HÖGSKOLAN
SKÖVDE

School of Humanities and Informatics
Master Dissertation in Computer Science 25p
D-level
Spring 2005

An Analysis of a Content of a Method Chunk Repository concerning Interoperability Problems

Andreas Ottoson

HS-IKI-MD-05-002

An Analysis of a Content of a Method Chunk Repository concerning Interoperability Problems

Submitted by Andreas Ottoson to the University of Skövde as a dissertation towards the degree of M.Sc. by examination and dissertation in the School of Humanities and Informatics.

2005-08-15

I hereby certify that all material in this dissertation which is not my own work has been identified and that no work is included for which a degree has already been conferred on me.

Signed: ____

Supervisor: Per Backlund

An Analysis of a Content of a Method Chunk Repository concerning Interoperability Problems

Andreas Ottoson

Abstract

The increasing complexity of Information Systems (IS) calls for IS development methods to be adapted to the specific situations of the projects at hand. Method engineering is important because it focus on the creation of new methods that can be used in the system development process with the aim of constructing new information systems. The size and complexity of projects for developing information systems are becoming larger and more complicated. Therefore, development methods turn out to be one of the most significant key factors to achieve great success of development projects.

The discipline of Situational Method Engineering (SME) focuses on the creation of new project specific methods. SME is a reuse strategy to assemble reusable method fragments or method chunks originating from different methods. New methods can be constructed from a method repository by selecting the chunks that are the most appropriate to a given situation. Thus, method chunks are the basic building blocks for constructing methods in a modular way.

This dissertation have identified, analyzed and categorized a set of interoperability problems for the content of a method chunk repository. These problems have been represented as a set of patterns. The patterns will facilitate the understanding of specific interoperability problems that belong to a specific method chunk. If we are aware of the different interoperability problems that exist it is more likely that we can use a method chunk successfully within the context of SME.

Keywords: Method, Method Engineering, Method chunk, Method chunk repository, Interoperability, Interoperability problem.

Acknowledgements

I would like to thank my supervisor Doctor Per Backlund for his valuable guidance and support. It has been an honor to work with you during this MSc year. The engouragement that you have given me has been invaluable.

Table of contents

List of figures

1 Introduction

This chapter describes the problem area and states the aim for this dissertation. A closer presentation of the different chapters in this dissertation is also outlined.

1.1 Problem Area

Interoperability is the most critical problem facing businesses that need to access information from multiple information systems.
What need to be done is to facilitate semantic interoperability (Park & Ram, 2004).

Using a successful method for a specific situation or project is a difficult task when building a successful information system (IS), since there exists many different methods and new methods are constantly being developed (Brinkkemper, 1996). Methods are important within the context of information systems development. If you do not have a method in use when creating an information system there will not be created a successful information system or the information system might end up as a failure (Andersen, 1994).

The increasing complexity of IS calls for IS development methods to be adapted to the specific situations of the projects at hand. Therefore, development methods turn out to be one of the most significant key factors to achieve success of development projects (Brinkkemper, Saeki & Harmsen, 1998).

According to Brinkkemper (1996) methods are being developed and employed over years, but a structure to take stock, generalize, and evaluate is actually needed. That is why the concept method engineering is important within the area of information systems development.

Method engineering is important because it let us create new methods and tools which can be used in the system development process with the aim of constructing new information systems. The size and complexity of projects for developing information systems are becoming larger and more complicated. Therefore, development methods and supporting tools turn out to be one of the most significant key factors to achieve great success of development projects (Brinkkemper, Saeki & Harmsen, 1998). Even though information system developers recognize the utility of system development methods, they have a lot of difficulties to use them (Ralyté, 2004). Each system development project has a different engineering situation asking for a specific method to satisfy it. Ralyté (2004) means that one solution to this problem is to adapt an existing method to the specific situation at hand. However, traditional methods are difficult to change. The discipline of Situational Method Engineering (SME) focuses on the creation of new project specific methods to resolve the problem with traditional methods.

This dissertation is part of a task group, within the INTEROP, named: **Methods and Methods Engineering for interoperability**. The main goal of this task group is to provide a systematic assembly-based method engineering approach for constructing methods supporting different interoperability problems.

In order to understand what interoperability problems to support these have to be elicited from a number of State-of-the-art (S-o-A) reports in order to be further analysed.

I have identified and elicited interoperability problems in these S-o-A reports developed in INTEROP. In order for the interoperability problems to be easier to understand I chose to categorise them into a set of patterns. These patterns (my results) are one part of a completed method chunk as illustrated in figure 1.

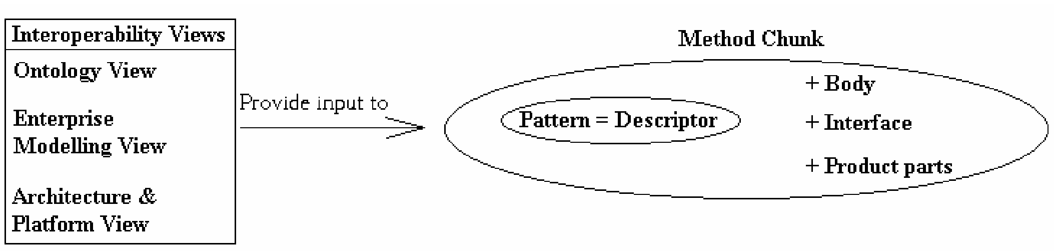


Figure 1: The results as part of a method chunk.

Figure 1 shows that a completed method chunk has to have a body, an interface, a product part and a pattern/descriptor. An identified pattern can be seen as a descriptor. In other words the patterns are describing interoperability problems categorized within three views. The different views are Ontology, Enterprise Modelling (EM) and Architecture & Platform (A&P). These views are considered important and have been proven to be useful earlier by two major research projects within the area of interoperability (IDEAS and ATHENA).

When all parts (see Figure 1) are brought together, a completed method chunk is created. The method chunks will be stored in a method repository and allows IS-developers to create their own method for a specific situation, thus providing a systematic assembly-based method engineering approach for constructing methods supporting different interoperability problems. However the results are also important within the concept of interoperability on a general level. Some of the elicited interoperability problems have been further analysed with respect to other author's view. In this way strengths and weaknesses can be compared, thus allowing an understanding on a more generalised level.

Basically method chunks for interoperability have not yet been defined (Ralyté & Rolland, 2001). In the area of research I find it important and interesting to do an analysis of a content of a method chunk repository for methods supporting interoperability problems. This dissertation will identify interoperability problems along with proposed solutions. In order to do this a number of State-Of-The-Art articles have been received for an explicit review. The S-o-A articles have been created by the INTEROP community, and it is my task to do an analysis of this material.

1.2 Aims and Objectives

This dissertation intends to categorize and analyse interoperability problems from a large number of state-of-the-art reports. These reports have been created by a research community called the INTEROP. The work will provide a set of interoperability patterns and discuss how they may be used.

The aim of this dissertation is to identify, analyze and categorize a set of interoperability problems for the content of a method chunk repository concerning interoperability problems.

In order to categorize a set of interoperability problems any instances of problems, concerning interoperability, found in the S-o-A will be elicited.

To achieve this aim, six objectives have been formulated in order to enhance the process. These are the objectives:

The following objectives will be achived by doing a literature study.

1. Characterize the concept of methods and method engineering.
2. Characterize the concept of method chunk repository.
3. Give an understanding of interoperability.

The following objectives will be achived by doing an analysis of the S-o-A material.

4. Identify a set of interoperability problems from the S-o-A reports to be resolved.
5. Identify a set of solutions for the identified interoperability problems.
6. Categorize the identified interoperability problems and propose a part of a structure for a method chunk before inserted into a method repository.

1.3 Thesis Outline

This chapter provides an overview of the content for every chapter within this dissertation.

This dissertation is targeted against reader groups within an academic society interested in method engineering, especially the INTEROP community. The results are relevant for building a method chunk. The results may also be of interest for reader groups studying interoperability within Information System Development in general.

Chapter 2 describes and explains relevant concepts in order to understand the area of interest and the aim of this dissertation. The relevant concepts are; methods, method use, method engineering, situational method engineering, method chunk, method chunk repository and interoperability.

Chapter 3 presents the research approach for this dissertation. It provides an overview of decisions taken in order to achieve the formulated objectives of this dissertation, and eventually reach its aim. The process of reaching the objectives is also described, as well as the relevance and importance of the achieved results.

In chapter 4 the achieved results are presented. The results are further analysed and compared to existing literature. The results are presented as a set of patterns and the analysis is presented after every specific pattern.

Chapter 5 discusses the process and the results. It also provides some directions for future research.

2 Interoperability

Establishing interoperable systems is a complex operation and goes far beyond the technical interconnectedness of databases and systems. Interoperability emerges from the need to communicate data across different domains for a specific purpose. (Park & Ram, 2004)

2.1 On Interoperability

Park & Ram (2004) mean that Interoperability is the most critical problem facing businesses that need to access information from multiple information systems. What need to be done is to facilitate semantic interoperability among distributed and heterogeneous information systems (Park & Ram, 2004). We must also be able to handle semantic conflicts (from heterogeneous sources) of various kinds, this can be done by using Ontologies.

Various definitions on interoperability exist. Interoperable means 'able to operate in conjunction' according to the Oxford dictionary. Another way to look at the concept is from a software engineering point of view, interoperability means that two co-operating software systems can easily work together without a particular interfacing effort. It also means establishing communication and sharing information and services between software applications regardless of hardware platforms (INTEROP documentation, WP9). IEC TC65/290/DC defines Interoperability as:

> *"The ability to integrate data, functionality and processes with respect to their semantics"* (INTEROP documentation, WP9, page. 13).

Interoperability can also be defined as:

> *"The ability of two or more devices, regardless of manufacturer, to work together in one or more distributed applications. The application data, their semantic and application related functionality of each device is so defined that, should any device be replaced with a similar one of different manufacturer, all distributed applications involving the replaced device will continue to operate as before the replacement, but with possible different dynamic responses"* (INTEROP documentation, WP1, page 269).

The INTEROP Community propose the following definition:

> *"The capability of a system or a product to work with other systems or products without specific effort from the user"* (INTEROP, page. 4).

It is considered important to understand that when two or more interoperable parts can do something more together interoperability is achieved. However, the concept of interoperability is thought of as complex since there exist many definitions which vary. In some of the above mentioned definitions the concept of semantic is introduced, therefore the relevance of semantics in contrast to interoperability is considered important to discuss, which will be done in the following chapter.

Chen & Doumeingts (2003) mean that semantics runs through all layers of an Organization. Semantics involves the area of linguistics dealing with symbols and differences in the meaning of words. Semantics could infer an approach to treat meaning as a relationship between signs and human behaviour (Stamper, 2000).
Chen & Doumeingts (2003) incorporates semantics across three layers as illustrated in figure 2.

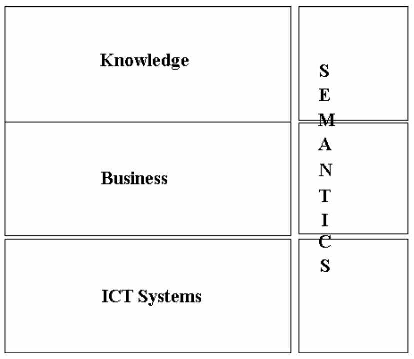


Figure 2: Semantics within an organisation.

- The knowledge level deals with knowledge assets required to run the business, such as that concerning production, administration, human resources management, laws and regulations, etc.
- The business level deals with organizational structures models, decision making processes and relations to the external environment.
- The Interoperability Content Taxonomy (ICT) level relates to the technologies that are required for interoperability by inter and intra-organization communication, co-operation and collaboration.
- The semantic level is essential for relationships between each of the three other levels.

Businesses must have a common and agreed understanding of the terms of business. This will be enabled by the creation of a working ontology for the capturing of semantics. (INTEROP documentation, WP 1)

Backlund, P., Söderström, E. & Wangler, B. (2004) say that interoperability must be achieved on all layers of an enterprise in order to achieve meaningful interoperation between different enterprises. In order to get the necessary understanding between enterprises that want to collaborate semantic descriptions can be used according to Backlund et al. (2004).

The repository of method chunks enables the possibility to store method chunks addressing different IS interoperability problems as for example requirements specification for interoperable systems and applications, integration of IS components, etc. These chunks can then be assembled and tuned to respond some particular interoperability situation. SME approaches have to deal with interoperable method chunks and have to provide mechanisms to define differences or similarities of method chunks, in order to enable method chunks assembly. Therefore, the first challenge of SME is the method chunk definition. The method chunks for IS interoperability must be able to help finding method solutions to resolve different IS interoperability problems. (Arni-Bloch, 1999)

Arni-Bloch (1999) says that to be able to enable assembly based method engineering there have to be a repository containing a large number of method chunks. One of the possibilities to quickly fill the method repository of an organization is to exchange method chunks between organizations. The question is how to ensure the compatibility between different kinds of method chunks? (Arni-Bloch, 1999). The incompatibility can be detected in different levels: classification parameters, modeling languages, process or product models specification. Moen (2001) say that interoperability is multifaceted and can be analysed into different levels of abstraction. By understanding these levels interoperability can be easier to categorise and understand. The concept is a fundamental challenge for networked information discovery and retrieval (Moen, 2001). But this interoperability challenge is not specific to method engineering domain. It can be compared with other kind of inter-organizational exchanges such as information, data, and applications. (Arni-Bloch, 1999)

2.2 Different aspects of Interoperability

This dissertation will retrieve interoperability problems from three different perspectives; enterprise view, ontology view and architecture and platform view since these domains are considered important and have been elicited earlier by IDEAS (Interoperability framework). IDEAS analyzed interoperability aspects from an enterprise view (i.e. between two or more enterprises) an architecture & platform view (i.e. between two or more applications/systems), and an ontological view (i.e. the semantics in interoperability) (INTEROP documentation, WP9).

Enterprise View

The aim of this view is managing to represent in which way different Models adopted in the inter-operating enterprises could affect the IDEAS reference framework.

Enterprise modelling can be defined as:

> *"The set of activities or processes used to represent the various parts of an enterprise model in order to address some desired modelling finality"*
>
> (INTEROP documentation, WP 1).

Finality indicates the creation of explicit facts and knowledge that add value to the enterprise or can be shared by business applications and users for the improvement of performance within the enterprise. The improvement of the enterprise performance is the resultant of many parameters that can be viewed as the goals of EM. One of the superior goals of enterprise modelling is to support the analysis of an enterprise, and more specifically, to represent and understand how the enterprise works, to make better decisions about enterprise operations and organization, or to control, coordinate and monitor some parts of the enterprise. (INTEROP documentation, WP 1)

Architecture & Platform View

The aim of this view is managing to represent in which way different architectures and data adopted in the inter-operating enterprises could affect the IDEAS reference framework.

This view is concerned of the architectures and platforms in terms of standards and standardization projects that provide architectural specifications relevant to enterprise integration, on e-business frameworks that provide architectural guidelines to support interoperability of inter-enterprise systems, on web services architectures, as well as on development standards and environments. (INTEROP documentation, WP 1)

Ontological View

The aim of this view is managing to represent a vocabulary that has a semantic meaning in order for agents to interoperate to each other (i.e. semantic web). With other words Ontologies are important in order to communicate.

Ontologies have the advantage of using its ability to capture the knowledge within a certain domain in order to provide a good conceptualization of data objects and their relationships. Its knowledge is domain-specific, but independent of particular schemas and applications. (Park & Ram, 2004)

By understanding the concept of interoperability in the context of the above mentioned dimensions it will be easier to understand that there exist several interoperability problems within each of these dimensions. These problems are important to elicit in order to use a method chunk with success.

According to Fitzgerald (1997) the lack of empirical research on systems development in real organisational contexts has been criticized by researchers, and there have been interests for a clearer understanding of the realities of software development. Therefore, more research is needed into the actual practice of systems development in organizations.

This statement is considered important for the method repository and for the completed method chunks as well as for the concept of interoperability. In order for us to know how the repository can be best implemented the theories have to be used in practice. They must be used in a real environment to let us understand difficulties and hopefully fill this gap.

In this dissertation important concepts are necessary to be further defined, explained and discussed in order for the achiveness of objective 1, 2 and 3, but also to provide an entirety.

2.3 Methods and Method Use

Avison and Shah (1997) and Andersen (1994) consider system development as a very complex process since technology is evolving everyday. Today there are few organisations that have the competence to develop their own information system within the corporation. Stone (2003) states that the system development process must be: based on best practises, self improving, easy to implement, acceptable by experienced system developers and adjustable for projects no matter the size in order to be successful. An information system can be developed in many ways and different types of methods can be used. The word ‘method’ comes from the Greek ‘methodos’, which means way of investigation. A good method can help to capture, develop and describe the requirements on the information system (Andersen, 1994). A good method should also help to elicit the correct system within a reasonable amount of time according to Andersen (1994). Information systems development methods are often used during development of information systems in order to guide and support the information systems development process. Hence, methods are created and used to support information systems developers as they carry out different tasks, aiming to reach some goals (Ågerfalk & Åhlgren, 1999). Jayaratna (1994) gives the following definition of a method:

> *“An explicit way of structuring one’s thinking and actions”*
>
> (Jayaratna, 1994, page. 2).

Brinkkemper (1996) defines a method as:

> *“An approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rules, structured in a systematic way in development activities with corresponding development products”*
>
> (Brinkkemper, 1996, page. 275).

In another article Brinkkemper (2000) gives a different definition of the concept method:

> *“A method is an integrated collection of procedures, techniques, product descriptions, and tools, for effective, efficient, and consistent support of the IS engineering process”* (Brinkkemper, 2000, page. 125).

The concept “method” is thought of as complex and difficult to understand, based on the above definitions and explanations of what a method is. Examples of methods for information system development are SSADM and RUP. Methods are usually described in textbooks and manuals giving the step-wise structuring of the development activities and the structural requirements for the final product (Brinkkemper, 1996). Using the correct method for a specific situation or project is a difficult task when building a successful information system, since there exist many different methods and new methods are constantly being developed.

There is also criticism against system development and methods today. Fitzgerald (2000) says that system development methods are not being accepted and used by developers. The reason for non-use is that currently available methodologies do not suit the profile of the development prevailing in the analyzed organizations. The same author mean that most of the system development methodologies in use today have their origin in a set of concepts that came to prominence in the 10- year period from about 1967 to 1977. From my point of view this can be interpreted as a problem of digging in the same place without finding much new.

In order to characterize the use of methods three “tensions” are described; the concept usage tension, the concept implementation tension and the product usage tension (Lundell & Lings, 2004). To summarise the three tensions *the concept implementation tension* (between the method proponents and the product developer) is a discrepancy between proposals for new concepts (methods) and it’s embodiment in the tool that is being delivered. *The product usage tension* (between the IS developer and the product developers) is a discrepancy between the tool that is being delivered and the stakeholders expectations for it. This discrepancy can result that the stakeholders formulate their own opinions on how effective a tool is. At last the *concept usage tension* (between the method proponents and the IS developer) is a discrepancy of how a method is described and how it is being used in reality. Figure 3 illustrates these important tensions.

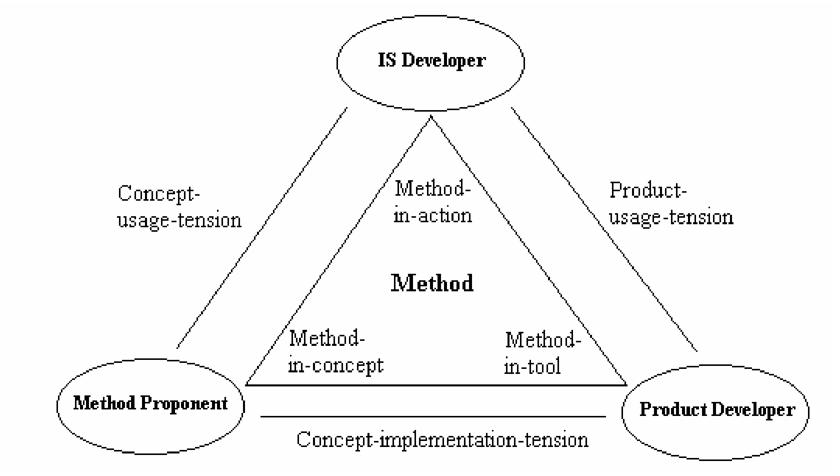


Figure 3: The stakeholder triangle (From Lundell and Lings (2004)).

Iivari and Maansaari (1998) say that the large selection of development methods implies that there are several roles involved in using them, for example; analysts, developers, system users, and project managers. A method may act like a tool since it serves different roles within system development.

The tensions in figure 3 are based on different perspectives for a specific actor. The actors involved have the following roles:

- IS developers – Develops the information system.
- Product developers – Develops the CASE tool.
- Method proponents – Develops the method.

There are three tensions between the actors mentioned above. However the tension most important for this dissertation is the concept usage tension, since it is considered important to understand how methods are used.

The concept usage tension: This tension focuses on how IS developers should act according to a method and how the IS development process actually proceeds with respect to the same method. By getting an understanding of the tension, a better use of a method will be achieved. This is considered important since the tension should be in mind when developing new methods in order for them to be of better use. Introna and Whitley (1997) state that a method means much more than how it is described. A successful use of a method can be connected to a wider understanding than just the method knowledge. A personal knowledge about the surroundings has to be taken into account. Without this knowledge the usage of a method often fails (Lundell & Lings, 2004).

Concerning the concept of methods I intend to use Brinkkemper´s definition (Brinkkemper, 2000, page. 125) since it is concidered important to support the information system engineering process. The knowledge of how to use methods (concept usage tension) in order to create a successful information system is also considered relevant for the concept of methods and method use.

Brinkkemper (1996) means that the term method engineering is necessary because it provides a structure within the area of methods that are being developed. In order to gain a better understanding of the term method engineering the following chapter describes and defines the concept.

2.4 Method Engineering

In this chapter the concepts of method engineering will be defined and further discussed.

As software engineering is concerned with all aspects of software production, method engineering is dealing with all engineering activities related to methods. Iivari (2000) mean that the need to customize methods is explained by the variety of systems that are developed and the various situations in which information systems can be developed. Brinkkemper defines method engineering as:

> *"The discipline to design, construct and adapt methods, techniques and tools for the development of information systems"* (Brinkkemper, 2000, page. 125).

2.4.1 Situational Method Engineering

The increasing complexity of Information systems (IS) asks for new IS development methods constructed "on the fly" to be adapted to the specific situations of the projects at hand. Situational Method Engineering responds to this need by offering techniques to construct methods by assembling reusable method fragments stored in some method repository. (Ralyté & Rolland, 2001)

SME facilitates the construction of situational methods. Let us first consider what a situation is before the concept of situational method can be understood. Brinkkemper (2000) defines a situation as:

> *"The combination of circumstances at a given moment, possibly in a given organisation, or in a given role"* (Brinkkemper, 2000, page. 126).

This definition let us understand the concept of situational method which is defined as:

> *"An information system engineering method tailored and tuned to a particular situation"* (Brinkkemper, 2000, page. 126).

Arni-Bloch (1999) defines a situational method as:

> *"The method which is constructed for a specific project in order to perfectly satisfy its specific situation"* (Arni-Bloch, 1999, page. 2).

This dissertation will use Brinkkemper´s definition for a situational method since it is clear that we are talking about the development of IS and the method is specific for a particular situation. Brinkkemper et al. (1999) mean that SME is the discipline to build project-specific methods, called *situational methods,* from parts of already existing methods. SME can also be seen as the construction of methods which are tuned to specific situations of development projects (Welke & Kumar, 1991). According to Ralyté & Rolland (2001) SME is a reuse strategy to assemble reusable method fragments or method chunks originating from different methods. In this way a new method can be tailored to the project situation at hand. The concepts method fragment and method chunk will in this dissertation mean the same thing. The following definitions will help us in understanding what method chunk really is:

> *"Method chunks are coherent pieces of IS development methods"* (Brinkkemper, 1996, page. 278).

> *"A method chunk is a cohesive, autonomous and interoperable part of a method"* (Arni-Bloch, 1999, page. 2).

> *"Method chunk is ´process-driven´ in the sense that a chunk is based on the decomposition of the method process model into reusable guidelines"* (Ralyté & Rolland, 2001, page. 3).

Based on these definitions method chunks are considered important and necessary when dealing with SME. It is also relevant to understand that the concept of method chunk also should be considered as a process. This process is further discussed in the next chapter.

2.4.2 The Method Reengineering Process

Ralyté & Rolland (2001) mean that the core of a method chunk is its guideline to which are attached the associated product parts needed to carry out the process encapsulated in the guideline. A guideline is defined as:

"A set of indications on how to proceed to achieve an objective or perform an activity" (Ralyté & Rolland, 2001, page. 4).

New methods can be constructed by selecting the chunks that are the most appropriate to a given situation from a method repository. Thus, method chunks are the basic building blocks, for constructing methods in a modular way. SME favors the construction of modular methods that can be modified and augmented to meet the requirements of a given situation (Ralyté & Rolland, 2001). Ralyté & Rolland (2001) mean that the prerequisite for modular method construction is a method repository (Method base) containing a large collection of method chunks. This requires a reengineering process of existing methods to produce the method chunks that populate the method repository. Figure 4 shows the reconstruction of the existing IS engineering methods in a modular way. The result is a collection of reusable method chunks, which are stored in the method base. Once the method base is populated with a number of chunks, the construction of a new method is possible through the retrieval of those chunks that match the characteristics of the project situation at hand and their assembly to form the new method.

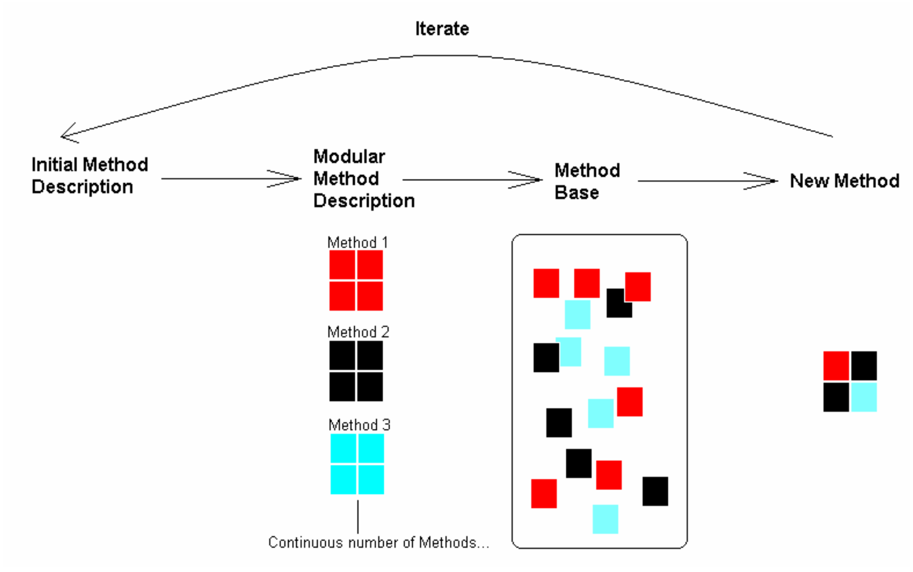
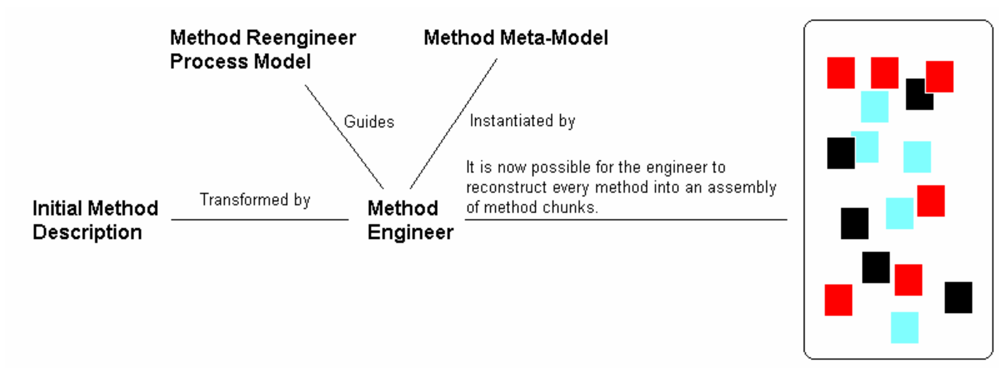

Iterate
Initial Method Description
Modular Method Description
Method Base
New Method
Method 1
Method 2
Method 3
Continuous number of Methods...

Figure 4: Method chunk reassembly process.

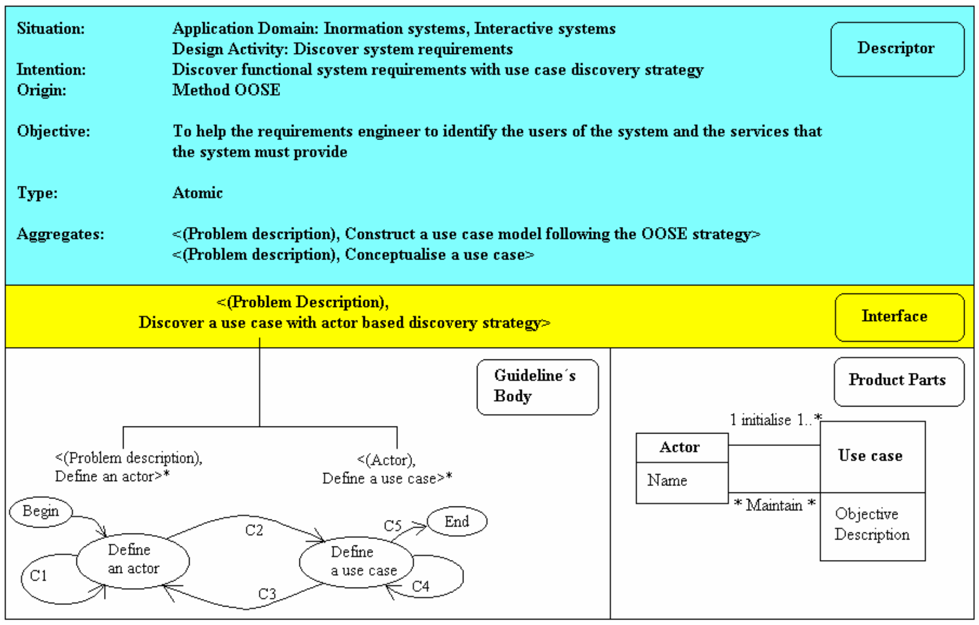
In order for an engineer to create new method chunks from existing methods there also exist an approach for method reengineering (Figure 5 summarizes this approach). This approach can be seen as a guideline that help or support the engineer to apply the method chunk reassembly process which is described above (Ralyté & Rolland, 2001).



It is now possible for the engineer to reconstruct every method into an assembly of method chunks.

Figure 5: The approach for method reengineering.

As shown in figure 5 the method reengineering process model guides the method engineer in the reconstruction of every method into an assembly of method chunks. While guided by this process model, the method engineer instantiates the method meta-model to describe the identified method chunks. The main objective of the method reengineering approach is for the method engineer to define a guideline, identify a method chunk and finally to define a method chunk. Figure 6 is an example of how a defined method chunk can be represented.



Situation:	Application Domain: Inormation systems, Interactive systems Design Activity: Discover system requirements
Intention:	Discover functional system requirements with use case discovery strategy
Origin:	Method OOSE
Objective:	To help the requirements engineer to identify the users of the system and the services that the system must provide
Type:	Atomic
Aggregates:	<(Problem description), Construct a use case model following the OOSE strategy> <(Problem description), Conceptualise a use case>

<(Problem Description),
Discover a use case with actor based discovery strategy>

Figure 6: Example of a completed method chunk (From Ralyté & Rolland (2001)).

The aim for this dissertation has a strong connection to the descriptor (illustrated in figure 6) within the context of a method chunk. The idea is to create different patterns that belong to a specific method chunk.

In software development a pattern can be described as a written document that describes a general solution to a design problem that recurs repeatedly in many projects (Borchers, 2001). The same author says that software designers adapt the pattern solution to their specific project. Graziella (1996) mean that patterns use a structured approach to describing a design problem, its proposed solution, and any other factors that might affect the problem or the solution. A successful pattern should have established itself as leading to a good solution in three previous projects or situations (Graziella, 1996).

In the area of interest which in this case is interoperability problems a pattern will facilitate the understanding of specific interoperability problems that belong to a specific method chunk. If we are aware of the different interoperability problems that exist it is more likely that we can use a method chunk successfully within the context of SME (INTEROP community).

3 Way of working

This chapter describes the selected research approaches among with how the process was planned in order to achieve the aim for this dissertation. A closer presentation of the INTEROP community is also outlined.

3.1 Research approach

The research approach is of a qualitative nature where the extraction of data is focused on “soft” data by an interpretative analysis. A deductive approach (see figure 7) is going to be used within this dissertation. From existing theories (S-o-A) conclusions have to be made concerning interoperability problems and how these should be represented. Figure 7 shows the difference of using a deductive approach from an inductive one.

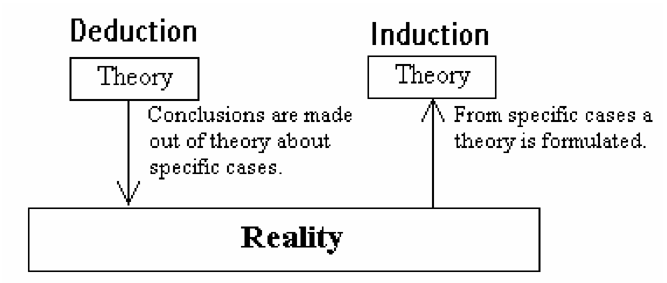

Deduction
Theory
Conclusions are made out of theory about specific cases.
Induction
Theory
From specific cases a theory is formulated.
Reality

Figure 7: Representation of Deduction and Induction (Patel & Davidson, 2003).

The work will commence with a study of literature which will fulfill objective one, two and three. The study of literature will also contribute to the fundamental understanding of the aim of this dissertation.

This dissertation will use an explorative investigation since information will be collected from various sources in order to achieve as much knowledge as possible concerning the specific problem area. I also intend to use a descriptive investigation because knowledge exists in the area of interoperability problems. However, these problems have not been characterized or systemized in the shape of models or method chunks. My way of working is to represent different interoperability problems as a set of patterns. In other words the patterns are descriptive.

In this dissertation a qualitative research approach is preferred since interpretative analysis is useful when creating a set of patterns.

3.2 Planning the process

In order to be able to elicit and categorize different interoperability problems it is important to make a deep investigation of the material that exists in the field. After that work is done the relevant material should be selected and made accessible. I chose to create different meta-models in order to categorize the material with respect to a set of interoperability problems. In this way an overview is represented and it is easier to find the original sources of facts.

According to Andersen & Schwenche (1998) research is about making the right decisions of how to do something in a correct manner. Successful research demands decisions that are made with awareness. One way of achieving these decisions is to plan your project carefully (Andersen & Schwenche, 1998). As for this dissertation planning was a vital ingredient because the material available had to be elicited, analysed and compared with carefulness in order to achieve the formulated objectives for this dissertation. Figure 8 shows the process to reach this dissertation's objectives.

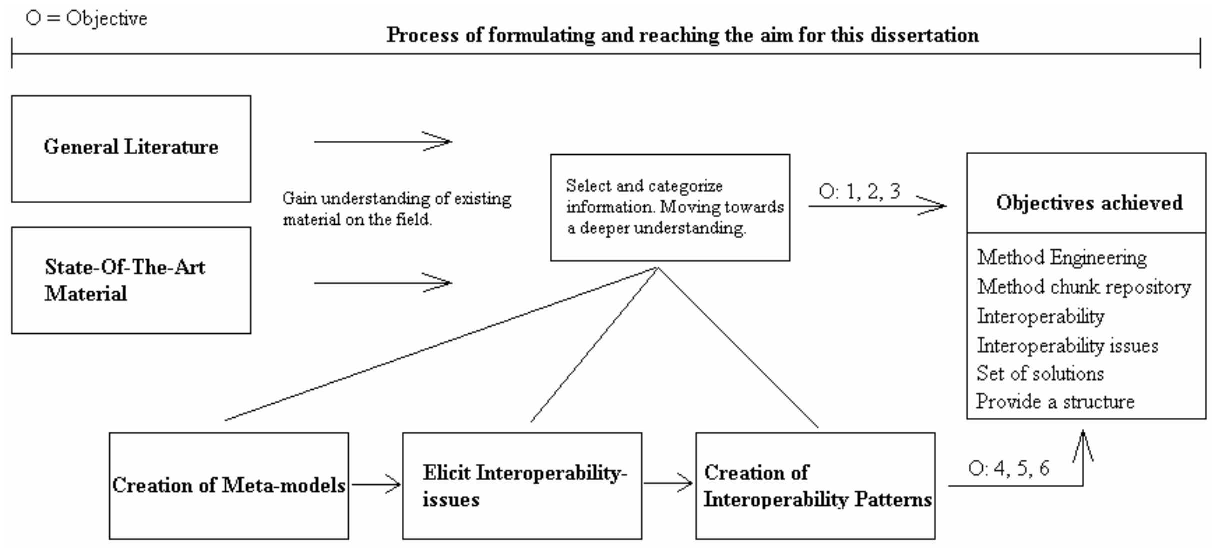

O = Objective
Process of formulating and reaching the aim for this dissertation
General Literature
State-Of-The-Art Material
Gain understanding of existing material on the field.
Select and categorize information. Moving towards a deeper understanding.
O: 1, 2, 3
Objectives achieved
Method Engineering
Method chunk repository
Interoperability
Interoperability issues
Set of solutions
Provide a structure
Creation of Meta-models
Elicit Interoperability-issues
Creation of Interoperability Patterns
O: 4, 5, 6

Figure 8: Planning the supply of material.

First a survey of existing material was made. Decisions of what information that was of importance were guided by a literature study. At this stage I was moving towards a deeper understanding of the area of interest.

In the next phase I made decisions concerning how to present and give an understanding of the problem. It is of importance to make the problem and the amount of information in the field easier for the reader to understand. Hence, some kind of simplification is needed when presenting the results. I chose to use a set of patterns since they represent an equal view of every interoperability problem elicited.

3.3 The INTEROP community

The results from this dissertation are of importance for a scientific community called INTEROP. INTEROP is a Network of Excellence supported by the European Commission for a three-year-period. INTEROP aims to create the conditions of an innovative and competitive research in the domain of Interoperability for Enterprise Applications and Software. Within the INTEROP community a number of universities collaborate. There are different task groups based on the participants different specialties. This dissertation is active in a task group named: **Methods and Methods Engineering for interoperability**. The main goal of this task group is to provide a systematic assembly-based method engineering approach for constructing methods supporting different interoperability problems. The project will combine two levels of knowledge: (1) Method Knowledge Level called *Methods / Method Chunks for Interoperability* and (2) Method Engineering Knowledge Level called *Assembly-based Method Engineering Approach for Interoperability*.

My area of research concerns the development of *Method chunks* for interoperability. The goal of the task group is to provide a repository of method chunks (INTEROP method repository) supporting different interoperability problems in the domains of Enterprise Modelling (EM), Ontology (ONT) and Architecture & Platforms (A&P). These domains are considered important and have been analysed earlier by for example the Athena project and IDEAS. The definition of the method chunks will be based on the work already realised by different WPs. Besides, new method chunks will be proposed. The activities to achieve this goal are the following:

1.1 To analyse the use of methods in organisations, especially when dealing with interoperability problems;

1.2 To make an inventory of the methods proposed by the INTEROP community;

1.3 To define method chunks based on the methods proposed by the INTEROP community and to put them in the INTEROP method repository;

1.4 To identify cases/scenarios in order to illustrate situation assessment for different method chunks selection and assembly.

This dissertation is strongly involved in point 1.3 since my aim is to contribute to the creation of method chunks as they are put into the INTEROP method repository. In other words my results will be used by other people within the same research community.

4 Results

The results of this dissertation have been created by an extensive analysis of the received S-o-A articles (appendix G). By eliciting interoperability problems from the material available, a set of patterns could be created. The patterns are supposed to be consequent in order to be easy to understand. This was done by allowing the patterns to include the same elements. In this case three elements are of importance:

1. Interoperability problem.
2. Context of the Interoperability problem.
3. Possible solution to the identified interoperability problem.

This chapter will present the patterns created together with a comparison to relevant literature concerning every identified interoperability problem. This is done in order to gain a better understanding of the elicited interoperability problems. The patterns will be categorized under the relevant domain (Ontology view, EM view and A&P view) which they belong to. For a better overview of the elicited interoperability problems and how they are connected to each other see appendix B, C, D, E and F. Appendix A is an overview of work package 1.

For a pattern to be classified in this dissertation and thereby be further analysed the three components must have been put together allowing the completeness of a created pattern. However, if there is a gap in any of these elements (information may not exist) the interoperability problem will not be further analysed, but instead be categorized as an incompleted interoperability problem. These incompleted interoperability problems are represented in appendix F.

A second literature study has been done for every classified interoperability problem. This literature study has been done in order to gain a more general understanding for a specific interoperability problem elicited. The results will make it possible to achieve objective four, five and six. In other words the presented results are a final way of reaching this dissertation's overall aim.

The patterns have different names and they also have a letter. The letter points out in what appendix the interoperability problem exists. This is done in order to facilitate the understanding of the original source for the problems.

4.1 Ontology View

IP 1: Semantic Overlapping

Problem – Things or properties within an enterprise is represented more than once in for example a model, this is called semantic overlap.

Situation (context) – An enterprise modeling language is free of overlap when no class of things or type of property is represented more than once in the language definition.

Solution – Integrate tightly a variety of models and modeling languages in order to facilitate enterprise modeling languages and models without semantic overlap, this is called facet modeling (A facet is a model of a component or system that provides information specific to a domain of interest. To support heterogeneity in designs, each facet may use a different domain model to provide domain-specific vocabulary and semantics. Facets are written to define various system aspects and are then assembled to provide complete models of components, component aggregations and systems).

Figure 9: Semantic Overlapping

The main characteristic of a "semantic overlap" is redundancy (see figure 9). By using different models and modeling languages there is a risk that the same objects or "things" can be represented several times with the help of different notations or paradigms. Redundancy is something that we do not want, because it can create confusion among developers (what is what?). It can also be a reason for ineffectiveness (Andersen, 1994). Dinesh & Solomon (2001) has done an empirical study that examines the efficacy of a using a consulting system for conceptual modelling with the aim of reducing modelling errors. By using this consulting system the incidence of errors committed by designers engaged in conceptual modelling can be reduced. The system can prevent errors in the presence of redundancy (Dinesh & Solomon, 2001). Interoperability problem one (semantic overlap) has it's origin from Work package 5.

IP 2: Data Integration

Problem – The data integration problem is concerned with the use of data from different (data) sources in one application. The data from the different sources needs to be presented to the user in a unified (and understandable) way.

Situation (context) – Data integration enables users to ask queries in a uniform fashion, without having to access independently each data source (or even know about the existence of multiple sources).

Solution – In an information integration system, users ask queries over a *mediated schema*, which captures only the aspects of the domain that are salient to the application. The mediated schema is only a logical one (no data is stored in it), and mappings are used to describe the relationship between the mediated schema and the schemas of the data sources.

The *semantic web* goes one step further. Here, there is no central mediated schema, tasks involve actions in addition to queries, and coordination is achieved using ontologies. Here, mappings between ontologies are necessary for some (software) agents to interoperate.

Figure 10: Data Integration

The data integration problem is common within the context of sharing data sources (see figure 10). For example there is a problem of integrating data from sources that lack common object identifiers (Cohen, 2000). What is needed is a natural-language for objects. By the use of mappings two different "names" (one and the same object has two different names) can be interpreted into a meaning of one and the same thing. A typical example is the object "car" which also can be an "auto", two different names representing the same object (Cohen, 2000).

IP 3: Ontological Integration

Problem – In an ontological situation we want two or more ontologies to represent the same thing, or the agents whom are using different ontological languages must be able to understand each other in order to communicate. With other words ontologies must be able to integrate in order to achieve a consensus. Exactly how to do this is a problem.

Situation (context) – two approaches for the use of different ontologies can be identified: the (1) *local model* and the (2) *global model approach.* The difference between these two approaches is whether, in interactions with the systems, the user can use his/her local data model, or whether the user needs to conform to a global model when interacting with the system:

In the *local model*, or local ontology, approach, every user is represented by an agent and this agent presents the user with its own local data model. The agent performs the translation between the user's local model and either the global model or other local models in order to allow interaction with multiple data sources in the sys tem. An example of the local model approach is the KRAFT project.

In the *global model*, or global ontology, approach, every user accesses the information through the global data model using a mediator, which is "a system that supports an integrated view over multiple information sources". Note that in the local model approach, a user agent will, in most cases, also contact a mediator in order to allow inter-operation of the systems, which contain multiple information sources.

Solution – Two major paradigms in ontology integration can be distinguished:

Ontology Merging and *Ontology Aligning.*

Ontology Merging consists in creating a new ontology from two or more existing ontologies. In this case, the new ontology will unify and replace the original ontologies. This often requires considerable adaptation and extension of the existing ontologies.

Ontology Aligning brings two or more ontologies into mutual agreement, making them consistent and coherent. Here, the ontologies are kept separate, but at least one of the original ontologies is adapted such that the conceptualization and the vocabulary match in overlapping parts of the various ontologies. However, the ontologies might describe different parts of the domain in different levels of detail.

Figure 11: Ontological Integration

Ontologies must be able to integrete (see figure 11). By reusing Ontologies in a well-defined way, domain models can be constructed more easily and made more robust according to Pirlein & Studer (1995). The same authors introduce an approach (KARO) which provides various means of retrieving and adapting components of an ontology as part of a domain model construction process. By integrating the proposed approach into a model-based and incremental knowledge engineering environment the reuse of a predefined ontology can be integrated into the development process of expert systems in a systematic way (Pirlein & Studer, 1995). Noy & Musen (2003) say that researchers in the ontology-design field have developed the content for Ontologies in many domain areas. This has led to a large number of Ontologies covering overlapping domains. In order for the Ontologies to be reused, they have to be merged or aligned to one another (Noy & Musen, 2003). The same authors have developed a suite of tools for managing multiple Ontologies. The tools provide users with a framework for comparing, aligning and merging Ontologies.

IP 4: Invalid Mapping

Problem – If either of the two ontologies involved in ontology mapping changes (evolves), the mapping might become invalid.

Situation (context) – We want different data sources to be able to integrate with each other. By using the semantic web different users have their own “agents” in order to access data sources. In order for these agents to communicate and get valid data there have to exist an ontology vocabulary. The ontologies enable the agents with relevant information. The agents can “understand” that some things are the same. For example they can understand that a “car” and an “auto” are having the same meaning. Though, one can change an ontology vocabulary.

Solution – The mappings between the ontologies need to evolve together with the ontologies: versioning of both the ontologies and the mappings is required.

Figure 12: Invalid Mapping

Mappings might become invalid if an ontology evolves (see figure 12). Crow & Shadbolt (2001) describes the Internet-based multi-agent problem solving (IMPS) architecture. It is designed to facilitate the retrieval, restructuring, integration and formalization of task-relevant ontological knowledge. The IMPS uses a multi-agent architecture to combine knowledge-levels models with a selection of web knowledge extraction in order to provide a clean syntactic integration of ontology knowledge from diverse sources and support a range of ontology merging operations at the semantic level (Crow & Shadbolt, 2001). The techniques described by Crow & Shadbolt (2001) can be applied to more task-based integration of knowledge from diverse sources. Most important the article mentions how to handle the update of both formal and informal Ontologies.

IP 5: Schema Integration

Problem – Most work on schema match has been motivated by *schema integration,* a problem that has been investigated since the early 1980s. The problem statement of schema integration is as follow: Given a set of independently developed schema, to construct a global schema which allows representing the same information as those schemas. This is the problem of integrating independently developed ontologies into a single ontology.

Situation (context) – Since schemas are independently developed, they often have different structure and terminology (WP 5, page 8). This can obviously occur when the schemas are from different domains, such as a real estate schema and property tax schema. It also occurs even if schemas represent the same real world domain, because they were developed by different people.

Solution – The first step in integrating schema is to identify and characterize some *inter-schema relationships*. This first step is a process of schema matching. Once inter-schema relationships are identified, matching elements can be unified under a coherent, integrated schema or view. During this integration, or sometimes as a separate step, programs or queries are created that permit translation of data stored according to the original schema into the integrated representation. To ensure semantic equivalence, something called the “Federated Information System” (FIS) must know about correspondences (logical metadata) between queries, federated and component schemas. These correspondences can be expressed through ontological descriptions or rules. They can be defined by humans, by giving expressions in some language, or inferred automatically.

Two kinds of federations can be distinguished: tight and loose federations

(WP5, page 10).

As I understand these federations have to be considered in order for different agents to communicate with different data sources.

Figure 13: Schema Integration

Beynon-Davies, Bonde, McPhee & Jones (1997) say that a number of formal approaches to conduct schema integration have been proposed, although they have not been successful. The same authors mean that it is the collaboration work of schema integration that is the problem (see figure 13).

IP 6: What to express?

Problem – When dealing with the representation of ontologies one problem is the difficult task of understanding what to express.

Situation (context) – By setting up a framework for surveying ontology representation languages, the information can then be used to in the description of specific languages and formalism, so to have a basic unified view for comparison. The framework is based on three main classification criteria (what to express, how to express and how to interpret the expression).

The first criterion takes into account that ontology is a generic term for denoting domain representation, but specific ontology languages may concentrate on representing certain aspects of the domain.

Solution – The focus on what to express are the following:
Class/relation. This class is used for referring to languages aiming at representing objects/classes/relations.

Action/process. This class is used for referring to languages that provide specialized representation structures for describing dynamic characteristics of the domain, such as actions, processes, and workflows. These languages may also incorporate mechanisms for the representation of the static aspects of the domain (e.g., objects and classes), but they usually provide only elementary mechanisms for this purpose, whereas they are much more sophisticated in the representation of the dynamic aspects.

Everything. This class is used for referring to languages that do not make any specific choice in the aspects of the domain to represent, and, therefore, may be in principle used for any kind of contexts and applications.

Figure 14: What to express?

When dealing with Ontologies it is important to understand what to express (see figure 14). Within the context of Ontologies Marit & Oddrun (1999) have developed a meta-model that defines conceptual building blocks of an information space. This meta-model takes knowledge as well as information sharing into account by letting Ontologies represent problem domains important to an enterprise. With the help from the meta-model, the management will be easier and solutions that provide knowledge sharing within an enterprise will be achieved (Marit & Oddrun, 1999). This is one reason why it is important to understand the meaning of using Ontologies within an enterprise.

IP 7: How to express?

Problem – When dealing with the representation of ontologies one problem is the difficult task of understanding how to express something.

Situation (context) – This criterion takes into account the basic formal nature of the ontology languages.

Solution – The focus on how to express something are considered by the following classes of languages (that we can use to express something within the context of ontologies):

Programming languages

Conceptual and semantic database models

Information system/software formalisms

Logic-based

Frame-based

Graph-based

XML-related

Visual languages

Temporal languages

Figure 15: How to express?

Exactly how to express something witin the context of Ontologies is considered as a problem (see figure 15). Top & Akkermans (1994) makes a distinction between several ontological viewpoints. They introduce three engineering Ontology views that have their own specific roles in the modelling task. These ontology views are: functional components, physical processes and mathematical constraints. By combining these different views we can use a particular approach to modelling that is called evolutionary modelling. Borst, Akkermans & Top (1997) mean that Ontologies can be broken up into small pieces with strong internal coherence but relatively loose coupling, thus reducing ontological commitments. Poli (2002) mean that within the context of ontologies three problems have to be addressed:

1. What are the boundaries of ontology?
2. What types are there of ontology?
3. What is the structure of ontology?

There also exist three main kinds of information:

1. Ontological
2. Quasi-ontological
3. Non-ontological

And three types of Ontologies:

1. Descriptive
2. Formal
3. Formalized

When dealing with Ontologies it is important to understand what type of ontology that we are dealing with (Poli, 2002).

IP 8: Interpret an expression

Problem – When dealing with the representation of ontologies one problem is the difficult task of understanding how to interpret an expression.

Situation (context) – This criterion takes into account the degree in which the various languages deal with the representation of incomplete information in the description of the domain.

Solution – A possible solution to this problem is to consider the following classes of languages:

Single model - Languages of this class represent a domain without any possibility of representing incomplete information. An ontology expressed in this language should be interpreted as an "exact" description of the domain, and not as a description of what we know about the domain. This means that the ontology should be interpreted in such a way that only one model of the corresponding logical theory is a good interpretation of the formal description.

Multiple models (or, several models) - Languages of this class represent a domain taking advantage of the possibility of representing incomplete information. An ontology expressed in this kind of language should be interpreted as specifying what we know about the domain and that the amount of knowledge we have about the domain can be limited.

Figure 16: Interpret an expression

Ontologies are an application-independent way to represent knowledge about the entities of an enterprise (see figure 16). In order for us to understand the context of ontologies the following aspects are important (WP8, page 69):

- How to build an ontology
- What tools are available to support the building process
- How to reach consensus about disputed elements in an ontology
- Which workflows are available to create ontologies

- How to maintain ontology versions and control their evolution
- How to learn ontologies automatically from sources
- What are the specificity of ontologies for enterprise application integration

4.2 Enterprise Modelling View

IP 9: Non-holistic view

<u>Problem</u> – By not seeing the whole enterprise as wholeness, the fundamental parts of the enterprise will not be understood and seen from a contextual point of view.

<u>Situation</u> (context) – Enterprise Modelling is the set of activities or processes used to develop the different parts of an enterprise model with a definite objective. The most important **benefit** of enterprise models is their capacity to add value to an enterprise, since they are able to make explicit facts and knowledge, which can be shared for users and different enterprise applications in order to improve enterprise performance.

Enterprise Modelling is not only focused on the generation of a single model that is able to represent all the enterprise, but in generating different types of models or views. Each of these views represents one of the enterprise dimensions: organisational, process, informational, resources, etc. at a different abstraction level. These models should be integrated so that the resultant model is something more than the sum of the individual views, and allow having a holistic and integrated vision of the enterprise that helps in the comprehension and improvement of its performance (WP7, page 14).

<u>Solution</u> – Enterprise Modelling is clearly a prerequisite of the enterprise integration, since things need to be modeled in order to be integrated. Without models it is almost impossible to understand the wholeness. Enterprise Modelling should include the following aspects: functional, informational, resources and organisational, and should make possible the representation of materials, information and control flows.

Figure 17: Non-holistic view

Using models could be a solution to the Non-holistic view (see figure 17). According to Van Gicgh (1991) modeling will not only provide a better understanding of the entire enterprise but the following aspects will also be achieved: simplification (understanding), formalization, optimization, solvability, generality and specifity (more details). Depending on these aspects it is considered important to model in an enterprise, especially to gain a better understanding of the involved enterprise. Kruchten (2002) mean that by using a standardised modeling language, like UML (Unified Modeling Language) the system developers can communicate and inform their decisions in a legible way. Therefore communication is considered as an important ingredient when creating models within an enterprise. A good communication among the involved stakeholders will most definetly lead to a better understanding of the whole enterprise.

IP 10: Communication failure

<u>Problem</u> – Multiple Enterprise Modelling Languages and tools that should model similar aspects of enterprises are not able to communicate among themselves.

Situation (context) – UEML (Unified Enterprise Modelling Language) is a thematic network founded by the European Commission. The project intends to resolve the problem derived from the existence of multiple Enterprise Modelling Languages and Tools.

Solution – Define a common language; the *Unified Enterprise Modelling Language* (UEML), that could be used as a standard of exchange among Enterprise Modelling tools (WP7, page 18).

Figure 18: Communication failure

The lack of integration between different modelling languages and tools are one reason for not being able to communicate (see figure 18). As mentioned in the background Lundell and Lings (2004) have introduced three tensions; the concept usage tension, the concept implementation tension and the product usage tension (see figure 3). Regarding communication failure it is considered relevant to understand the concept implementation tension. This means that the manufactures have addressed complexity in a CASE-environment (Computer Aided Software Engineering) and the Information system development process. In order to reduce this complexity they have created new tools in a limited way. The developers have adjusted the tools for importing and exporting features. This limitation means that the developers can focus on their own speciality without worrying of loosing market shares. The manufactures have to take important decisions about what the tool at hand should be able to support and what to not support (Lundell & Lings, 2004). It is considered important that the limitation of new tools created to support interoperability decrease among the manufacturers. Instead they should collaborate and get a better understanding of how different tools and languages should be in order to be able to integrate with each other.

IP 11: Model Integration

Problem – There exists great quantity of Enterprise Modeling Languages and they are overlapped. But the integration of the models generated with these languages is complicated, since tools do not exist to integrate models generated with different languages.

Situation (context) – Enterprise Modelling Languages provide constructs to describe and model the people roles, operational processes and functional contents, as well as support information and production and management technologies.

Solution – Achieve a common format, as UEML, which enables the exchange between different models and the establishment of an environment for reusing existing models.

Figure 19: Model Integration

Tools do not exist to integrate or translate models generated with different languages, this is concidered as the Model Integration problem (see figure 19). Models use heuristics that enable them to capture known properties of a real system while ignoring unknown. The presence of such incomplete or uncertain information embedded within each model that is then incorporated into a larger integrated system results in a semantic interoperability problem (Mackay et al., 1996). Commonly, model developers miss to explicitly define important context information which results in the loss of important information when the model is passed on to another user. Without proper context a model is easily misused. By providing inappropriate input data or by combining the model with another model also with poorly defined context a semantic integration problem results (Mackay, 1999). Linguistic terms can present semantics in a way that is meaningful to the end-user. This has the benefit of reducing the cognitive demands on a user who is trying to assemble environmental systems knowledge from existing disparate sources, whether they are in the form of current integrated models or in the form of information resources distributed across a global network. (Mackay, 1999)

IP 12: Lack of mechanisms

Problem – Mechanisms for the exchange of enterprise models do not exist. Tools do not exist in order to integrate its models with models carried out with other tools.

Situation (context) – Software Environments should support enterprise engineering and integration processes, implementing an Enterprise Engineering Methodology and supporting Enterprise Modelling Languages with the goal of analyzing, designing and using enterprise models.

Solution – We need tools that support several languages and methodologies and which can adapt new languages, contexts and syntax.

Figure 20: Lack of mechanisms

What is needed is a unified language devoted to the area of Enterprise Modelling (see figure 20). The language, named UEML (Unified Enterprise Modelling Language) is not intended to replace existing languages but is intended to provide a uniform interface to enterprise modelling tools and a neutral format for exchange of enterprise models. It therefore builds on previous languages and provides constructs to cover function/process, information, resource and organization/decision aspects of business entities. (Vernadat, 2002)

IP 13: Modelling standards

Problem – Enterprise Modelling Standards have had really little or null industrial impact due to they are associated with a specific platform or technology; which can not be achieved by a great number of enterprises. The problem concerned is how to deal with Enterprise Modelling Standards?

Situation (context) – Enterprise Modelling Standards are necessary for enterprise integration and interoperability, and there are a lot of them concerning Enterprise Modelling.

Solution – MDA (Model Driven Architecture) intends to promote the use of models as fundamental way for designing and implementing systems; and to separate the functional specification of a system from the details of its implementation in a specific platform.

Figure 21: Modelling standards

It is considered important to deal with Enterprise Modelling Standards (see figure 21). Chen and Vernadat (2004) have addressed the following categories regarding modelling standards; modelling and engineering (frameworks and languages), systems and sub-systems (shop floor, control systems and manufacturing data), IT services and infrastructures (model execution and integration, open distributed processing and others). A number of problems have also been identified. These problems concerns inconsistency between standards, inconsistency of terminology used in the different standards and lacking industrial applicability due to high level of abstraction of the standards. (Chen & Vernadat, 2004)

4.3 Architecure and Platform View

IP 14: Interoperability communication

Problem – Interoperability communication. How should communication between different platforms and components be able to understand and interact with each other.

Situation (context) – Common Object Request Broker Architecture (CORBA) is a standards-based distributed object computing infrastructure for object-oriented applications, which has a multi-language nature. The main goal of CORBA is to give a solution that addresses all the problems of interoperability. Using CORBA, two objects can interoperate whether they are on the different systems (for example, one can be running on a Windows machine and the other on a UNIX server on the other side of the world).

CORBA uses an Object Request Broker (ORB) to send requests from objects executing on one system to objects executing on another system. The ORB allows objects to interact in a heterogeneous, distributed environment, independent of the computer platforms on which the various objects reside and the languages used to implement them. For example, a C++ object running on one machine can communicate with an object on another machine which is implemented in COBOL or Java.

Solution – CORBA (Source: OMG) allows applications to communicate with one another without being aware of the hardware or software systems or the location of the application. A client can transparently invoke a method on a server object. The object can be on the same machine or on a remote machine on the network using the middleware, Object Request Broker (ORB). The ORB intercepts the call, finds an object that can implement the request, invokes its method passing the required parameters and returns the results to the client. Thus, ORB provides interoperability between applications on different machines in heterogeneous distributed environments and brings together multiple object systems (WP9, page 137).

Figure 22: Interoperability communication

Domínguez, Capilla & Cueva (2002) say that one of the main problems for developing software components is how they can interoperate in different and heterogeneous platforms or systems (see figure 22). The same authors mean that common platforms that make transparent the interoperability of components are necessary. Further, a platform is outlined (OVIBUS) that makes it possible for components to interoperate between different technologies (Dominguez et al., 2002).

Misikangas & Raatikainen (2000) have successfully shown that some of the incompatibility problems between different platforms can be solved with a special architecture in which all platform specific code is separated and used independently (broken down into elements). By using this proposed design technique it is possible to build agents that can migrate between different platforms. However, the agents can not use all features of the underlying platform (Misikangas & Raatikainen, 2000).

IP 15: ESA Introduction

Problem – Introduction of a new type of ESA (Enterprise Software Application**)** in the enterprise and connection to the existing one.

Situation (context) – Enterprise Modelling can allow enterprises to know and understand much better their business in order to make it more agile, to align their objectives with the market needs, and to improve their performance.

<u>Solution</u> – To answer this problem, we need to generate customized Enterprise Software Applications (ESA) starting from the model of the enterprise at the conceptual level, using what we can call a "generator of ESA".

The main sources are **UEML** and **IDEAS**, two European Projects founded by the European Commission related to Enterprise Modelling and interoperability Modelling carried out in **ATHENA** Project.

Figure 23: ESA Introduction

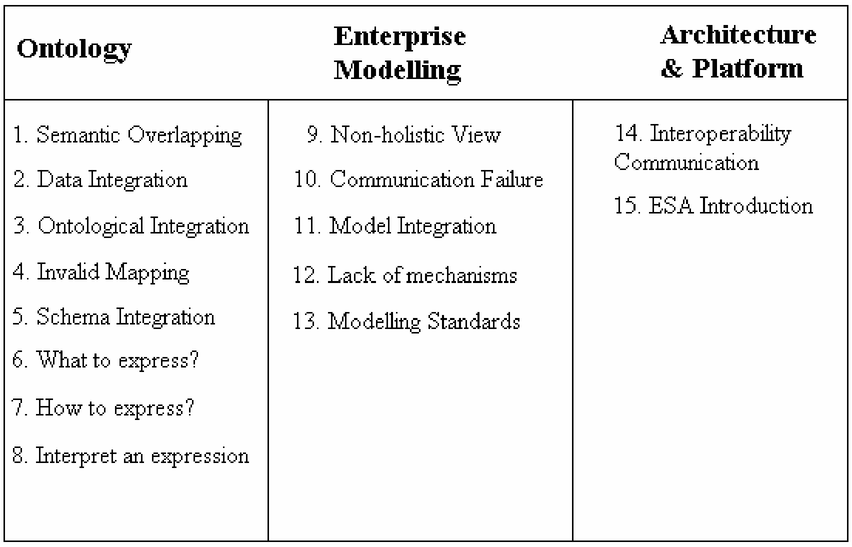
Introduction of new type of enterprise software applications in a coorporation is concidered problematic (see figure 23). Themistocleous, Irani and Kuljis (2004) say that Enterprise Application Integration (EAI) technologies support a shared architecture, where systems evolve and merge together. EAI software specifically addresses integration problems from a technical perspective. It combines a variety of integration technologies to build a centralised integration infrastructure. EAI addresses the need to integrate systems through insertion of functionality from different applications. Themistocleous et al. (2004) mean that the integration will lead to a significant cost reduction within an organisation. EAI will hopefully allow the flow of information, among different applications within disparate systems, to lead to the effective solving of client-server's scalability and interoperability problems (Themistocleous et al., 2004).

4.4 Summary of the results

As mentioned earlier Brinkkemper (1996) means that methods are being developed and employed over years, but a structure to take stock, generalize, and evaluate is actually needed.

The results elicited from this dissertation are one step towards an approach to structure methods into a method repository. The created patterns are supposed to support the creation of method chunks in a way so they can be completed, fully defined and stocked.

A created pattern is supposed to work as a descriptor within a method chunk. As an overview of the achieved results figure 24 summarizes the classified elicited interoperability problems mentioned above.



Ontology	Enterprise Modelling	Architecture & Platform
1. Semantic Overlapping	9. Non-holistic View	14. Interoperability Communication
2. Data Integration	10. Communication Failure	15. ESA Introduction
3. Ontological Integration	11. Model Integration	
4. Invalid Mapping	12. Lack of mechanisms	
5. Schema Integration	13. Modelling Standards	
6. What to express?		
7. How to express?		
8. Interpret an expression		

Figure 24: Summary of the elicited interoperability problems.

The identified interoperability problems have been categorized under three different views (Ontology, EM and A&P). Appendix B, C, D and E show how the elicited interoperability problems are connected as well as their origin. These connections are considered important in order to get an overview of all the interoperability problems and how they belong together. Together the identified interoperability problems may achieve solutions or approaches in order to solve bigger problems. These problems are main problems and have been identified by the INTEROP community.

5 Conclusion & Discussion

This chapter reflects on the process of achieving the objectives for this dissertation as well on the results. In addition, suggestions for future work are presented.

5.1 Conclusion

In order to reach the aim and objectives, literature studies and analyses of work packages have been performed. One problem was how to get an overview of the content from all available work packages and how to select the most important material. Every work package was connected to one and another in some way, for me it was important to understand these connections. By gaining knowledge of the content for every work package it became clearer how the work packages were connected to each other. By creating a meta-model of the first work package (appendix A) an overview of the relevant work packages for this dissertation could be represented. Meta-models were also created in order to show how different interoperability problems belong together.

The presented results (patterns) are of importance for scientists in the field (especially within the INTEROP) and for scientists studying interoperability on a general level. The problems have been analysed, elicited and categorized. This categorization makes it easier to understand what interoperability problems that exist under the three views; ontology view, Enterprise Modelling view and A&P view. These problems are important to understand in order to create a complete method chunk, thus allowing the creation of new project specific methods in the area of information system development. The elicited interoperability problems are of major concern not only in the INTEROP community, but also on a higher level of abstraction. This generalized view has been interpreted due to the second literature study performed in this dissertation.

5.2 Critical review of the results

The results from this thesis clearly indicate that there exist several interoperability problems. However, not all interoperability problems have a proposed solution. There have been difficulties to find a specific solution to some of the identified interoperability problems. There might not be solutions to these interoperability problems within the S-o-A articles, which is a reason for further analysis in future work.

To achieve the aim of this dissertation, six objectives were formulated.

With the help of a literature study the concepts of methods, method engineering, method chunk repository and interoperability have been explained, defined and characterized. With other words the following objectives was achieved:

1. Characterize the concept of methods and method engineering.
2. Characterize the concept of method chunk repository.
3. Give a fundamental understanding of interoperability.

With the help of the S-o-A material interoperability problems have been elicited and characterized into a set of patterns. One pattern includes one interoperability problem, the surrounding context and its proposed solution. The created set of patterns allowed the following objectives to be achieved:

4. Identify a set of interoperability problems from the S-o-A reports to be resolved.
5. Identify a set of solutions for the identified interoperability problems.
6. Categorize the identified interoperability problems and provide a structure.

By doing a second literature study based on the elicited interoperability problems, the results are compared to existing literature. This is considered a major strength of the results since every interoperability problem can be analysed and generalized on a more specific level than earlier. In other words the second literature study should provide a better understanding for the context of the relevant problem. However, my contribution to the academic field is only on a theoretical level and the patterns created among with their practical usefulness remains to be analysed in more detail.

Due to the amount of pages within the available work-packages, there might be a risk that not all interoperability problems have been elicited. Also, some interoperability problems might have been interpreted in a wrong way or the solutions might have been. The created meta-models (see appendix A, B, C, D and E) are one way of understanding where interoperability problems have been elicited and how they relate to each other.

One problem was the difficulty of choosing the right granularity for each interoperability problem. Some problems vary in abstraction level which makes it more difficult to understand the context and therefore propose a solution. The interoperability problems that have been compared to existing literature are considered to be on a satisfied abstraction level, as for the others an analysis was not available to do. One reason for this is the critical amount of time left for this dissertation to be completed. However, this is a good basis for future work which leads us to the next chapter.

5.3 Future work

Continue work on testing the theoretical surrounding in practice. Use the patterns created as a part of a method chunk.

Since method chunks have not yet been used in an enterprise context there did not exist any respondents for interviews. Interviews and case studies for investigating how enterprises use method chunks are considered important and necessary as future work.

There is clearly a need for empirical studies that evaluate the usefulness of a method repository in a real environment.

Not all interoperability problems have proposed solutions or context descriptions. A further analysis to elicit these solutions and descriptions are of importance for completeness of the created patterns. This work is relevant for the development of a method repository, where my created patterns can be used as descriptors.

References

Akkermans, H. & Top, J. (1994) *Tasks and Ontologies in Engineering Modelling.* International Journal of Human Computer Studies, Vol. 41, No. 4, pp. 585–617.

Akkermans, H., Borst, P., & Top, J. (1997) *Engineering Ontologies.* International Journal of Human Computer Studies, Vol. 46, No. 2, pp. 365–406.

Andersen, E. (1994) *Systemutveckling –principer, metoder och tekniker.* Studentlitteratur, Lund (in Swedish).

Andersen, E., Grude K.V. & Haug, T. (1994) *Målinriktad projektstyrning.* Tredje upplagan, Studentlitteratur, Lund (in Swedish).

Andersen, E. & Schwencke, E. (1998) *Projektarbete.* Studentlitteratur, Lund (in Swedish).

Arni-Bloch, N. (1999) *Towards a CAME Tools for Situational Method Engineering.* Available at Internet: http://interopesa05.unige.ch/INTEROP/Proceedings/Doctoral/PerPaper/III-1-ArniBloch.pdf [Collected 05.03.01].

Avison, D. & Fitzgerald, G. (1995) *Information System Development: Methodologies, Techniques and Tools.* Second edition, McGraw-Hill, London.

Avison, D. & Shah, H. (1997) *The information systems Development life cycle.* Glasgow: Bell och Bain Ltd.

Backlund, P, Söderström, E. & Wangler, B. (2004) *Four Interoperability Scenarios.* INTEROP documentation, working paper (draft) at the University of Skövde.

Bergvall, M. & Welander, T. (1996) *Affärsmässig systemförvaltning.* Studentlitteratur, Lund (in Swedish).

Beynon-Davies, P., Blonde, L., McPhee, D. & Jones, C. (1997) *A Collaborative Schema Integration System.* Computer Supported Cooperative Work, Vol. 6, No. 1, pp. 1-18.

References

Borchers, J. (2001) *A Pattern approach to interaction design.* Wiley, Chichester.

Brinkkemper, S. (1996) *Method engineering: engineering of information systems development methods and tools.* Journal of Information and Software Technology, Vol. 38, No. 4, pp. 275-280.

Brinkkemper, S. (2000). *Method Engineering with Web-enabled Methods*. State Of The Art and Research Themes on Information Systems Engineering, London, United Kingdom.

Brinkkemper, S., Saeki, M. & Harmsen, F. (1998). *Assembly Techniques for Method Engineering*. 10th Conference on Advanced Information Systems Engineering.

Chen, D. & Doumeingts, G. (2003) *European initiatives to develop interoperability of enterprise applications — basic concepts, framework and roadmap.*
Annual Reviews in Control Vol. 27, pp. 153–162.

Chen, D. & Vernadat, F. (2004) *Standards on enterprise integration and engineering – state of the art.* International Journal of CIM, Vol. 17, No. 3, pp. 235-253.

Cohen, W. (2000) *Data integration using similarity joins and a word-based information representation language.* ACM Transactions on Information Systems, Vol. 18, No. 3, pp. 288–321.

Crow, L. & Shadbolt, N. (2001) *Extracting Focused Knowledge from the Semantic Web.* International Journal of Human Computer Studies, Vol. 54, No. 1, pp. 155–184.

Dinesh, B. & Solomon, A. (2001) *Consulting Support During Conceptual Database Design in the presence of Redundancy in Requirements Specifications: An empirical study.* International Journal of Human-Computer Studies, Vol. 54, No. 1,

pp. 25–51.

Domínguez, F., Capilla, R., Cueva, J. (2002) *OVIBUS: A Scalable Platform for Combining Heterogeneous Components*. ICSR7 Workshop on Component-based Software Development Processes, Austin, Texas, April 15-19. Available at Internet: http://www.idt.mdh.se/CBprocesses/papers/13%20fdominguez-icsr7.pdf

[Collected 05.08.10].

Fitzgerald, B. (2000) *Systems Development Methodologies: The Problem of Tenses.* Information Technology & People, Vol. 13, No. 3, pp. 13-22.

References

Fitzgerald, B. (1997) *The Use of Systems Development Methodologies in Practice: A Field Study*. The Information Systems Journal, Vol. 7, No. 3, pp. 201-212.

Graziella, T. (1996) *Communication patterns and textual forms.* Oxford Intellect.

Iivari, J. (2000) *Information Systems Development as Knowledge Work: The body of systems development process knowledge.* Information Modelling and Knowledge Bases XI (Eds. Kawaguchi, E., Hamid, I. A., Jaakkola, H. and Kangassalo, H.) IOS Press.

Iivari, J. and Maansaari, J. (1998) *The Usage of development methods: are we stuck to old practices?* Information and Software Technology, *Vol.* 40, pp. 501-510.

INTEROP Network of Exellence IST – 508011. Available at Internet: www.interop-noe.org [Collected 05.01.10].

Jayaratna, N. (1994). *Understanding and Evaluating Methodologies*. McGraw-Hill Book Company, London.

Kruchten, P. (2002) *The Rational Unified Process An Introduction 2nd Ed.* Reading: Addison Wesley.

Langefors, B. (1995) *Essays on infology : Summing Up and Planning for the Future*. Studentlitteratur, Lund.

Lundell, B. and Lings, B. (2004) *Method in action and method in tool: a stakeholder perspective,* Journal of Information Technology, Vol. 19, No. 3, pp. 215 – 223.

Mackay, S. (1999) *Semantic Integration of Environmental Models for Application to Global Information Systems and Decision-Making.* Department of Forest Ecology & Management. Institute for Environmental Studies University of Wisconsin – Madison. SIGMOD Record, Vol. 28, No. 1, pp. 13-19.

Mackay, D.S., Gardels, K. Lopex, X., Foster, H. and Radke. J. (1996) *Semantic Integration of Environmental Models for Application to Global Information Systems and Decision-Making.* Interoperability of Geographic Information.
University Consortium on Geographical Information Science Research Priority, http://www.ncgis.ucsb.edu/other/ucgis/research_priorities/paper5.html

References

Marit, N. & Oddrun, O. (1999) *Modeling shared Information Spaces (SIS).* Group´99: International Conference on Supporting Group Work, pp. 199–208.

Misikangas, P. & Raatikainen, K. (2000) *Agent Migration Between Incompatible Platforms.* 20th IEEE International Conference on Distributed Computing Systems (ICDCS'00), Vol. 4, No. 4.

Moen, W. (2001) *Mapping the INTEROPERABILITY Landscape for Networked Information Retrieval.* JCDL´01 Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 50-51.

Pirlein, T. & Studer, R. (1995) *An environment for reusing Ontologies within a knowledge engineering approach.* The role of formal ontology in the information technology, Vol. 43, No. 5, pp. 945–965.

Noy, N., & Musen, M. (2003) *The PROMPT suite: interactive tools for ontology merging and mapping.* International Journal of Human-Computer Studies, Vol. 59, No. 6, pp. 983–1024.

Park, J. & Ram, S. (2004) *Information Systems Interoperability: What Lies Beneath?.* ACM Transactions on Information Systems, Vol. 22, No. 4, pp. 595–632.

Patel, R. & Davidson, B. (2003) *Forskningsmetodikens grunder.* Tredje upplagan, Studentlitteratur, Lund.

Poli, R. (2002) *Ontological methodology.* International Journal of Human-Computer Studies, Vol. 56, No. 6, pp. 639-664.

Ralyté, J. & Rolland, C. (2001). *An approach for Method Reengineering*. Proceedings of the 20th International Conference on Conceptual Modeling (ER2001), Yokohama, Japan, LNCS 2224, Springer 471 – 484.

Ralyté J. & Rolland, C. (2001). *An Assembly Process Model for Method Engineering*. 13th Conf. On Advanced Information Systems Engineering, CAISE'01 Interlaken, Switzerland, LNCS 2068, Springer-Verlag.

Ralyté J. (2004) *Towards Situational Methods for Information systems Development: Engineering Reusable Method Chunks*. Proceedings of the Thirteenth International Conference on Information Systems Development, Vilnius, Lithuania.

References

Stamper, R. (2000) *Understanding the roles of signs and norms in organisations – a semiotic approach to information systems design*.
Behaviour & Information Technology, Vol. 19, No. 1, pp. 15-27.

Stone, G. (2003) *The problem with software development.* Dunstanthomas Portsmouth. Available at Internet: http://consulting.dthomas.co.uk/ooad_articles_resources/Problem_with_Software_Development.pdf [Collected 05.03.08].

Themistocleous, M., Irani, Z. and Kuljis, J. (2004) *Extending the Information System Lifecycle through Enterprise Application Integration: A Case Study Experience.* Proceedings of the 37th Hawaii International Conference on System Sciences (HICSS´04), Vol. 8, No. 8, pp. 1-8.

Van Gigch, J. P. (1991) *System design, modeling and metamodeling*. Plenum press, New York.

Vernadat, F. (2002) *UEML: towards a unified enterprise modelling language.* International Journal of Production Research, Vol. 40, No. 17, pp. 4309-4321.

Welke, R. & Kumar, K. (1991). *Method engineering: a proposal for situation-specific methodology construction,* A research agenda, Cotterman and Senn(eds), Wiley.

Appendix A

WP 1

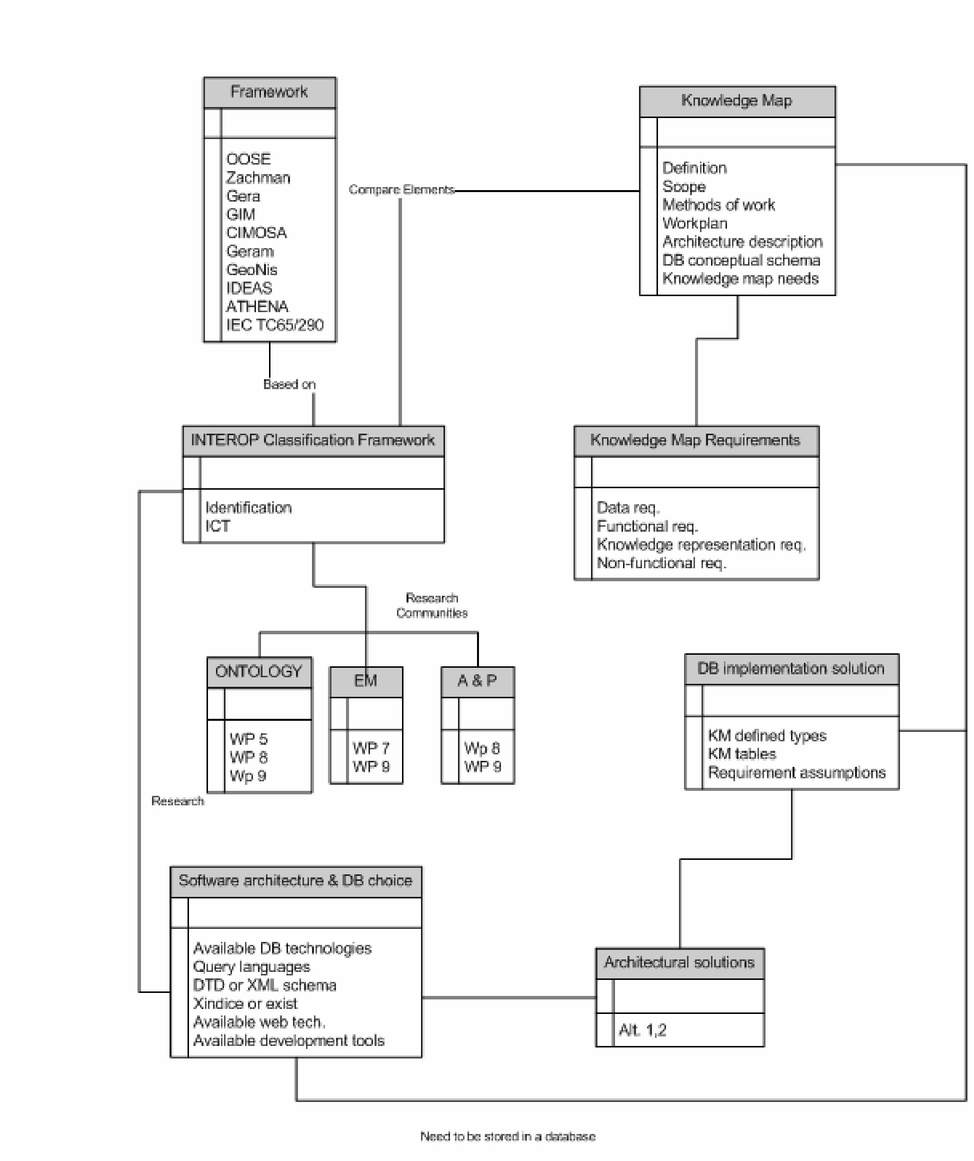


Figure A1: Overview of WP 1

Appendix B

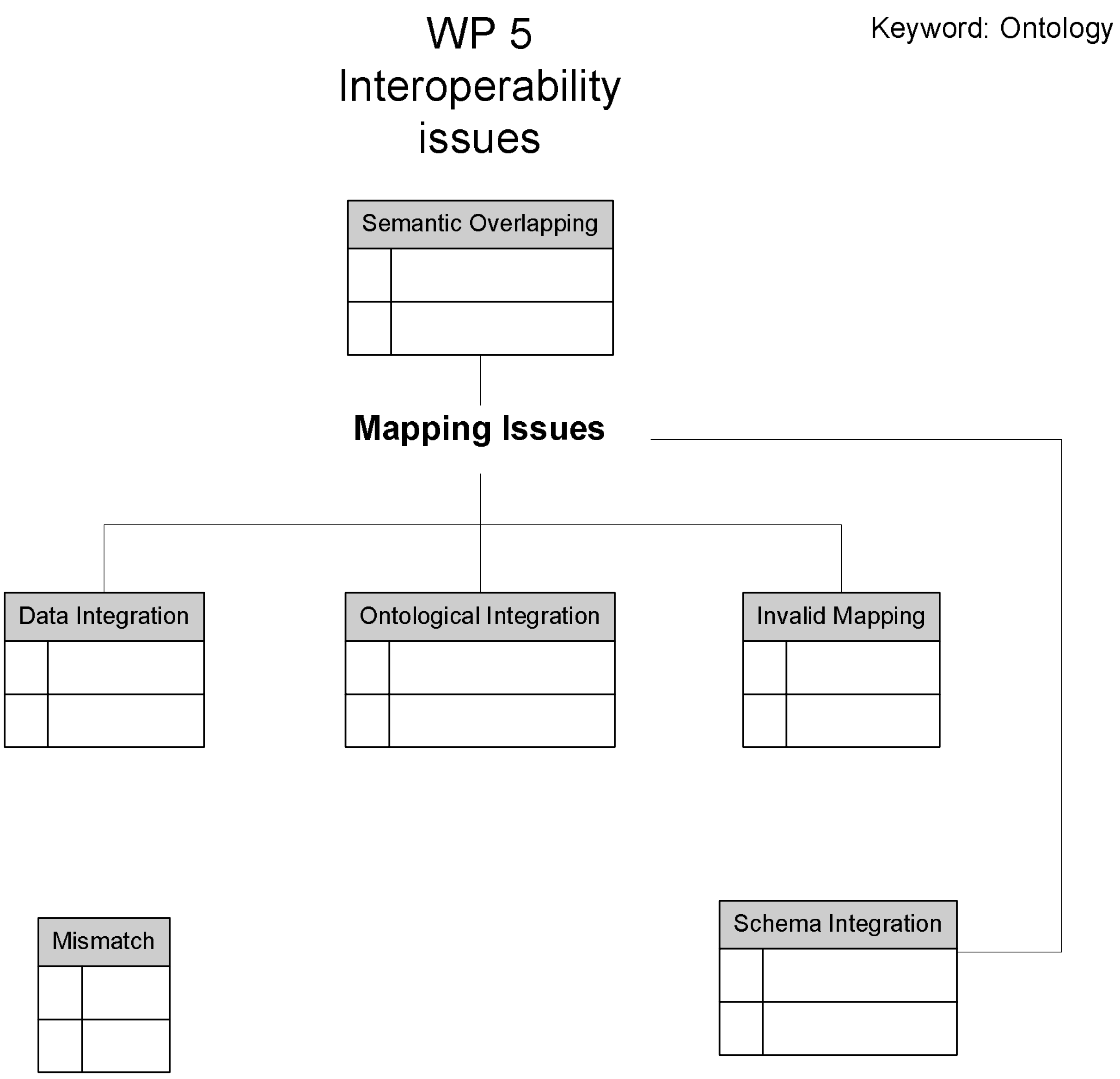
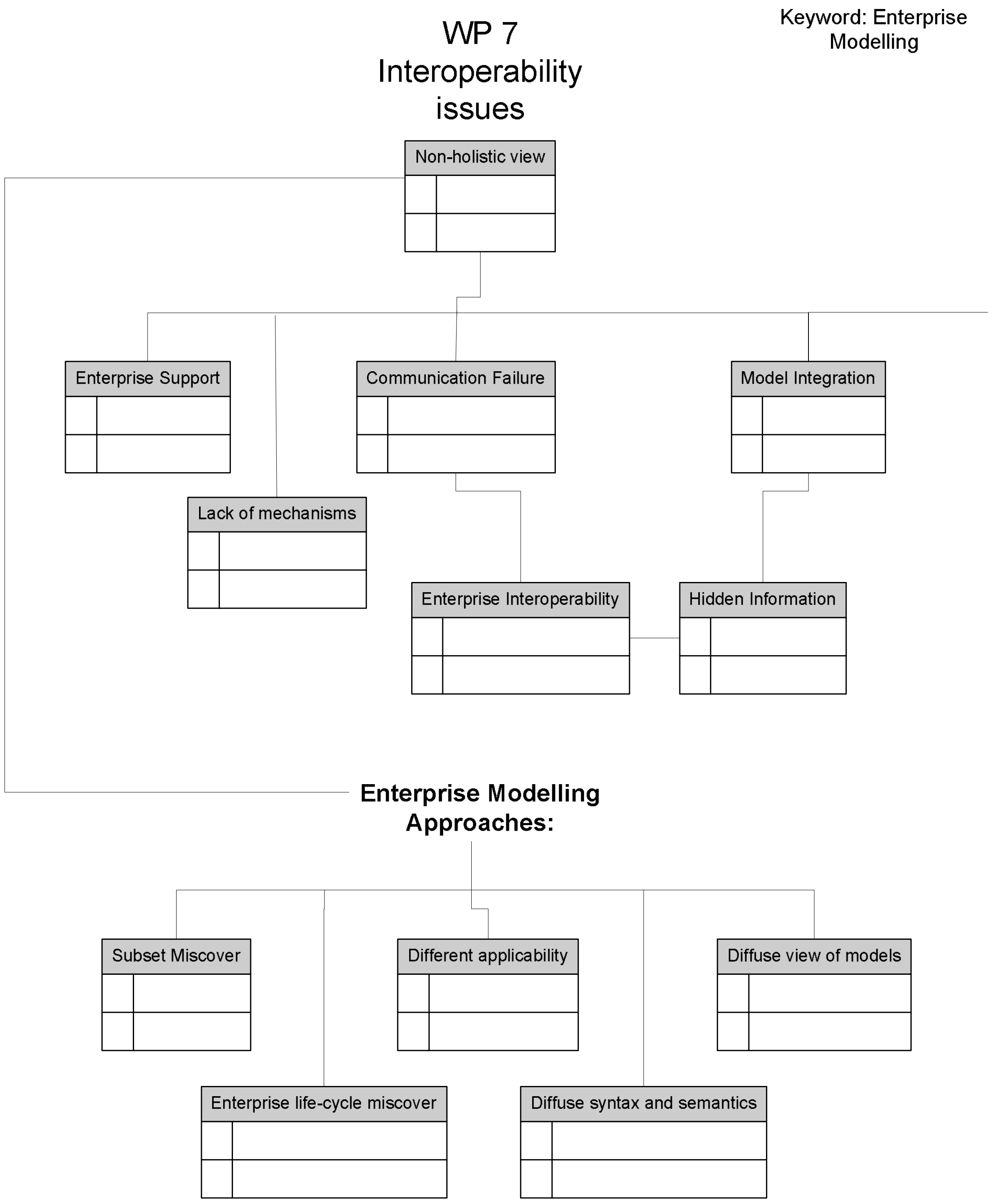


Figure B1: WP 5 Interoperability Problems

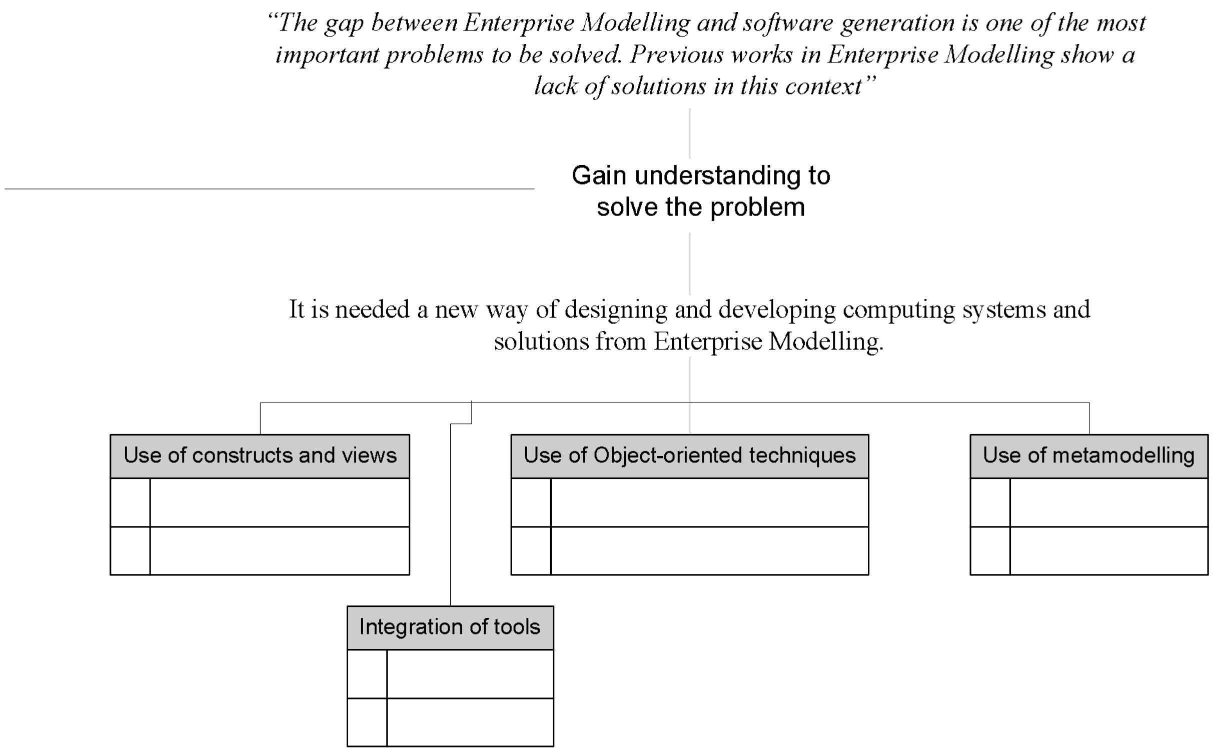
Appendix C



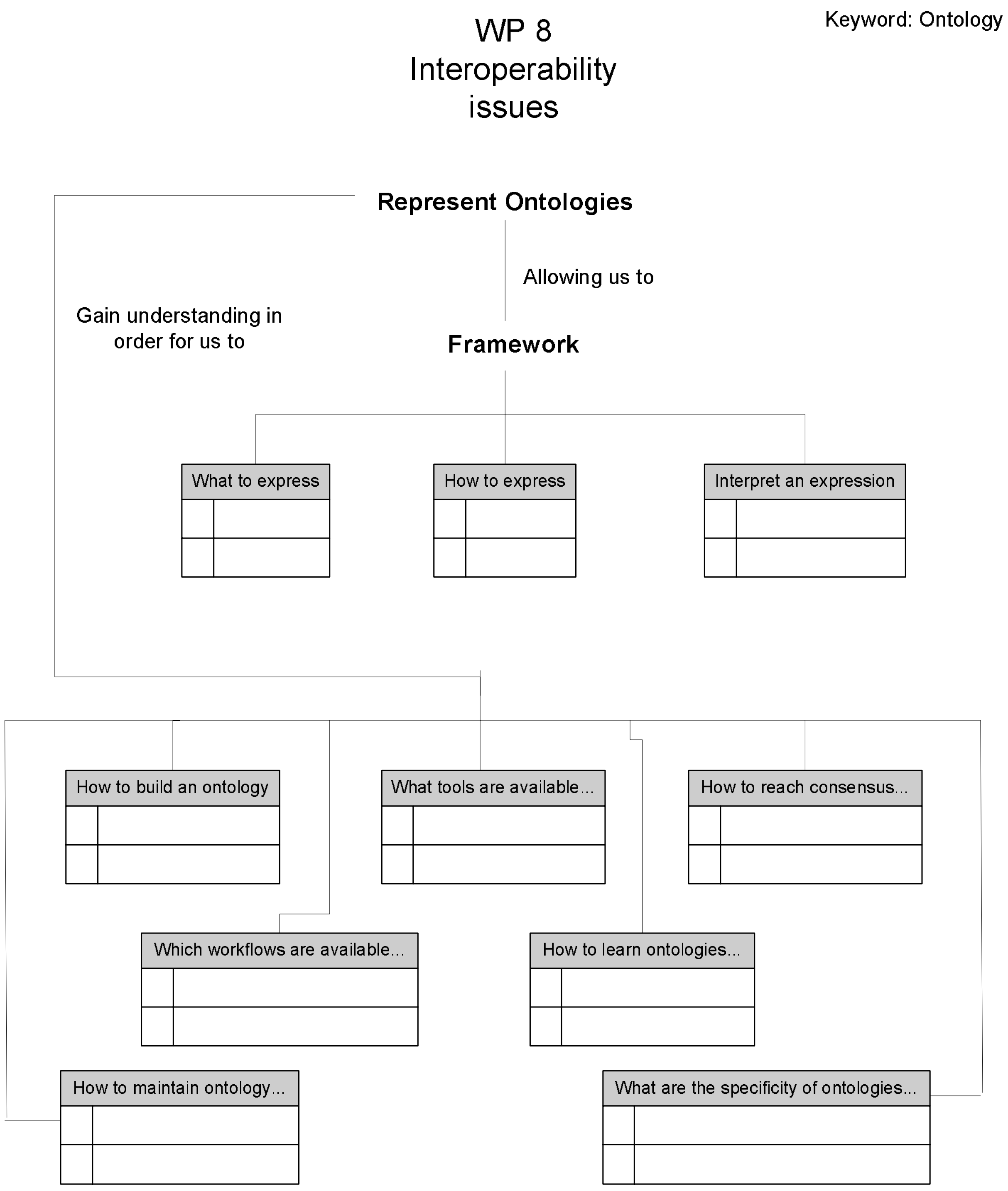
Keyword: Enterprise Modelling
WP 7 Interoperability issues
Non-holistic view
Enterprise Support
Lack of mechanisms
Communication Failure
Model Integration
Enterprise Interoperability
Hidden Information
Enterprise Modelling Approaches:
Subset Miscover
Different applicability
Diffuse view of models
Enterprise life-cycle miscover
Diffuse syntax and semantics

Figure C1: WP 7 Interoperability Problems

Appendix C cont.


"The gap between Enterprise Modelling and software generation is one of the most important problems to be solved. Previous works in Enterprise Modelling show a lack of solutions in this context"
Gain understanding to solve the problem
It is needed a new way of designing and developing computing systems and solutions from Enterprise Modelling.
Use of constructs and views
Use of Object-oriented techniques
Use of metamodelling
Integration of tools

Appendix D



WP 8 Interoperability issues
Keyword: Ontology
Represent Ontologies
Allowing us to
Gain understanding in order for us to
Framework
What to express
How to express
Interpret an expression
How to build an ontology
What tools are available...
How to reach consensus...
Which workflows are available...
How to learn ontologies...
How to maintain ontology...
What are the specificity of ontologies...

Figure D1: WP 8 Interoperability Problems

Appendix E

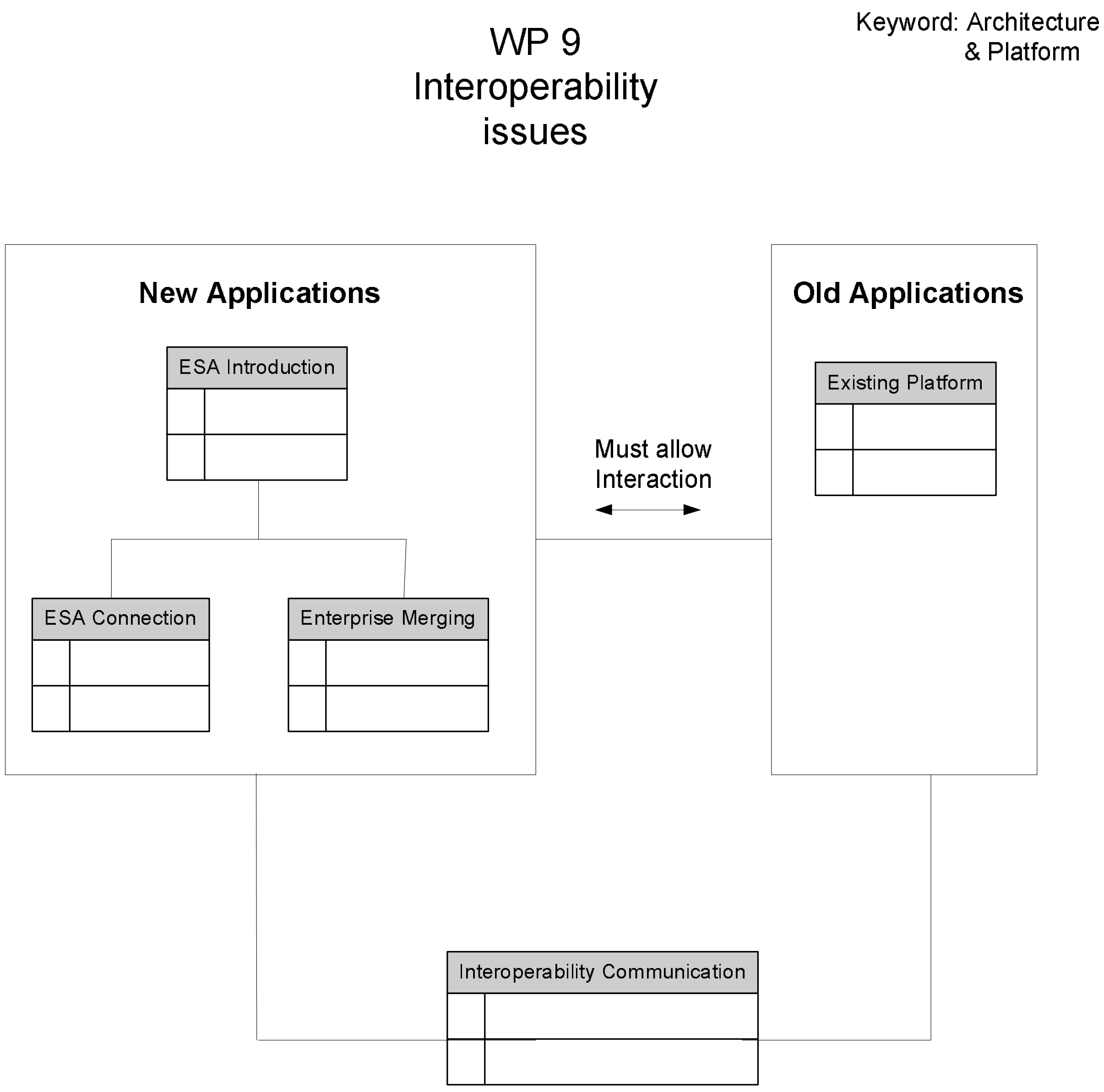


Figure E1: WP 9 Interoperability Problems

Appendix F

Incomplete Interoperability Problems

Ontology View

IP 16: Mismatch

Problem – It is sometimes difficult to understand that there exist a distinction between two different levels at which mismatches appear. It is a problem by not understanding this distinction, because mismatching can make information/data interpreted in the wrong way.

Situation (context) – A distinction is made between two levels at which mismatches appear. The first level is the language (meta-model) level. This is the level of the language that is used to specify ontology. Mismatches at this level are mismatches between the mechanisms to define classes, relations and so on.

The second level is the ontology or model level, at which the actual ontology of a domain lives. A mismatch at this level is a difference in the way the domain is modeled. The distinction between these two levels of differences is made very often (WP5, page 57).

Solution – There are no proposed solutions to the different mismatches.

Enterprise Modelling

IP 17: Enterprise support

Problem – Nowadays, a great quantity of Enterprise Modeling Languages exist, one major weakness is the support to enterprises in dynamic environments.

Situation (context) – For dynamic roles, cooperation in time and supporting of specific processes. Permanent changes in this kind of enterprises require a controlled way for managing the maturity of structures and processes. Enterprise Modeling Methodologies are not able of dealing with different levels of maturity. Besides, the Enterprise Modeling Languages are weak in easy and transparent externalization of dynamic roles and policies in extended enterprises.

Solution – No proposed solution…

IP 18: Subset miscover

Problem – Cover different subsets of different components of the enterprise engineering world.

IP 19: Enterprise life-cycle miscover

Problem – Cover different parts of the enterprise life-cycle dimension.

IP 20: Different applicability

Problem – Have different applicability according to the genericity dimension.

IP 21: Diffuse view of models

Problem – They address the model contents views with varying degrees.

IP 22: Diffuse syntax and semantics

Problem – In general, they provide similar concepts with distinct names, which are hard to compare because the languages have different syntaxes and semantics.

IP 23: Hidden information

Problem – Code generation can be hard if much important information (lack of abstraction) may be hidden or not represented, generating an inefficient code.

Situation (context) – The objective is to know what items of enterprises are modeled in order to better understand further transformation of them.

Solution – No proposed solution…

IP 24: Enterprise interoperability

Problem – There exist a great quantity of Enterprise Modelling Languages, Methodologies, Tools and Standards, which cover all enterprise dimensions. However, the main problem in this context is the interoperability among them.

Situation (context) –

<u>Solution</u> – It is needed a language that enables the enterprise model exchange among several Enterprise Modelling Tools. This is the research framework in UEML Project and A1-ATHENA Project.

Architecture and Platform View

IP 25: ESA connection

<u>Problem</u> – Connection of ESA between several enterprises.

<u>Situation</u> (context) –

Solution –

IP 26: Enterprise merging

<u>Problem</u> – Merge of enterprises and preparation of the merging by using enterprise modeling techniques.

<u>Situation</u> (context) –

Solution –

Appendix G

Content of the S-o-A material

www.interop-noe.org

Work Package 1

Knowledge map for Enterprise Modelling, Ontology, Architecture and Platform for interoperability researches inside INTEROP.

Network of Excellence - Contract no.: IST-508011
Title - Knowledge map of research in interoperability in the INTEROP NoE, 1st version.

Classification:	Public
Project Responsible:	University of Namur (UoN), Partner #5
Authors:	Michaël Petit (UoN)
Contributors:	John Krogstie (SINTEF), Guttorm Sindre (NTNU)
Task:	1.1 and 1.2
Status:	Version 1
Date:	11 August, 2004

Work Package 5

Common Enterprise Modelling Framework (CEMF).

Network of Excellence - Contract no.: IST-1-508011
Title - MAPPING REPRESENTATION LANGUAGES.

Document type:	Working Document
Authors:	Giuseppe Berio and Christian Tami
Companies:	UoT
Tasks:	5.5.4
Status:	Draft
Version:	1.0
Date:	31st July, 2004

Work Package 6

Design Principles for interoperability – Good practices and solutions.

Network of Excellence - Contract no.: IST-1-508011

Title - Design Principles for Interoperability.

Coordinator:	David Chen
Task:	6.3
Status:	Working document
Version:	4.0
Document no.:	WP6-004
Date:	10 November, 2004

Work Package 7

Method and specification to generate customised Enterprise Software.

Network of Excellence - Contract no.: IST-1-508011

Title - State of the art of methods to derive IT specifications from models and to develop IT specific components based on IT modelling.

Classification:	Public
Project Responsible:	GRAISOFT
Authors:	DOUMEINGTS
Contributors:	BERRE
Task:	T1;T2;T3;T4
Status:	Final Version
Date:	15 January, 2005

Work Package 8

Ontology-based integration of Enterprise Modelling and Architecture & Platforms.

Network of Excellence - Contract no.: IST-1-508011
Title - State of the art and state of the practice including initial possible research orientations.

Classification:	Confidential
Project Responsible:	UHP LORIA (4); CNR-IASI (11)
Main Authors:	CNR-IASI (11), UNIROMA–DIS (39), UNITILB (44), KTH (47)
Contributors:	WP8 Partners
Task:	8.1
Status:	Version 1.2
Date:	18 November, 2004

Work Package 9

Model driven and dynamic, federated enterprise interoperability architectures and interoperability for non-functional aspects.

Network of Excellence - Contract no.: IST-1-508011
Title - “State-of-the art for Interoperability architecture approaches” - Model driven and dynamic, federated enterprise interoperability architectures and interoperability for non-functional aspects.

Classification:	Public
Project Responsible:	SINTEF
Authors:	Arne-Jørgen Berre, Axel Hahn, David Akehurst, Jean Bezivin, Aphrodite Tsalgatidou, François Vermaut, Lea Kutvonen, Peter F. Linington
Task:	9.1
Status:	Version 1.0
Date:	November 19, 2004