82 are extractions of the obtain results: two set of data describing the product structure of a product (Derive F7X) and of an equipment (operator Broussar) were made available on application playing the role of real PDM Systems, with specific vocabulary and models of each of the enterprises.

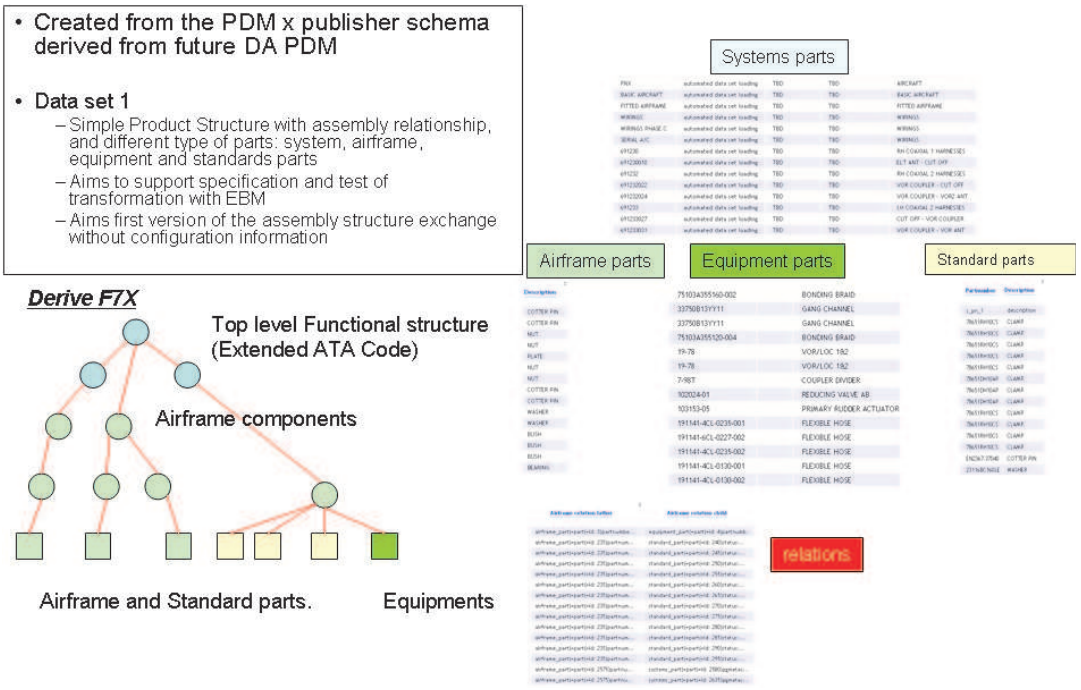


Figure 81: Dassault Aviation Data Set with used terminology

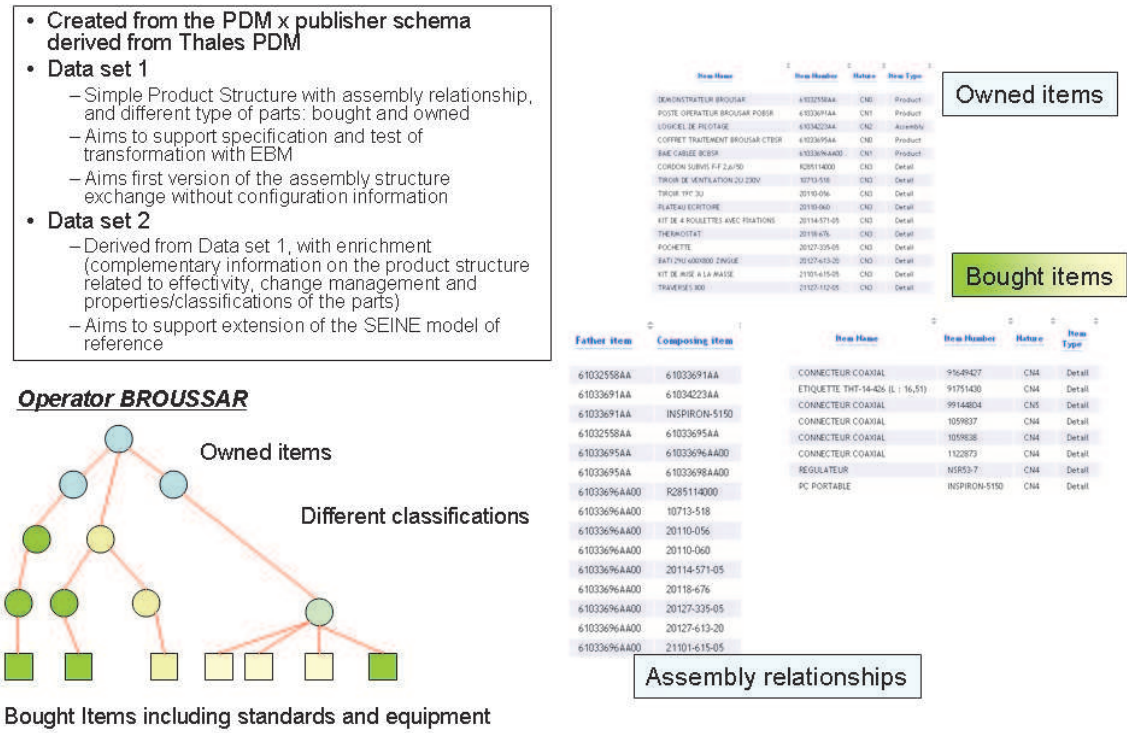


Figure 82: Thales data set with used Terminology

A similar intermediate server was produced as an intermediate Product data repository, in order to be able to validate result of the transformation and in order to help the users to understand and validate the mappings.

What was discovered realizing the demonstration was that interchange between PDM systems was not only requiring translation from one language to one other, but also that some transformations are required as each system is considering different the specific viewpoint of its owner. Identification, typing and configuration management rules are different for each system, leading to consider some transpositions when sharing and exchanging information. This is illustrated by Figure 83, where the breakdown of the equipment is to be transformed according the viewpoint of the integrator before to be attached to the product breakdown. Such a need implies extending transposition services proposed for multi-representation formalized by extended hypermodel, when having a model traveling from a ground to another ground.
Through this pilot, it was pointed out that usage of very structuring databases for each application is an obstacle for data flow between systems. As a consequence, the intermediate data storage could be a place allowing aggregating and federating heterogeneous data, and for which usage of extended hypermodel usage could be very relevant.

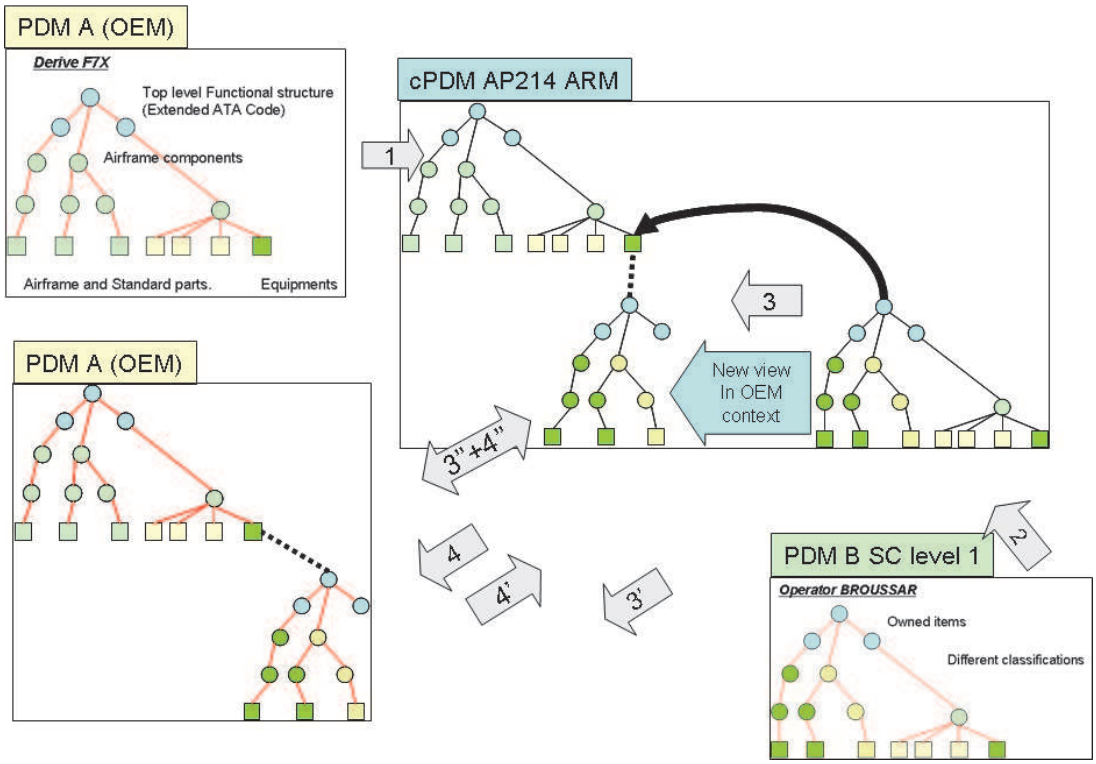


Figure 83: Interchange of data between different PDM

Several issues are to be addressed consequently to be addressed for a collaborative hub allowing interconnecting PDM systems. It includes multi-identification, transformation related to context change, control flow between the systems, creation of new views automation, versioning, effectivity and standard parts libraries sharing.
Interoperability here is no more an ICT problem, but related to the collaboration processes.

12.3 - Collaborative environment set up

The collaborative environment was set up firstly through definition of a set of architectural models of reference based on standardized eBusiness component usage, for the extended enterprise, for the collaboration platform and for the product data servers.
It was a direct application of the defined principles to create a framework for interoperability within a collaborative space.
Firstly, a multi layered eBusiness architecture of reference was produced, considering the community as a system of system, with the networked organization connected to a collaborative space (Figure 84), the collaboration platform providing services for the whole ecosystem (Figure 86), enterprises with their front office applications and back office applications (Figure 85), and finally the application and its interface with actors, external business processes and other the inter-enterprise service bus (Figure 87).

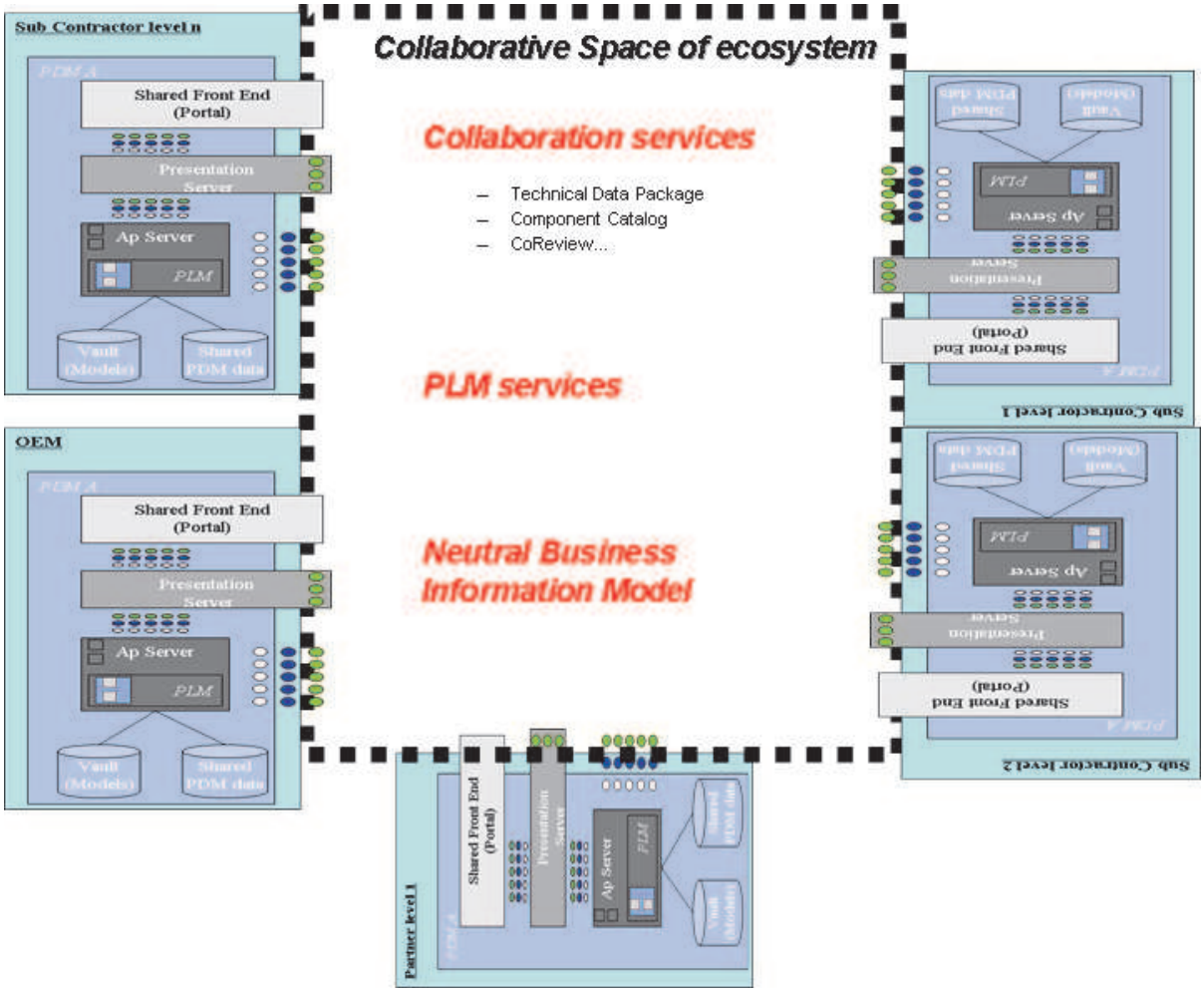


Figure 84: Collaborating ecosystem

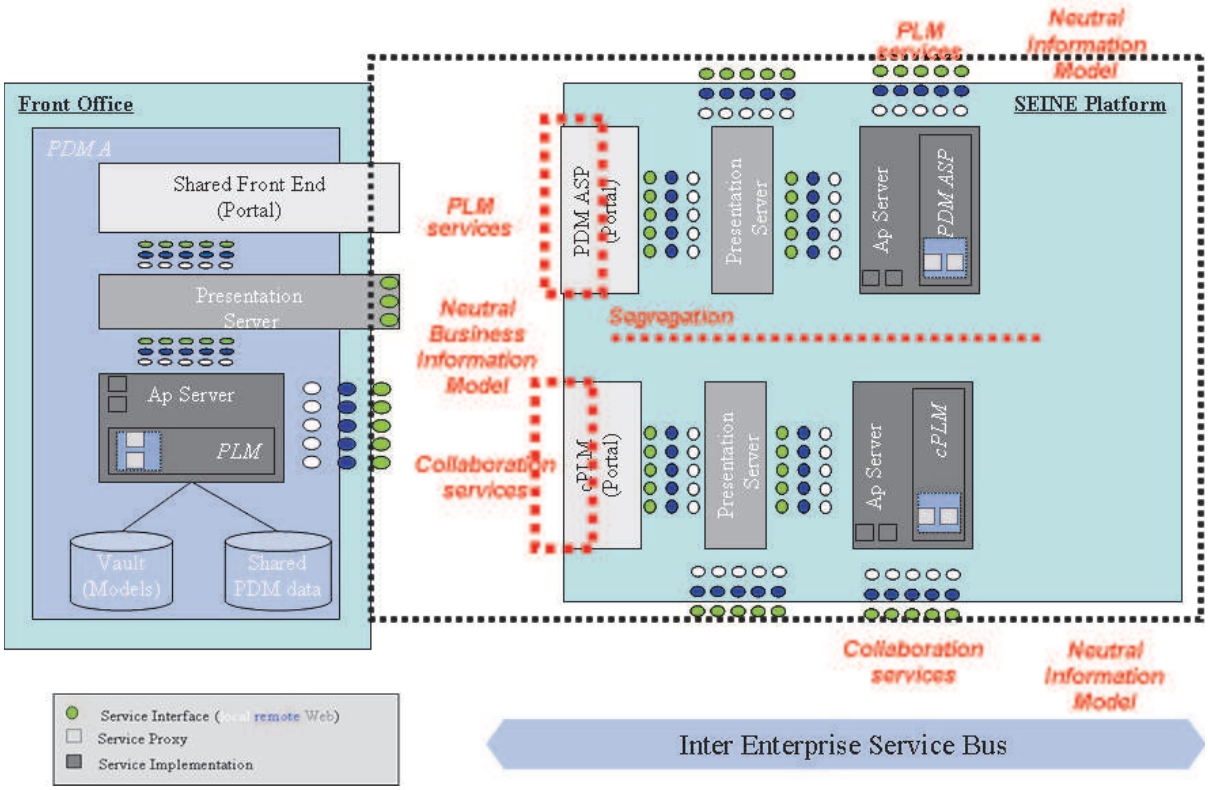


Figure 85: Collaborative platform supporting the network of enterprise

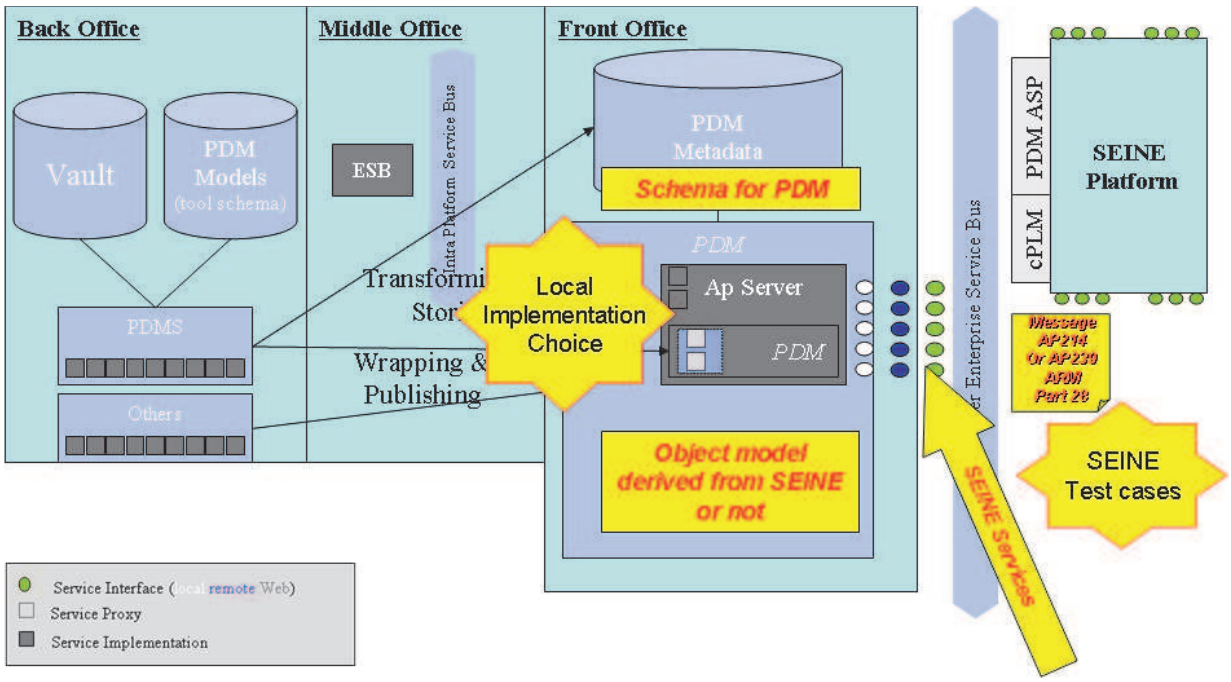


Figure 86: Enterprise applications within the "enterprise system"

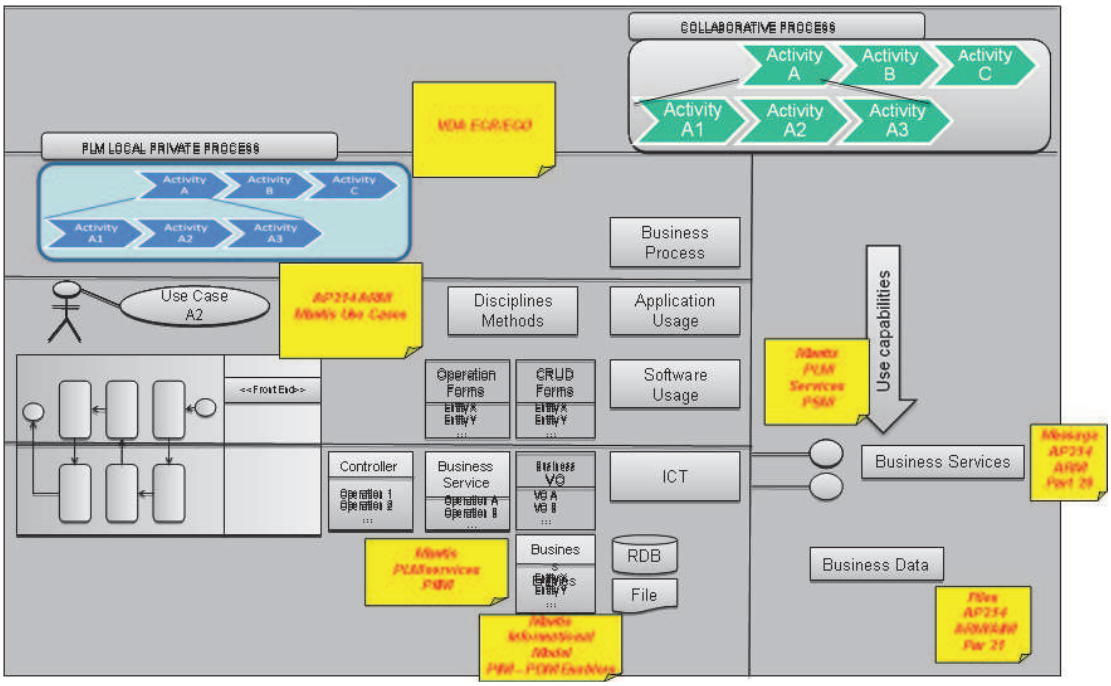


Figure 87: Enterprise application and its interfaces with the outside

Then the development environment was defined, on the basis of a UML modeler based on UML 1.4 and on AndroMDA case tools with accurate profiles for generation on execution environment. A Model driven approach was adopted, allowing projecting Computer Independent Model including standards and specific business models on the components of the produced architecture. (Figure 88) summarize the approach adopted, which is the one established and promoted on the thesis.

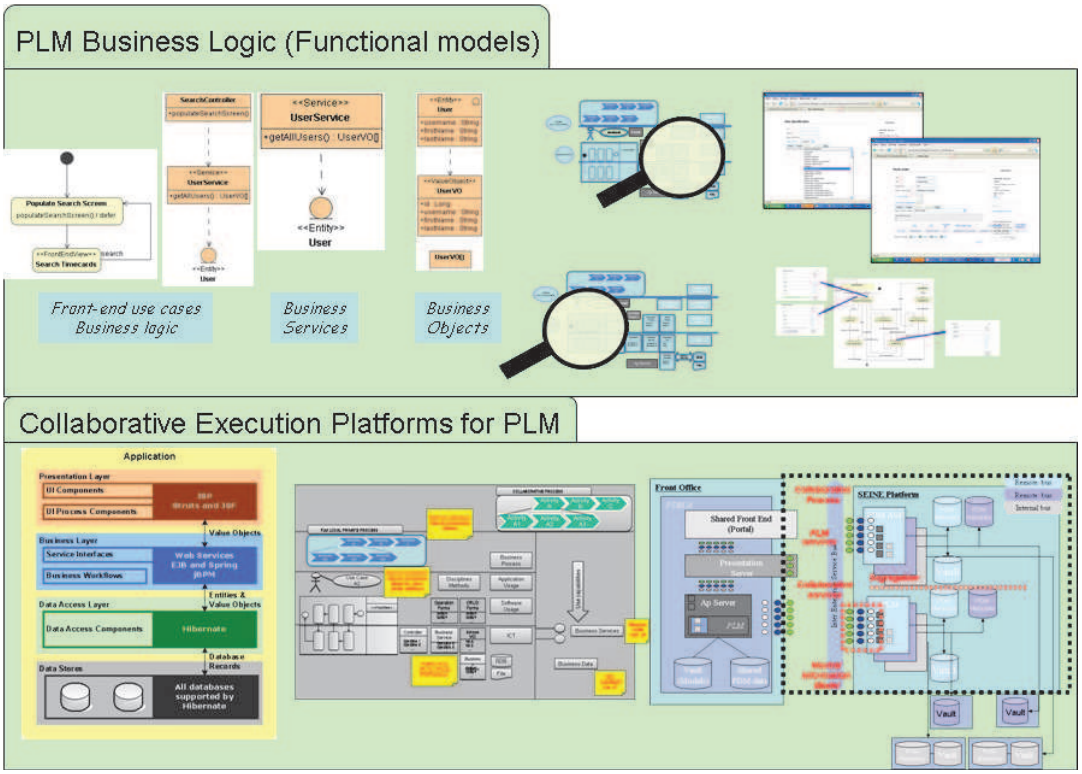


Figure 88: Model Driven engineering approach

The generated models were produced for the three servers in order to support web services generation, with different models for persistence or for service consumption due to some issue with code generation and application server implementation. (Figure 89) shows the Platform independent model automatically generated from the AP214 application protocols. The model is projected two times, a first time as persistent objects, a second time as value object used for computation by web services.

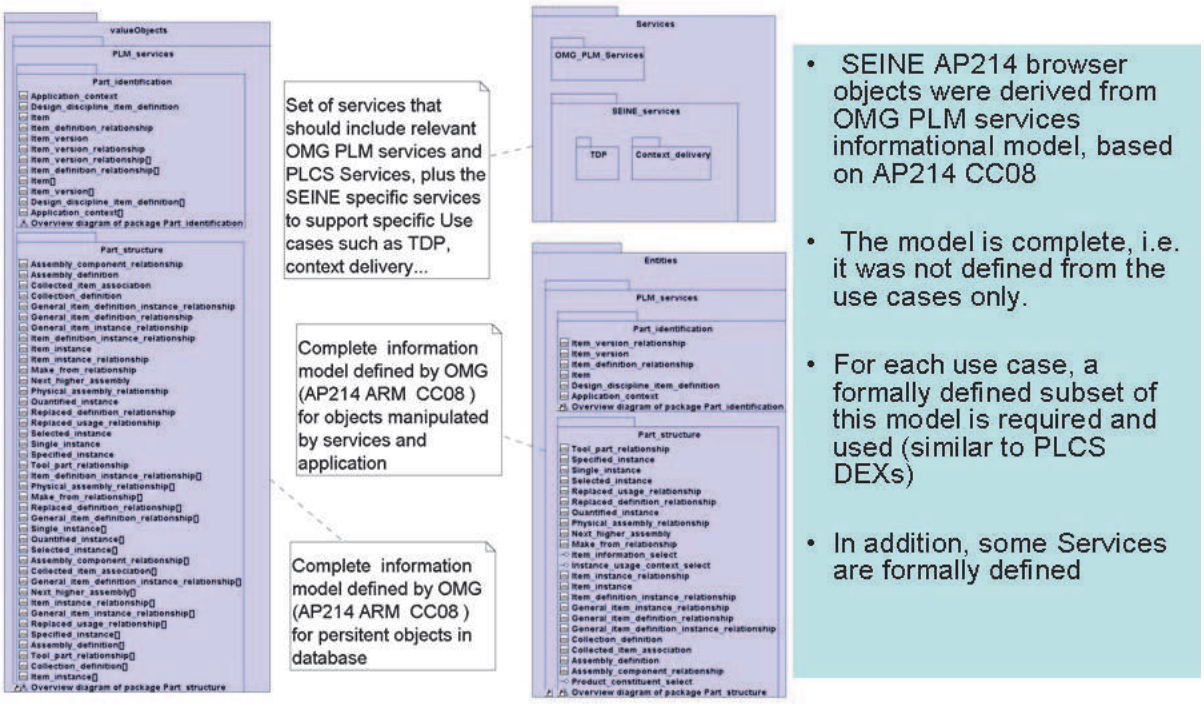


Figure 89: AP214 ARM PIM Model

Each server was then generated without having to code one line, using a code generator and automated deployment of components by application server. Figure 90 provides snapshot of the database and of the forms which were automatically generated. The semantic is preserved between these three representations of a some conceptual model.

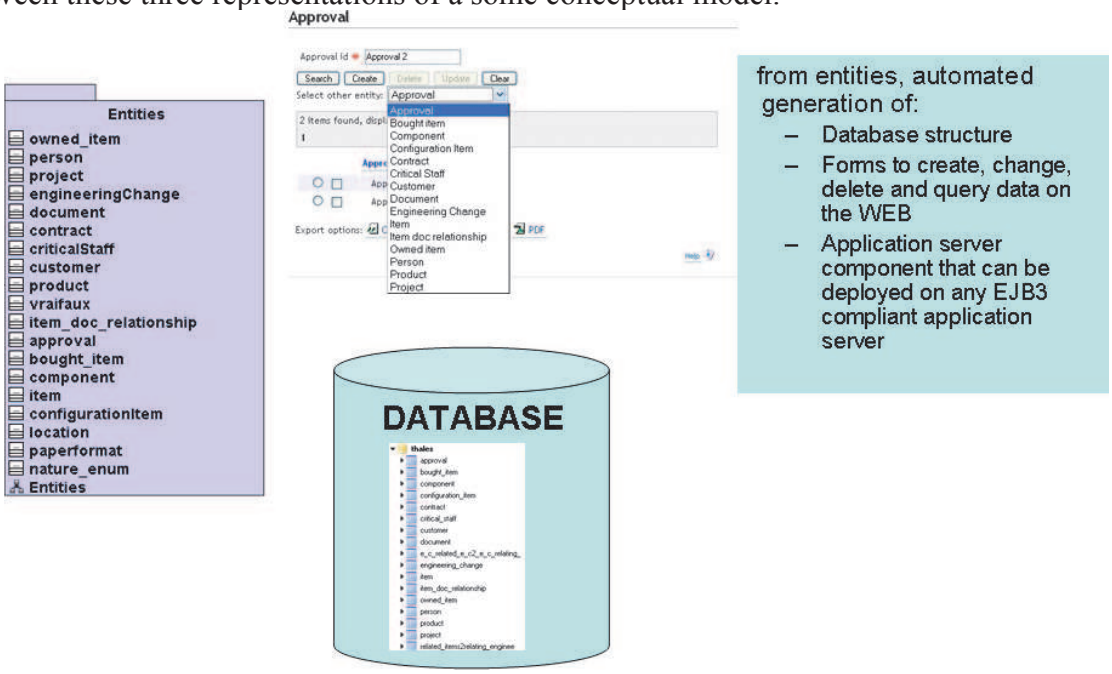


Figure 90: Generated Thales PDM Server

The approach was not only used for generation of database repository, but also for generation of web services and use case based on automatically generated screens from models.

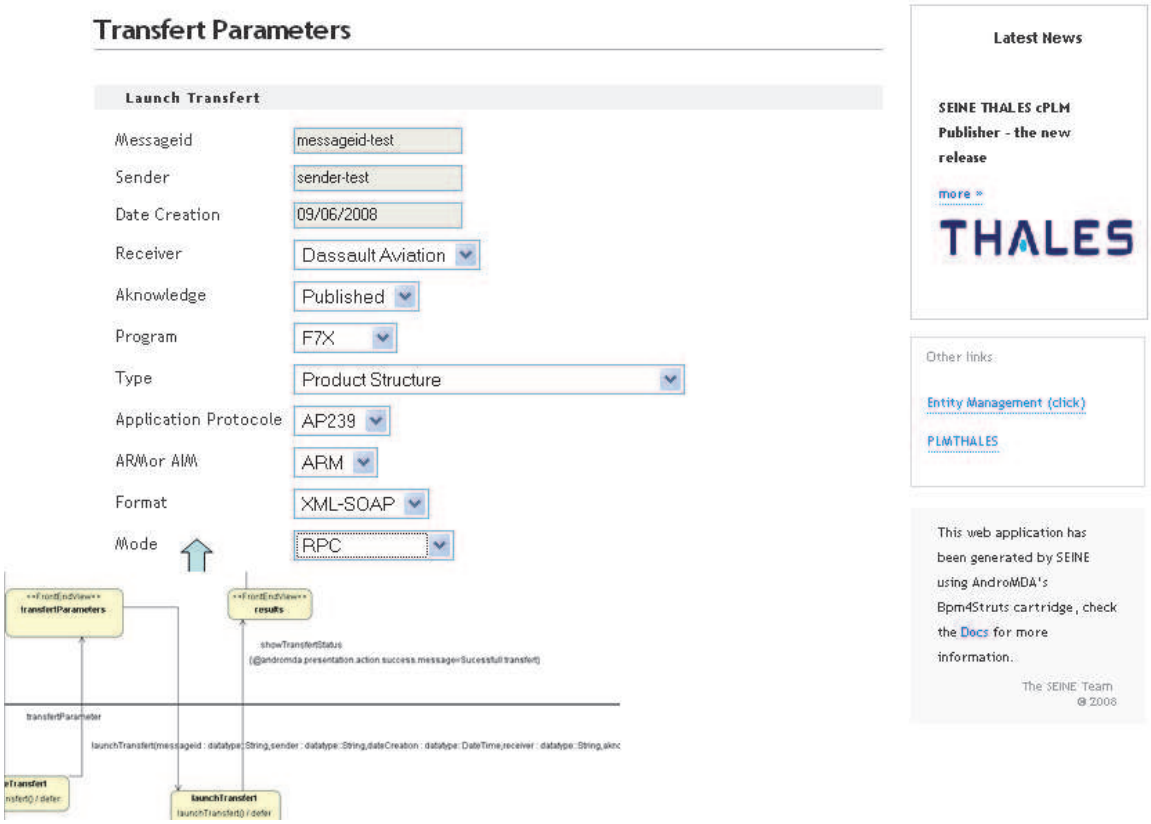


Figure 91: Automated interface generation for use case models

The services were also generated from models, in order to be connected using an open source Enterprise Service Bus based on open standards and with a BPEL engine component allowing to describe transformation process (implementation of the mapping) through XSLT transformation.

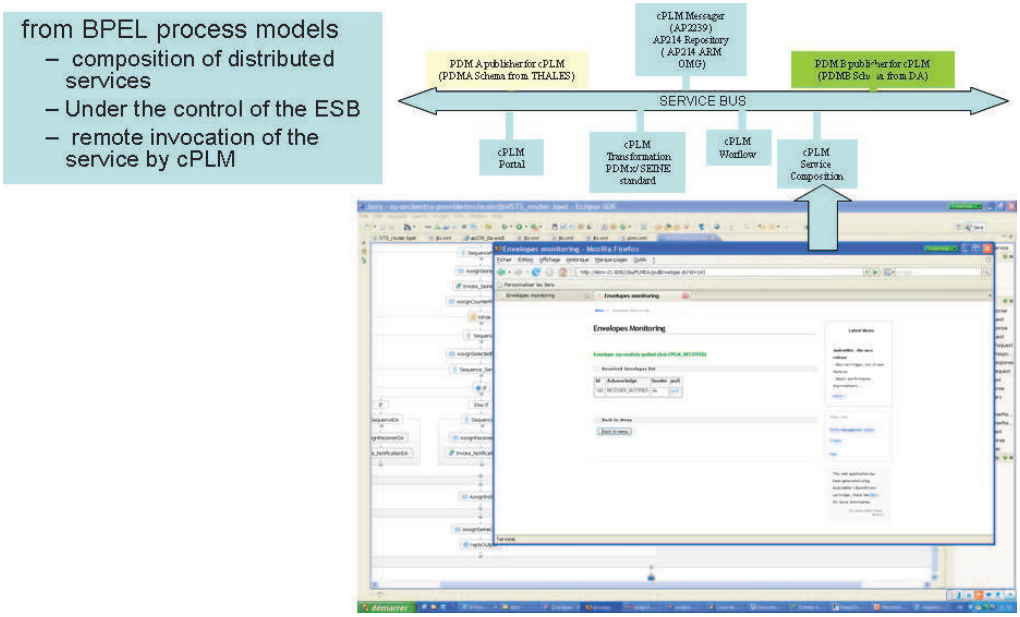


Figure 92: Service composition

Business processes from analyst and formalism using BPMN were aligned with use cases.

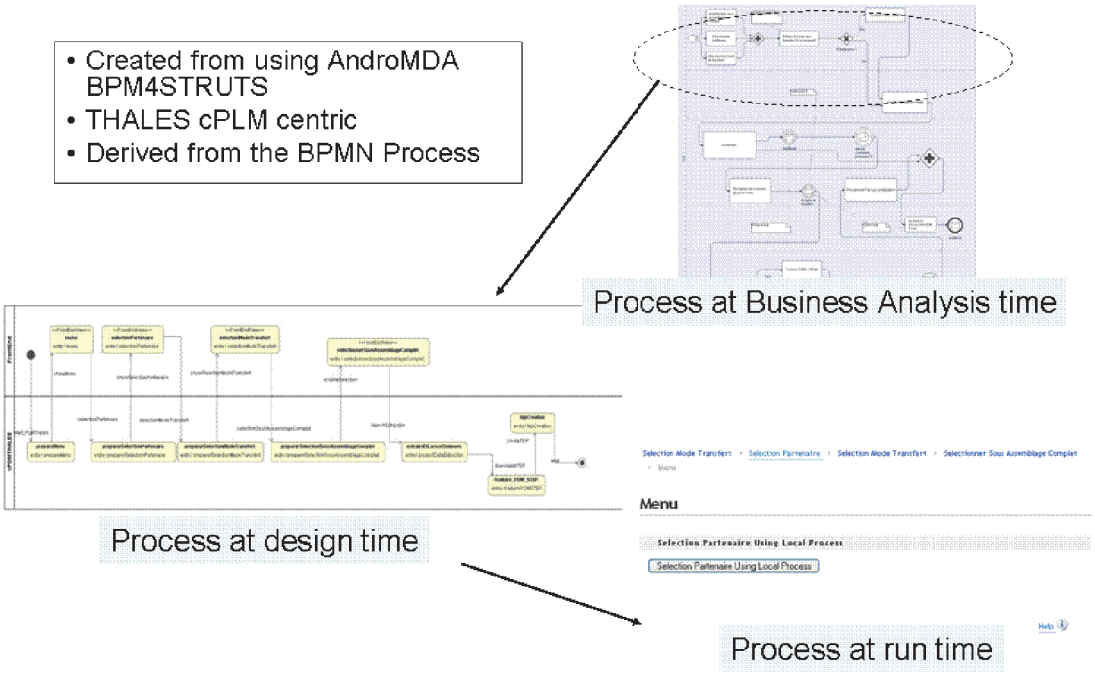
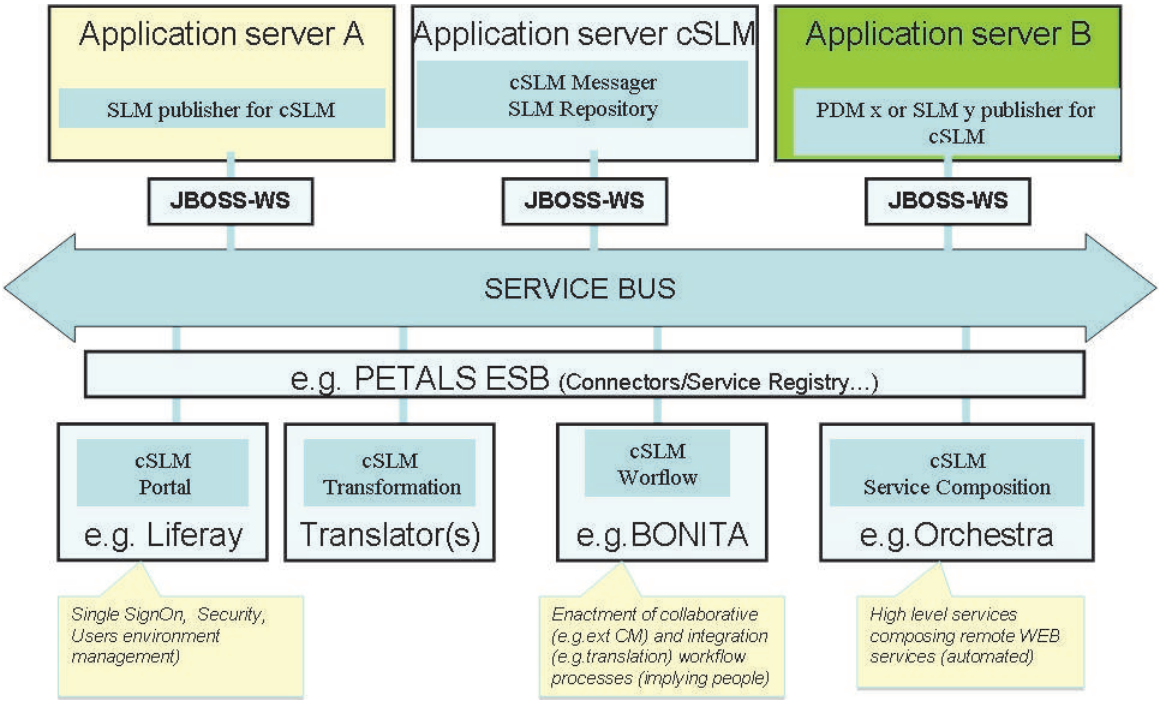


Figure 93: Link between business process and interaction process associated to the use cases

The underlying technical architecture was based on open standards and on corresponding commodities on the web.



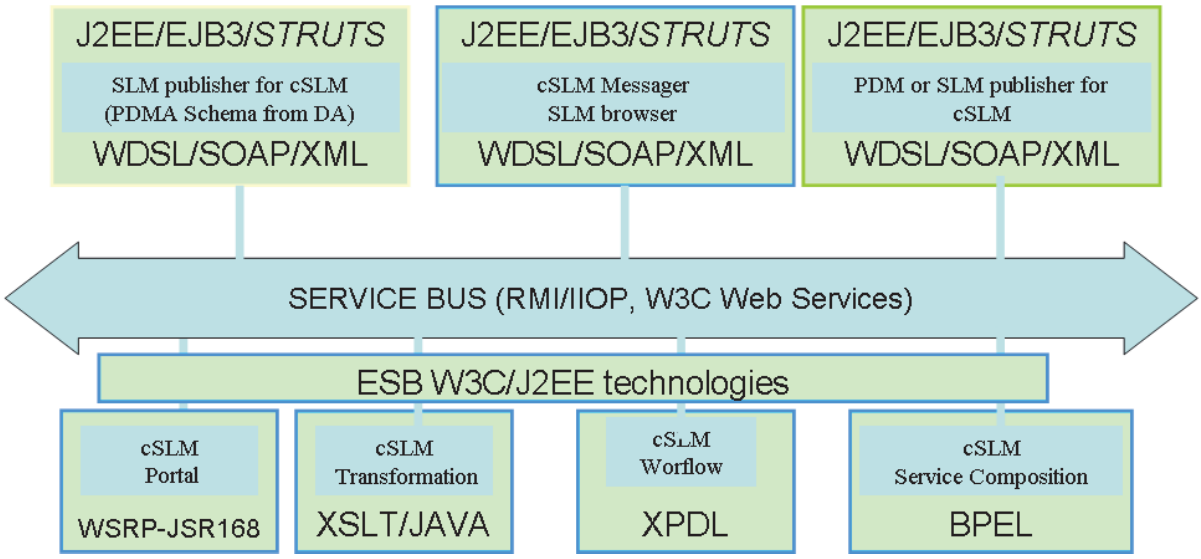


Figure 94: ICT architecture as set of validated grounds

The physical architecture was defined in order to allow remote collaboration.

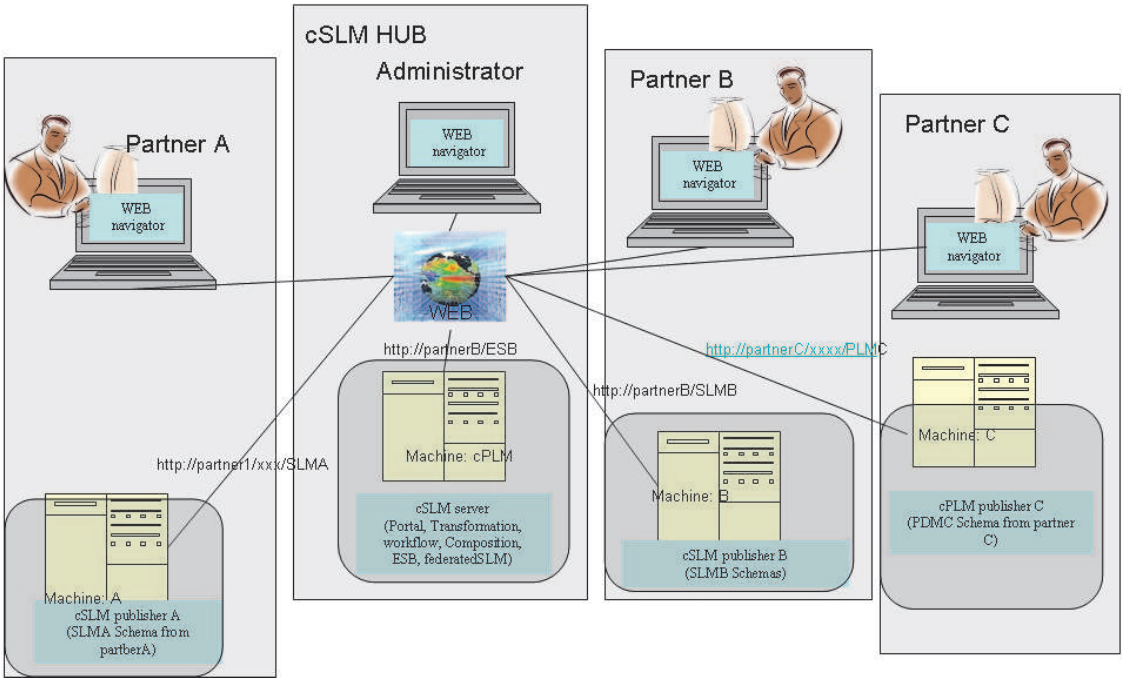


Figure 95: Distributed physical architecture

12.4 - Complementary SEINE development

In order to facilitate visualization of product structure by end user, a component was developed in order to provide a tree representation, on the basis of existing java scripts.

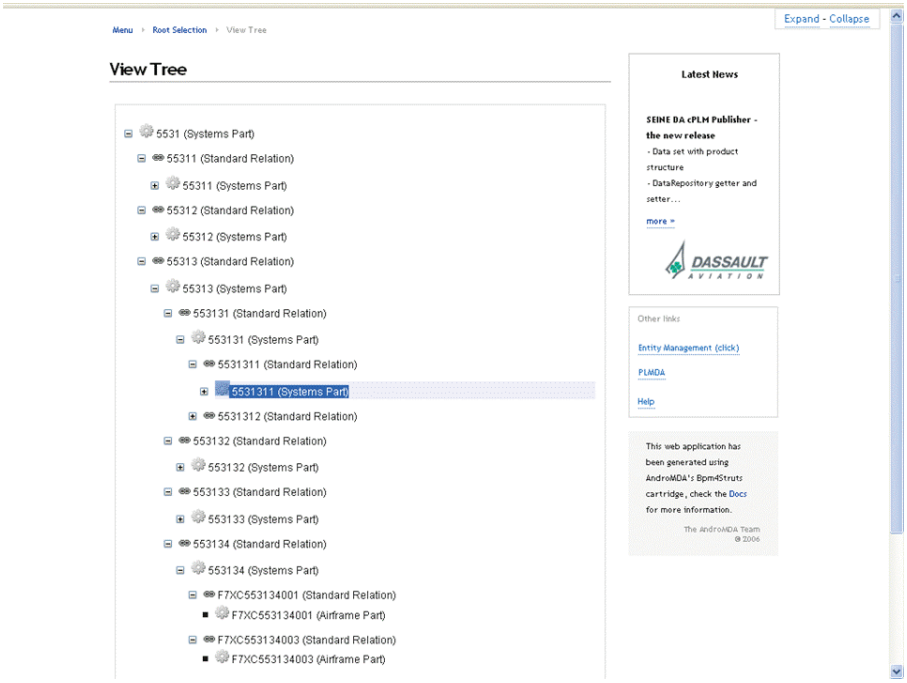


Figure 96: Representation of a Product Structure as a tree

It allowed thinking about graphical representation of complex product structure as formalized within an application protocol.

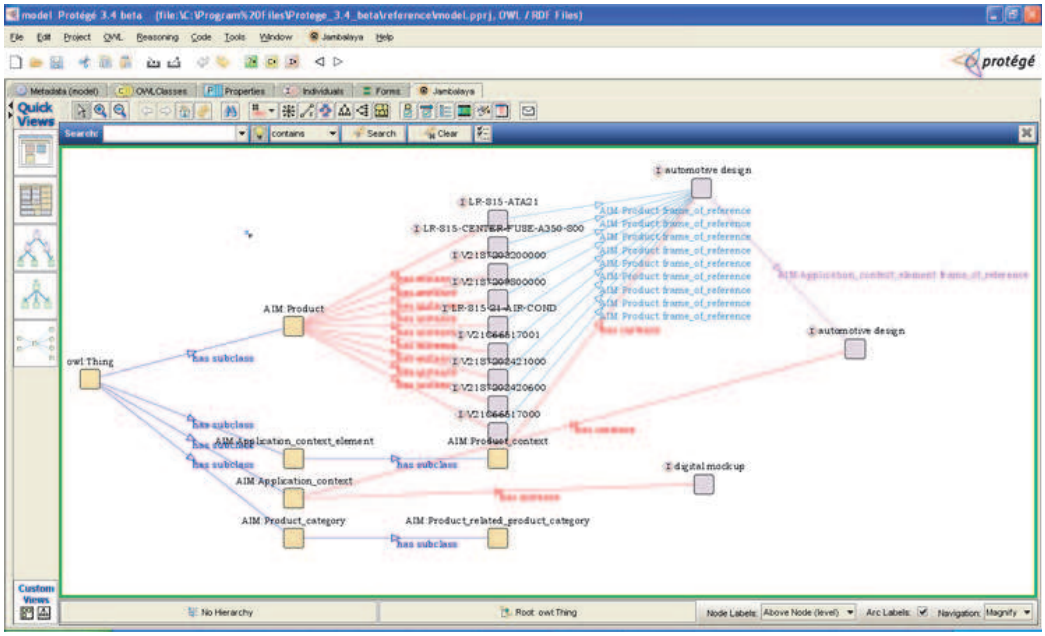


Figure 97: further usage of Protégé ground for graphical representation and reasoning

It allows identifying some semantic mismatch concerning “relationship” concept and their representation as a graph dedicated to an end-user. It also allowed identifying limitation of

Descriptive Logic to represent systems, in particular within a Product Lifecycle context. In particular, a property is defined as a relation between a domain and a range, and is characterized by the graph of related individuals. As a thing can't be at the same time a class, a property or an individual, it means that we are only able to state that two individual are or not related. Considering composition or aggregation relationships which are fundamental for systemic approach, and knowing that a design is not static all along the development process, it leads to important difficulties when willing to identify changes of decomposition or aggregation when data are exchanged between several tools. It led to re-think mapping of application protocol in OWL (Web Ontology Language) a non automated way, as relationship between entities are often represent in STEP as entities, and to envisage further usage of Protégé ground for graphical representation and reasoning (Figure 97).

As a consequence, application protocols were remodeled in OWL by hand, in order to produce new models supporting emerging generation of reasoning engines coming with the version 2 of OWL, with as perspective mapping realization using a reasoning engine, but also enrichment by annotation in order to deal with limitation of descriptive logic.

Finally, by highlighting the limitation of descriptive logic for system engineering, it allows to better understand interest of language such as UML (Unified Modeling Language) and SysML, and to consider an other way their integration within a extended hypermodel, being from modeling point of view or from graphical representation point of view.

12.5 -Lessons learnt from SEINE

SEINE provides the opportunity to evaluate interest of the developed approach in an industrial context. It allows extending the set of components of the execution platform, in particular through usage of Enterprise Service Bus based on JBI (Java Business Integration) standard. Through realization of a complex exchange scenario, it allows providing robust case allowing better assessment of used technologies, and their limitation (e.g. OWL for product structure decomposition).It also allows investigating model generation for services and for human machine interaction processes.
It pointed out issues related to presentation layer and representation for human, in particular for graphical representation. New perspective for extended hypermodel is to consider systematically the language used for presentation layers, and to formalize mappings avoiding semantic mismatch between models and graphical representation.
With emerging second versions of Web Ontology Language and Unified Modeling Language, and due to numerous difficulties related to poor mappings and associated implementations when dealing with complex types (e.g. date and time), a new strategy is under study consisting in developing demonstrators using open source libraries of reference. The goal is to produce more robust implementations and to point out issues with current standardized specifications. Figure 98 illustrate libraries to use according such or such ground, based on configured sets of versioned standards.
Finally, the idea of libraries of formal collaboration processes is also emerging, through the necessity in SEINE to build exchange scenario on precise processes. The maturity of process formalization language and available open source implementation on the web will be studied in future projects.

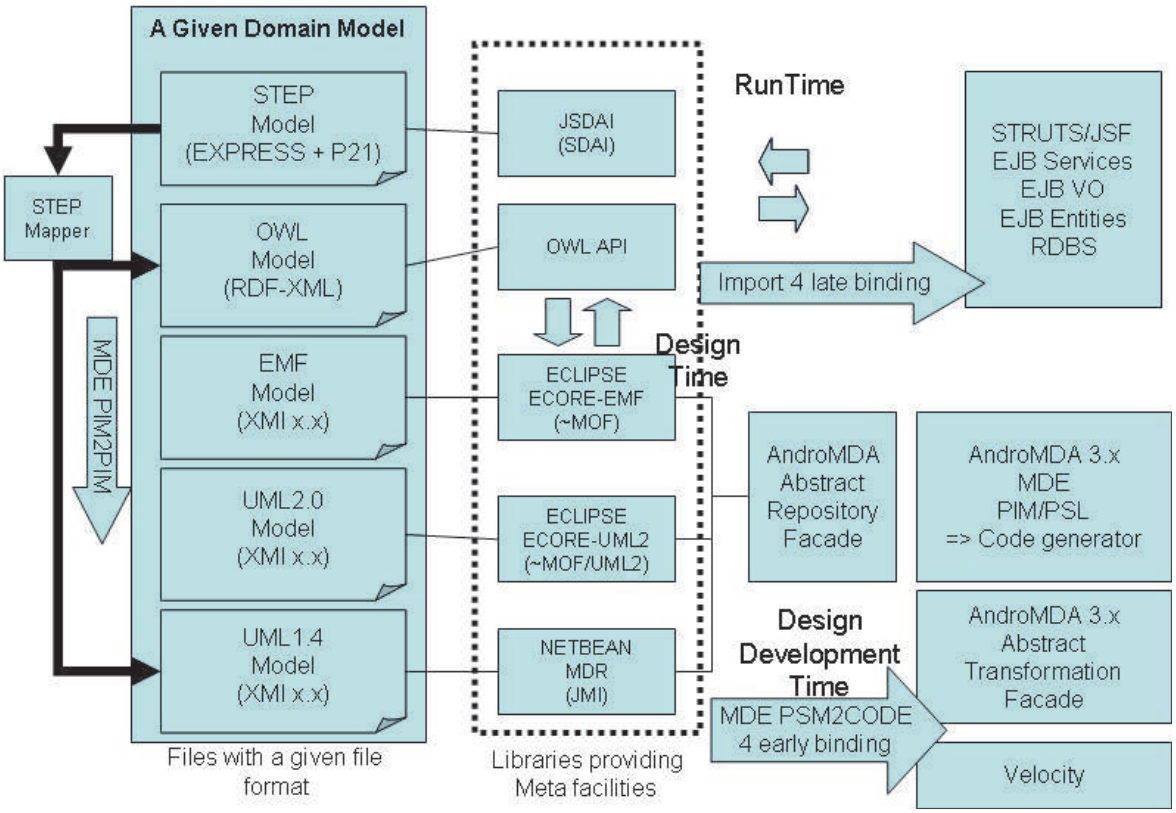


Figure 98: Map of open source libraries for configured set of standards and grounds

Global Conclusion

1-The innovative approach defined within my thesis

Within my thesis, I propose a holistic approach aiming to establish interoperability framework for collaboration space dedicated to a given eBusiness multi-disciplinary community, federating efficiently heterogeneous legacy solutions and emerging paradigms and technologies.
This approach is motivated by the industrial trends, the emerging interoperability needs for networked organizations and by the drawbacks of legacy solutions and approaches. Innovation is not measured only according to analysis of the state of the art, but also according to the state of the practice. The lessons learnt from research and industrial projects I contributed to or I analyzed during the fifteen last years help me to propose a theoretical approach based on a set of concrete experimentations. More than defining what interoperability is, the approach is more focusing on why interoperability is not obtained, and on who the interoperability actors and stakeholders are. For interoperability of enterprise applications within networked organization, communities are definitively important actors and stakeholders, and it is expected that they can reach the accurate level of maturity in order achieving expected interoperability.
Using models of reference allowing describing interoperability levels of the different systems implied, I characterized the ideal collaborative system, allowing achieving seamless exchange of information between distributed information systems and collaborative organizations, by mean of semantic integration, loosely coupled technologies and asynchronous communications. In order to achieve adaptive and continued interoperability, interoperability should be programmatic, constructive and operational, involving evolutionary decision, information and physical systems. Finally matured selected architectures of reference and domain standards should be used and governed.
Having clearly established industrial context and characterized expected interoperability between the different systems constituting a collaboration space, the proposed approach consist in iterative assessment of several combinations of open standards and technologies which enhance achieving federation of application at a reasonable cost. Model driven engineering allowing to project business logic on an execution service oriented platform is promoted, taking advantage of more advanced features coming from enterprise modeling, model driven software engineering, ontology and service oriented technologies. But what makes the originality of the approach is the consideration that several kinds of standards and associated frameworks are to be aggregated a posteriori, considering them as components on the shelves used in order to build a federated composite framework dedicated to support growing interoperability maturity of the considered eBusiness community.
Willing to develop such an approach, I identified numerous issues, in particular semantic preservation and data lost between the different artifacts of a model driven approach and between the different applicative components which have to be interconnected in order to support digital collaboration within the virtual enterprise. It consequently developed a new concept, the extended hypermodels for interoperability, closely integrated in the proposed approach, as a response to the identified issues.
I described the numerous pilot demonstrators, components, language mappings and technology assessments I produced through different research programs and industrial projects during my thesis, as technology and tools assessment for such or such a standardized ground. It allowed me defining a set of principles for interoperability and a set of

interoperability issues and brakes which constitute key item of the methodology I'm proposing, based on systematic capture of "know how" and experience in order to contribute to standardization policies and interoperability framework constitution for a given community.
Finally, the approach is validated within the particular context of applications supporting Product Lifecycle management within the Aerospace and Defense industry, which is quite complex and provides an accurate environment when willing to validate robustness and accuracy of the proposed innovative approach.

2- Discussion about the encountered difficulties and the choices

Addressing interoperability of technical enterprise applications within networked organization is quite complex. Complexity comes from the fact it is required to consider several kinds of systems, different processes, different domains of knowledge, different disciplines, at any level of granularity. Doing so, an important number of actors and stakeholders exist, having each a partial view of the whole system, different interests, different objectives and different backgrounds. In addition, numerous components on the shelves are today available, which have to be considered to constitute the interoperating systems, with growing importance of configuration management and of the management of their lifecycle.
As a consequence, it was quit difficult to delimit the perimeter and the content of what should be or not within the report, when willing to adopt a systemic and global view. A sufficient level of details should be provided, taking into account that numerous readers with very different backgrounds should be able to understand the role they are playing for interoperability of enterprise applications, and how the proposed approach should help to reach new emerging expectations to achieve pragmatic and continuous interoperability with a reasonable cost.
Other difficulty comes from the fact that the way to specify and to define interoperability will highly constrain the level of expectation and the focus. Or numerous communities address interoperability considering different systems and with different perspective, and without an overall picture allowing to clearly use their framework together. It includes researchers, and as a consequence, it was very difficult to write down the state of the art considering only research work. A very deep theoretical but also practical knowledge of existing solutions and standards is required to be able to work in a proper direction which will allow to propose accurate and useful innovation, which will not go against what is already existing, creating technological silos and considering not the need to consider legacy and its evolution with insurance of a continuous interoperability.
As a result, I had important difficulties to organize my thesis a usual way. My work is illustrating and applying the principles I propose in order to address interoperability. Definition of the context, definition of enterprise application interoperability, list of issues, list of principles and prototypes are part of the innovative approach I defined, associated with continuous iterations which will allow to enrich them and to make them evolve in order to ensure continuous adaptation to an always evolving and changing context.
I consequently should have organized my final report a lot of different ways, but always with the difficulty to define a linear way how to deal with continuously evolving interrelated systems of systems. I should have preferred producing mindmaps, with numerous entry points and without imposing a linear way to discover the subject, in order to give an easier perception of the complexity. I finally produced numerous complex figures, without detailed and analytic explanation, in order to make the readers "feel" the problems and the solutions, by mean of visualization. Importance of advanced visualization techniques to deal with complexity influenced my proposal for extending hypermodels in order to deal with

interoperability. Using mathematical theory related to hyper-graphs comes latter, in order to provide a mathematical and analytical foundation to extended hypermodels. I just initialized it, and I think a lot more is to be done. But strong dependence with selected set of standards to consider within a community, and the fact that numerous tools are not implementing standards with the accurate quality, made it difficult to validate, to enrich and to consolidate the concept of extended hypermodels for interoperability. Nevertheless, the initiated work allowed to point out numerous drawbacks with current standardized specification and to identify concrete proposal to improve their quality and usability for emerging interoperability needs.

3- The proposed approach and how to measure its added value and usability

Within my introduction, I detailed the numerous research studies and projects I contributed to during the ten last years, not in order to describe all the work I've done, but in order to put in perspectives how I formalized the problematic, why I fixed some constraints as principles, in particular relying on standards, and finally in order to give an idea of the added value and usability of the approach I propose.

The SAVE project allowed to produce a demonstration dealing with two interconnected legacy applications, being able to exchange and share fourteen objects, and without any idea of the usability within an industrial context. It required three years and three millions euros. The study I made latter demonstrated that is was not an industrial solution, that it was neither reusable nor extensible. So the investment here was fully lost for the research purpose, making it difficult to improve and build on top of obtain results and demonstrators.

The SEINE project allowed producing a prototype with generation of three open source applications from scratch, dealing with one hundred objects, including those of SAVE. It required one month and thirty kilo Euros. Using commodities on the web implementing open standards, they are highly reusable, extensible, and all the develop components and architectural principle make it possible to reuse results not only on the existing execution platforms, but on any other platform implementing the same standards, including Commercial Off-The-Shelves. It also makes it possible to assess robustness of proposed solutions, and to identify and address complex issues which are usually never addressed because never identified. Taking the example of federated product data management systems, it is a purpose which is very difficult to study, as it implies several competitors to work together in order to provide open and interchangeable solutions. In addition, using such commercial software product for prototyping is very expensive due to their complexity, the licensing issues, the cost of wrappings and the architectural issues. Considering emerging governance initiatives, the proposed approach is highly valuable by allowing supporting faster specification and qualification of interoperable solutions. Fast prototyping allows focusing on business cases and semantic issues, and not on technological issues related to implementation. It also allows faster and computer aided assessment of huge standards a more systematic and industrial way. A STEP application protocols contains thousands and thousands of pages, which are very difficult to read and to validate for most of the concerned actors. In addition, the proposed approach allows working on configured combination of standards, making easier to define and to assess "compatibility" of standards.

The proposed approach allows generating very fast robust platforms implementing the standards in order to be used as reference platform for computer aided specification and tests generation.

Finally, creation of hyperedges required to consider differently mappings between standards. Even without finalized formalization, it allowed to identify some drawbacks concerning existing standardized specifications and to propose changes in order to improve their quality and their usability within the context of model driven system engineering.

4- Perspectives

Practical perspectives

From a practical viewpoint, assessment of existing grounds and their accurate combination should be continued in order to provide robust environments allowing going further on assessment of set of configured standards. It is true for the different environments, being modeling, development or execution. Opportunity exists to use these environments for research purpose, standards governance purpose and computer aided specifications.
I also identified important issues with quality of existing implementations, which are not always covering fully implementation of standards. It should be accurate to change the strategy with providing dedicated development allowing producing more robust and compliant implementation of reference. It will be undertaken on the basis of using set of dedicated open source APIS of reference, and providing complementary services dealing with services dedicated to deal with extended hypermodels for interoperability.
From performed experimentation, it is planned to produce libraries of models of reference using different formalisms, which will be used for demonstration purpose and as input for requesting evolution of standards for better interoperability.
Important technical investigations are still to be done when willing to deal with complex data types such as date, sets, composite objects, etc. Work was done on reduced set of simple data types, for example those used for reasoning. But there is still a ground to extend set of data types that can be easily used for an extended landscape.
In addition, the way to parameter the different available tools for mapping should be studied in order to complete available tools on the shelves. For example, application servers are proposing default ways to deal with service remote publication, which is not suited for complex models. Most of the components can't be used as they are, and the proposed parameterization should be published and shared.
For future demonstrators and pilots, other functional components and standards will be included to compose the different platforms. It is already planned to include emerging enterprise portals with social networking and semantic capabilities, large triple stores and remote reasoning engines on execution platforms. Presentation components will also be included such as emerging AJAX frameworks with advanced visualization widgets.
It is also planned to use work performed on application protocols to contribute to the future ISO TC184 SC4 architecture, by highlighting how the proposed mappings, done in order to obtain robust extended hypermodel, can contribute to produce more useable standards allowing reducing cost of continuous interoperability despite continuous change and evolution of standards and technologies.

Theoretical perspectives

Development of extended hypermodel for interoperability was just initiated during the thesis, but should required further activities for more robust theoretical model. In addition, it can be applied to other domains than interoperability of technical enterprise within networked organizations. In particular, it can be applied to multi-disciplines design of a product implying different physics, at different scales. The focus here will be to provide a product model federating different representations of the product which are not always connected. It will then be possible to take advantage of advanced computer aided design and reasoning capabilities to assist the engineers for models validation and management.
Finally, the proposed approach can be extended to authoring applications supporting computed aided design, which are out of the scope of the thesis. The idea is to extend the set of standards with standardized language allowing to model methodologies and to integrate them on the landscape of covered standards. I identified SPEM and ISO15928 (System Engineering Process) as candidates for efficient support of computer aided system engineering activities.

Standardization perspectives

Comparing the languages used for standardization highlight that each language has specific qualities, related to the initial intention of standardization communities, which could be transferred to others. Numerous questions arose on the formal language to use when producing specification. The thesis highlight that to choose one is may be not appropriated, but we should combine usage of several languages. This is usually not done as we always need a model of reference, from which the others are deriving. But what if we can demonstrate that the different representations are equivalent and that we can switch from one representation to the other without information loss and with semantic preservation?
Doing so, it should be feasible to use several standards together, withdrawing the technological and paradigm silos, and making it easier for a given community to govern more easily the set of relevant standards they have elected in order to obtain continuous interoperability within evolutionary environments.

Bibliography

Archimate: “Archimate 1.0”, Open Group, http://www.opengroup.org/archimate/1.0

ARCHIMATE: “An integrated architectural approach for the description and visualization of different business domains”, http://www.archimate.org/en.about_archimate/archimate_project.html

Arlab F. and Vand Der Torre (2005), ”Interactive visualization of dynamic models” (2005), http://ftp.cwi.nl/CWIreports/SEN/SEN-E0516.pdf

ATHENA: “Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Application”, form 01/02/2004 to 31/03/2007, http://interop-vlab.eu/ei_public_deliverables/athena-deliverables/list-of-public-deliverables/list-of-deliverables-submitted-during-the-athena-project, http://www.modelbased.net/mdi/framework.html

Benguria G., Berre J., Elvesaeter B., Fischer K.,Friese T., Larrucea X., Muler J. and Neple T. (2008), web site at http://www.modelbased.net/mdi/index.html

Berre J., Elvesæter B. , Figay N. , Guglielmina C., Johnsen K., Karlsen D., Knothe T. and Lippe S. (2006),” The ATHENA Interoperability Framework”, IESA 2006

Bezivin J. (2002), “Aspect-Oriented Modelling: Oxymoron or Pleonasm”, AOPDCS’2002, Vienna, http://atlanmode.emn.fr/www/papers/oxymoron_pleonasm.pdf

BPEL 1.1: “Business Process Execution Language fror Web Services version 1.1”, OASIS, May 2003, http://xml.coverpages.org/BPELv11-May052003Final.pdf

BPEL 2.0: “Web Services Business Process Execution Language Version 2.0”, OASIS, April 2007, http://docs.oasis-open.org/wsbpel/2.0/primer/wsbpel-v2.0-Primer.pdf

BPMN 2.0: “Business Process Modeling Notation (BPMN) 1.2”, OMG, January 2009, http://www.omg.org/spec/BPMN/1.2/

Calson D. (2006), « Semantic Models for XML Schema with UML Tooling”, http://xmlmodeling.com/papers/SemanticModelsForXSD , SWESE 2006

CCM: “CORBA Component Model, v4.0”, OMG, April 2006, http://www.omg.org/cgi-bin/doc?formal/06-04-01.pdf

Clark, T. and R. Jones (1999) “Organisational Interoperability Maturity Model for C2”

COMET: “Component and Model-based Development Methodology”, http://modelbased.net/comet

CORBA: “Common Object Request Broker Architecture (CORBA/IIOP) Version 3.1”, OMG, January 2008, http://www.omg.org/spec/CORBA/3.1/

Formal/2008-01-04: “Part 1 – Interfaces”,
http://www.omg.org/spec/CORBA/3.1/Interfaces/PDF
Formal/2008-01-06: “Part 2 – Interoperability”,
http://www.omg.org/spec/CORBA/3.1/Interoperability/PDF
Formal/2008-01-08: “Part 3 – Components”,
http://www.omg.org/spec/CORBA/3.1/Components/PDF
Ptc/2003-01-10:, ptc/2003-01-06, ptc/2001-11-01: interfaces in IDL

Daclin N., Chen D. and Vallespir B. (2006), « Modèle de maturité pour l’interopérabilité des entreprises » at proceeding of « 6e journée STP, Valenciennes, 16-17 Novembre 2006 »

Dartigues C., Ghodous P., Gruninger M, Sriram R. (2007), “CAD/CAPP Integration using Feature Ontology”, Concurrent Engineering: Research and Applications 15(2):237-249, SAGE publications, ISSN 1063-293X, 2007.

DBE: “Digital Business Ecosystem”, http://www.digital-ecosystems.org

DI: “UML Diagram Interchange”, April 2006, http://www.omg.org/spec/UMLDI/1.0

EJB3: “JSR-220 Enterprise JavaBeans 3.0 Final Release”, Java Community, May 2006, http://jcp.org/en/jsr/detail?id=220

Feng S.(2003), “Intelligent Agents and Process Specification Language in Predictive Process Engineering”, at Networking in Metal Industry Business and Software Aspects, Raahe and Oulu, Finland, 25-28 February 2003, available at http://www.pbol.org/projects/stellnet/files/seminaari/nist-scf.ppt

Ferreira Da Silva C. (2007), “Découverte de correspondances sémantiques entre ressources hétérogènes dans un environnement coopératif.” Université Claude Bernard Lyon I, 2007.

Ferronato P. (2004): “Introduction to the Digital Business Ecosystem Project”, MDA Technical Forum , Tokyo, October 2004,
http://www.solut.net/pdfs/20041020TokyoDBEArchitecture_v1%5B1%5D.2.pdf

Figay N. (2006a), “Technical Enterprise Applications interoperability to support collaboration within the Virtual Enterprise all along lifecycle of the product”, I’ESA2006 Doctoral Symposium

Figay N. (2006b), « Collaborative Product Development: EADS Pilot Based on ATHENA results », 2006, within “Leading the web in concurrent engineering – Next Generation Concurrent Engineering”, IOS Press, ISBN I-58603-651-3

Ford C., Colombi J., Graham S. and Jacques D. (2008) “A Survey on Interoperability Measurement”, 12th ICCRTS “Adapting C2 to the 21st Century”

Ghafour A., Ghodous P., Shariat B., Perna E. (2007), “A Common Design-Features Ontology for Product Data Semantics Interoperability”, during the 2007 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2007)., Silicon Valley - United States of America. 2007.

Ghafour A. (2009), « Interopérabilité Sémantique des connaissances des modèles de produits à base de features », Computer Science thesis, Université Claude Bernard Lyon 1, 9 juillet 2009.

Ghodous P. (2002), « Modèles et Architectures pour l'Ingénierie Coopérative » Habilitation to drive research, Université Claude Bernard Lyon 1, 13 décembre 2002.

Gobert G. (2005), "From Computer Based Manipulatives to Hypermodels", available at http://isites.havard.edu/fs/docs/icb.topic603902.files/Bilogica Chapter v10.pdf

Hodgson R., Knublauch H. and Keller P.; "Ontology-Based XML Schemas for Interoperability between Systems and Tools", http://2006.xmlconference.org/programme/presentations/119.html, XML 2006

Hofmann M. (2003), "Essential preconditions for coupling model-based information systems," NATO M&S Group (NMSG) Conference on C3I M&S Interoperability, RTO-MP-123, Antalia, Turkey

Hoffmann P. (2008), "Context-based Semantic Similarity across Ontologies", Computer Science thesis, Université Claude Bernard Lyon 1, 16 Décembre 2008.

Hoffmann P., Feng S., Ameta G., Ghodous P., Qiao L. (2008), "Towards a Multi-View Semantic Model for Product Feature Description ", 15th ISPE International Conference on Concurrent Engineering: Research and Applications (CE 2008)), , Belfast, Irlande. 9p. Springer . ISBN 978-1-84800-971-4. 2008.

Horwittz H. (1995), "Linking Models to Data: Hypermodels for Science Education ». The High School Journal, 79(2), 148 - 156., 1995

IDEAS: Thematic Network "Interoperability Development for Enterprise Applications and Software"- Roadmaps, from September 2002 to June 2003, Delivables on http://interop-vlab.eu/ei_public_deliverables/ideas-deliverables/list-of-ideas-deliverables

INCOSE: International Council on Systems Engineering, web site at http://www.incose.org

ISO/IEC 15288:2002 "System Engineering – System life cycle processes", was produced by INCOSE

Jardim-Goncalvez R., Figay N. and Steiger-Garcao A. (2006), "Enabling Interoperability of STEP Application Protocols at meta-data and knowledge level", International Journal of Technology Management, Volume 36, Number 4/2006, p 402-401

JSR 168: "Java Specification Requests: JSR 168 : Portlets Specification ", Java Community, http://www.jcp.org/en/jsr.detail?id=168

JSR 286: "Java Specification Requests: JSR 286: Portlet Specification 2.0", Java Community, http://www.jcp.org/en/jsr.detail?id=286

Klein, L. (2008), « Semantic STEP : Linking ISO 10303 with the semantic Web", 10th NASA-ESA Workshop on Product Data Exchange, Noordwijk, The Netherlands, 2008-03-26, available at http://www.congrex.nl/08c13/presentations%5Cpde2008-07-Klein.ppt

Krause F.-L., Kaufmann U. (2007), "Meta-Modelling for Interoperability in Product Design", CIRP annals - Manufacturing Technology, Column 56, Issue 1, 2007, Pages 159-162

Kuhn O., Lima Dutra M., Ghodous P., Dusch T., Collet P. (2008) "Collaborative Architecture Based on Web-Services", 15th ISPE Conference on Concurrent Engineering, Springer-Verlag London ed. Belfast, Irlande du Nord, Août 2008. pp. 53-62. Advanced Concurrent Engineering . Springer London . ISBN 978-1-84800-971-4. ISSN 1865-5440. 2008.

Leal D. and Bohms M. (2007), "Product and process modelling standards on the Web", The 6th International Semantic Web Conference and the 2nd Asian Semantic Web Conference, 2007, Busan, Korea, November 2007, available at http://iswc2007.semanticweb.org/papers/Paper384-S-TEN_2007_poster_summary_v1.pdf

Lillehagen F., Krogstie J. And Grenager Solheim H. (2005): "From enterprise modelling to enterprise visual scenes", International Journal of Internet and Enterprise Management, Issue: Voume 1, number 2/2005, p 139-154

Malone P. (2007), "DE services in Ecosystem Oriented Architectures", "Digital Business Ecosystems", Office for Official Publications of the European Communities, 2007- ISBN 92-79-01817-5, Section 3, p119 – available at http://www.digital-ecosystems.org/book/Section3.pdf

MDA: "MDA Guide Version 1.0.1", OMG, omg/2003-06-01, June 2003, http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf

MOF 2.0: "Meta Object Facility version 2.0", OMG, January 2006, http://www.omg.org/spec/MOF/2.0/
Formal/2006-01-01: MOF Core Specification, http://www.omg.org/spec/MOF/2.0/PDF
Ad/2003-04-08: collection of Rose models and catalog files used as example, http://www.omg.org/spec/MOF/model/EMOF.mdl, http://www.omg.org/spec/MOF/model/MOF2.mdl,

Mori G., Paterno F. and Santoro C. (2003) "Tool Support for Designing Nomadic Applications", http://www.iuiconf.org/03pdf/2003-001-0048.pdf

Morris E., Levine L., Meyers C., Place P. and Plakosh D. (2004) "Interoperability (SOSI): Final Report, April 2004"

Morris E., Levine L., Place P., Plakosh D. and Meyers C. (2004) "System of System Interoperability",http://www.sei.cmu.edu/pub/documents/04.reports/pdf/04tr004.pdf

Muellers S. "Interactive Visualization of Semantic Graphs". .http://www.cg.tuwien.ac.at/research/publications/2009/mueller-2009-ivs/mueller-2009-ivs-paper.pdf , 2009

NEHTA (2007), "Interoperability Framework v2.0", available at http://www.nehta.gov.au/component/docman/doc_details/391-interoperability-framework-v20

Obstr L. (2008) "The Ontology Spectrum & Semantic Models", at National Science Foudation Accounting Interoperability Worskshop, April 2008, available at http://nsfaccoutingontology.wi.is/@api/deki/files/3/=LeoObrst05122008.pdf

Obstr L. (2009) "Ontologies & the Semantic Web for Semantic Interoperability", SICOP909

ODM 1.1: "Ontology Definition Metamodel Version 1.0, OMG, May 2009, http://www.omg.org/spec/ODM/1.0/

Formal/2009-05-01: v1.0, http://www.omg.org/spec/ODM/1.0/PDF
Ptc/2008-09-08: XMI, http://www.omg.org/spec/ODM/20080901
Ptc/2008-09-09: XMI, http://www.omg.org/spec/ODM/200809012
Dtc/2009-06-15 : "Reference Metamodel for the EXPRESS InformationModeling Language Specification" Beta 2 with bar change

OMG PLM Services V2: "Specifications of PLM services V2.0", OMG, http://www.omg.org/docs/mantis/06-12-01.pdf

- "Description and implementations of reference on the ProSTEP IVIP WEB site", http://www.prostep.org/en/projektgruppen/pdm-if/plmservices.htm
- VDA 4965 – "Engineering Change Management (ECM)", "Part 0: Engineering Change Management (ECM)"; "Overview general recommendations for the overall ECM Reference Process Part", http://www.prostep.org/fileadmin/freie_downloads/Empfehlungen-Standards/VDA/VDA_4965-0_Engineering-Change-Management_2.0.zip
- VDA 4965-1 - Engineering Change Request (ECR), "Part 1: Engineering Change Request (ECR)": ECM Reference Sub-Process used to communicate Engineering Change Request (ECR), http://www.prostep.org/fileadmin/freie_downloads/Empfehlungen-Standards/VDA/VDA_4965-1_Engineering-Change-Request_2.0.zip
- VDA 4956 - Recommendation for Product Data Exchange (Part 1), "VDA, PDM data exchange - international recommendation for PDM data exchange ", http://www.prostep.org/fileadmin/freie_downloads/Empfehlungen-Standards/VDA/VDA_4956-1_Product-Data-Management_1.0.pdf
- "PLM Services Implementation Guideline":Implementation Guideline concerning OMG PLM Services 2.0, http://www.prostep.org/fileadmin/freie_downloads/Guidlines-UseCases/ProSTEP-iViP_Implementation-Guideline_PLM-Services_2.0.pdf
- "ECR Implementation Guideline" :Implementation Guideline for "Engineering Change Requests", http://www.prostep.org/nc/de/hilfsseiten/downloadlogin.html?redirect_url=fileadmin%2Fgesicherte_downloads%2FGuidlines-UseCases%2FProSTEP-iViP_Implementation-Guideline_Engineeringe-Change-Request_1.0.pdf
- CC8 Recommended Practices - Specification and Cofiguration / Product Coding / Documents and Transformation Matrices , http://www.prostep.org/fileadmin/freie_downloads/Guidlines-UseCases/ProSTEP-iViP_Implementation-Guideline_STEP-CC08_1.2.pdf
- ISO 10303, STEP, AP214, Conformance Class 8 - Recommended Practices for the usage of STEP AP214 CC8

- Usage Guide PLM Services 2.0 , Implementation Guideline concerning OMG PLM Services 2.0 , http://www.prostep.org/fileadmin/freie_downloads/Guidlines-UseCases/ProSTEP-iViP_Implementation-Guideline_PLM-Services_2.0_01.pdf
- PDTnet Project - Application Scenario 1 "PDM Data Exchange", PDTnet Project, PDM data exchange - Flyer for scenario 1, product data exchange between partners with different PDM systems , http://www.prostep.org/fileadmin/freie_downloads/Guidlines-UseCases/UseCases/ProSTEP-iViP_Use-Case_PDTnet-PDM-Data-Exchange_1.0.pdf
- PDTnet Project - Application Scenario 2 "PDM Web Integration" , PDTnet Project, PDM data exchange - flyer for scenario 2, access to the PDM system of development partners via the Internet based on standardized protocols, synchronous and asynchronous data exchange , http://www.prostep.org/fileadmin/freie_downloads/Guidlines-UseCases/UseCases/ProSTEP-iViP_Use-Case_PDTnet-Web-Integration_1.0.pdf

OpenDevFactory: Sub project of “Usine logicielle”, www.usine-logicielle.org

OWL 1.1: “OWL WEB Ontology Language”, W3C, February 2004, http://www.w3.org/2004/OWL

“OWL Web Ontology Language Overview”, http://www.w3.org/TR/owl-features/
“OWL Web Ontology Language Guide”, http://www.w3.org/TR/owl-guide/
“OWL Web Ontology Language Reference”, http://www.w3.org/TR/owl-ref/
“OWL Web Ontology Language Semantics and Abstract Syntax”, http://www.w3.org/TR/owl-semantics/
“OWL Web Ontology Language XML Presentation Syntax”, http://www.w3.org/TR/owl-xmlsyntax/

OWL 2.2: “OWL 2 Web Ontology Language”, W3C, September 2009 (proposed recommendations) , http://www.w3.org/TR/

“OWL 2 Web Ontology Language Conformance”, http://www.w3.org/TR/2009/PR-owl2-conformance-20090922/
“OWL 2 Web Ontology Language Direct Semantics”, http://www.w3.org/TR/2009/PR-owl2-diect-semantics-20090922/
“OWL 2 Web Ontology Language Maping to RDF Graphs”, http://www.w3.org/TR/2009/PR-owl2-mapping-to-rdf-20090922/
“OWL 2 Web Ontology Language New Features and Rationale”, http://www.w3.org/TR/2009/PR-owl2-new-features-20090922/
“OWL 2 Web Ontology Language Document Overview”, http://www.w3.org/TR/2009/PR-owl2-overview-20090922/
“OWL 2 Web Ontology Language Primer”, http://www.w3.org/TR/2009/PR-owl2-primer-20090922/
“OWL 2 Web Ontology Language Profiles”, http://www.w3.org/TR/2009/PR-owl2-profiles-20090922/
“OWL 2 Web Ontology Language Quick Reference Guide”, http://www.w3.org/TR/2009/PR-owl2-quick-reference-20090922/
“OWL 2 Web Ontology Language RDF Based Semantics”, http://www.w3.org/TR/2009/PR-owl2-rdf-based-semantics-20090922/
“OWL 2 Web Ontology Language Document Overview”, http://www.w3.org/TR/2009/PR-owl2-overview-20090922/

"OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax", http://www.w3.org/TR/2009/PR-owl2-syntax-20090922/
"OWL 2 Web Ontology Language XML Serialization", http://www.w3.org/TR/2009/PR-owl2-xml-serialization-20090922/
"OWL 2 Web Ontology Language Document Overview", http://www.w3.org/TR/2009/PR-owl2-overview-20090922/

Panetto H.(2006), "Metamodels and models for the integration and the interoperability of production enterprise application (in french)", available at http://page-perso.cran.uhp-nancy.fr/Perso_Panetto

Panetto H., Terzi S., Morel G. and Gareeti M. (2007), "A holonic metamodel for product lifecyle management", International Journal of Product Lifecylce Management 2, 3(2007) 253-289

PLCS PLM services v1: "OASIS PLCS PLM Services v1", OASIS, 2006, http://www.oasis-open.org/apps/org/workgroup/plcs/download.php/18941/PLCSws_v11.zip
Online WSDL and XSD files for PLCS PLM Services (last version at http://www.plcs-resources.org/plcs_ws/v1/)

PLCS PLM services v2: "OASIS PLCS PLM Services v2", OASIS, Juin 2008, http://www.oasis-open.org/apps/org/workgroup/plcs/download.php/28692/PLCSPLMservicesv2_2008027.zip
Online WSDL and XSD files for PLCS PLM Services (last version at http://www.plcs-resources.org/plcs_ws/v2/)

POEM: "Process oriented Enterprise Modeling methodology", available on the Process Modeling City at http://bpmnpop.sourcefore.net

Price D., Barkmeyer E. , Denno P., Kaufmann U. (2006), "Enabling OMG Model Driven Architecture for ISO EXPRESS-based implementation", 8th NASA/ESA PDE Workshop April 2006, available at http://www.google.fr/url?sa=t&source=web&ct=res&cd=1&ved=0CAgQFjAA&url=http%3A%2F%2Fwww.modelalchemy.org%2Fphpatm%2Findex.php%3Faction%3Ddownloadfile%26filename%3DMEXICO%2520V1%2520UK.ppt%26directory%3DPresentations%26PHPSESSID%3D1cc002eb24233e772554a69456fbb1fb%26PHPSESSID%3D1cc002eb24233e772554a69456fbb1fb&ei=1sHQStqyPOLOjAeh4on5Aw&usg=AFQjCNH8I8Xq0TuRC6n6ZbtEXprmFuJsnw&sig2=lAhg0NhOeokwxUyROnRt_A

Ray S. (2002), "Interoperability Standards in the Semantic Web", Journal of Computing and Information Science in Engineering, 2:1, 65-69, 2002

Ray S. (2006), "The evaluation of ontologies", Chapter 7 in "Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences", Springer, 2006

RDF: "Resource Description Framework (RDF)", W3C, February 2004, http://www.w3.org/RDF
"RDF/XML Syntax Specification (Revised)", http://www/w3.org/TR/rdf-syntax-grammar

"RDF Vocabulary Description Language 1.0: RDF Schema", http://www/w3.org/TR/rdf-schema
"RDF Primer", http://www.w3.org/TR/rdf-primer/
"Resource Description Framework (RDF): Concepts and Abstract Syntax", http://www.w3.org/TR/rdf-concepts/
"RDF Semantics", http://www.w3.org/TR/rdf-mt/

RISESTEP: "enteRprise wIde Standard accEss to STEP distributed databases"Esprit project 20459, http://www.plm-interop.net/risestep

SAVE: "Step in A Virtual Enterprise" Bright Euram project 97-5073 , http://www.plm-interop.net/save

Schenck D. And Wilson P. (1994), "Information Modeling – The EXPRESS Way", Oxford University Press, ISBN 0-19-508714-3, Chapter 2, p17-29

SEINE: "Standards pour l'Entreprise Innovante Numérique Etendue", http://www.seine-plm.org

Slimany K. (2006), « Système d'échange et partage de connaissances pour l'aide à la conception collaborative », Computer Sciences thesis, Université C.B. Lyon 1.

Sowa J. (1984), "Conceptual structures: Information Processing in Mind and Machine", Addison-Wesley, Reading, MA.

SPEM 1.0: "Software Processing Engineering Metamodel Version 1.0", OMG, November 2002, formal/02-11-14, http://www.omg.org/cgi-bin/doc?formal/02-11-14

SPEM 2.0:"Software & Systems process Engineering Metamodel Specification Version 2",OMG, April 2008, http://www.omg.org/spec/SPEM/2.0
Formal/2008-04-01: Formal specification, http://www.omg.org/spec/SPEM/2.0/PDF
Ptc/07-08-08, Ptc/07-08-09: collection of schema files, http://www.omg.org/spec/SPEM/20070801/SPEM/20070801

S-TEN: "Intelligent Self-describing Technical and Environmental Networks" at http://www.s-ten.eu

STEP (2006), Standardization, ISO - International Organization for *STEP*, 2006, http://www.iso.org/iso/iso_cafe_step.htm.

Swenson K. (2007), "Welcome to introduction and Innovation 2007: Workflow and BPM in the New Enterprise architecture", The business and transformation conference, Washington, May 2007

Swenson K. (2008), "Welcome", introduction of Workflow, BPM & New enterprise Architecture workshop at Renaissance Washington DC, April 2008
Tolk A. (2003) "Beyond Technical Interoperability—Introducing a Reference Model for Measures of Merit for Coalition Interoperability," *Proceedings of the 8th ICCRTS*, Washington, D.C., June 17-19, 2003.

SysML 1.1: "OMG Systems Modeling Language Version 1.1", OMG, November 2008, http://www.omg.org/spec/SysML/1.1/
Formal/2008-11-02: SysML 1.1 specification, http://www.omg.org/spec/SysML/1.1/PDF
Ptc/2008-05-01: XMI serialization, http://www.omg.org/spec/SysML/20080501

TOGAF: "The Open Group Architecture Framework v9",Open Group, February 2009, http://www.opengroup.org/architecture/togaf9-doc/arch

Tolk A. (2004), "Composable Mission Spaces and M&S Repositories – Applicability of Open Standards" In proceedings of Spring 2004 Simulation Interoparebility Workshop (SIW), available at http://www.sisostds.org

Tolk A. and Muguira J. (2003) "The Levels of Conceptual Interoperability Model", Fall Simulation Interoperability Workshop Orlando, Florida

Tricot C. "Cartographie sémantique: des connaissances à la carte". http://ontology.univ-savoie.fr/tricot/download.php?file=recherche/CS/these/Cartographie%20semantique%20-%20Christophe%20Tricot%20-%20memoire.pdf , 2006

UML 1.4.2: "Unified Modeling Language 1.4.2", ISO/IEC 19501, OMG, September 2001
Formal/2001-09-67: http://www.omg.org/spec/UML/1.4/PDF
Ad/2001-02-15: http://www.omg.org/spec/UML/20010967/01-02-15.xmi
Ad/2001-02-16: http://www.omg.org/spec/UML/20010967/01-02-16.dtd

UML 2.1.2: "Unified Modeling Language 2.1.2", ISO/IEC 19505, OMG, February 2009 Formal/2009-02-04: "Infrastructure specification", http://www.omg.org/spec/UML/2.2/Infrastructure/PDF
Formal/2009-02-02: "Superstructure specification", http://www.omg.org/spec/UML/2.2/Superstructure/PDF
Ptc/2008-05-06: XMI, http://www.omg.org/spec/UML/20080501/Superstructure.xmi
Ptc/2008-05-07 : XMI, http://wwww.omg.org/spec/UML/20080707/uml-L0-model.xmi
Ptc/2008-05-08 : XMI, http://www.omg.org/spec/UML/20080501/uml-L1-model.xmi
Ptc/2008-05-09 : XMI, http://www.omg.org/spec/UML/20080501/uml-L2-model.xmi
Ptc/2008-05-10 : XMI, http://www.omg.org/spec/UML/20080501/uml-L3-model.xmi
Ptc/2008-05-11 : XMI, http://www.omg.org/spec/UML/20080501/uml-LM-model.xmi
Ptc/2008-05-12 : XMI, http://www.omg.org/spec/UML/20080501/infrastructure.xmi

WSRP 1.0: "Web Service for Remote Portlets Specification", OASIS Standard, August 2003, http://www.oasis-open.org/committees/wsrp

WSRP 2.0: "Web Services for Remote Porlets Specfication v2.0", OASIS Standard, April 2008, http://docs.oasis-open.wsrp/v2/werp-2.0-spec.html

XML: "Extensible Markup Language (XML) 1.0 fifth edition", W3C, November 2008, http://www.w3.org/TR/REC-xml/

XMI 2.1.1: "XML Metadata Interchange version 2.1.1", OMG, December 2007, http://www.omg.org/spec/XMI/2.1.1

Formal/2007-12-01: XMI specification, http://www.omg.org/spec/XMI/2.1.1/PDF
ptc/2007-10-06: XMI, http://www.omg.org/spec/XMI/20071001/07-10-06.xsd

XPDL 2.1: "Final XPEDL 2.1 specifications", WFMC, WFMC-TC-1025-Oct-10-08, 2008, http://www.wfmc.org/

XPDL 1.0: "Workflow Process Definition Interface – XML Process Definition Language", WFMC, October 2002, http://www.wfmc.org

Zachman J. "A framework for information systems architecture." *IBM systems journal* 26 (1987): 276-292.

Zeigler B. (2003) during the Panel Discussion on "Priorities for M&S Standards," Spring Interoperability Workshop, Orlando, FL

Annex

List of demonstrators and tools developed during the thesis

The list of demonstrator and tools which I develop during the thesis are the following:

1. STEP Mapper: a tool written in perl allowing to transform an EXPRESS Schema and a Part 21 file in an OWL ontology, and to transform an EXPRESS schema in annotated UML in order to use it for automated generation of application using AndroMDA
2. Networked Collaborative Product Development Platform: demonstrator for ATHENA program
3. Standard assessment platform for the SEINE project
4. AP214 ontology in OWL, manually produced
5. OpenDevFactory components: library of UML profile produced through transformation of existing models

List of publications

2006

"Enabling interoperability of STEP application Protocols at meta-data and knowledge level", N Figay, R. Jardim-Goncalves A. Steign Garcao. Indersience 36(4): 402-421 2006

2007

Conference MICADO INDAO 2007, Lyon : "Partage de données produit par le biais de processus de collaboration : Solutions proposées par le projet ATHENA »

Contribution to the book Enterprise Interoperability II New Challenges and approaches Volume package Enterprise Interoperability 2007, Harcover ISBN: 2007
"The ATHENA interoperability Framewok", A. Berre, B Elvesoeter, N. Figay, C. Guglielmina, S. Johnsen, D. Karlstern, T. Knother, XVIIIn 894 p 322

2008

Conference I-ESA 2008, Berlin, Germany
"Collaborative Product Development: EADS Pilot based on ATHENA results", N. Figay and P. Ghodous, pp 423-425 of proceeding

Contribution to "Unleashing the potential of the European Knowledge Economy Value Proposition for Enterprise Interoperability"

2009

Conference guest at Ontology Summit 2009 Panel Session: "Toward Ontology Based Standards", 26 March 2009 with online proceeding.

MEDES 2009: "Innovative Interoperability Framework for Enterprise Applications within Virtual Enterprises", N Figay, P Ghodous, accepted for publication

SITIS 2009:

"FLOSS as Enterprise Applications Interoperability Enabler", N. Figay, P. Ghodous

"Extended hyper model for interoperability within the Virtual Enterprise", N. Figay, P. Ghodous