# International Workshop on Enterprise Interoperability (IWEI 2008)

Marten van Sinderen, Pontus Johnson, and Lea Kutvonen (Eds.)

Munich, Germany, September 18, 2008
http://www.ics.kth.se/iwei/

## Proceedings

In conjunction with:

**The 12th IEEE International EDOC Conference**
**The Enterprise Computing Conference (EDOC 2008)**
**15-19 September 2008, Munich, Germany**
**http://www.edocconference.org**

## Editors

**Marten van Sinderen**
Centre for Telematics and Information Technology
University of Twente
PO Box 217
7500 AE Enschede, the Netherlands
m.j.vansinderen@ewi.utwente.nl
http://wwwhome.ewi.utwente.nl/~sinderen/

**Pontus Johnson**
School of Electrical Engineering
Industrial Information and Control Systems
Royal Institute of Technology (KTH)
Osquldas väg 12, 7tr
SE-100 44 Stockholm
pontus@ics.kth.se
http://www.ee.kth.se/php/index.php?action=people&cmd=extended&peopleid=403

**Lea Kutvonen**
Department of Computer Science
University of Helsinki
PO Box 68
FIN-00014 Helsinki, Finland
Lea.Kutvonen@cs.Helsinki.FI
http://www.cs.helsinki.fi/Lea.Kutvonen/

Enschede, The Netherlands, September 2008
CTIT Workshop Proceedings Series WP08-05
ISSN 1574-0846

# Table of Contents

# List of Authors

## Program Committee

| | |
|---|---|
| Scott Bernard | Carnegie Mellon University, Syracuse University, USA |
| David Chen | Université Bordeaux 1, France |
| Paul Davidsson | Blekinge Institute of Technology, Sweden |
| Guy Doumeingts | INTEROP-VLab/GFI, France |
| Yves Ducq | Université Bordeaux 1, France |
| Thomas Fischer | University of Stuttgart and the Otto Beisheim School of Management, Germany |
| Ricardo Goncalves | New University of Lisbon, UNINOVA, Portugal |
| Leonid Kalinichenko | Russian Academy of Sciences, Russian Federation |
| Stephan Kassel | University of Applied Sciences Zwickau, Germany |
| Kurt Kosanke | CIMOSA Association, Germany |
| Marc Lankhorst | Telematica Instituut, The Netherlands |
| Kai Mertins | Fraunhofer IPK, Germany |
| Raul Poler | Polytechnic University of Valencia, Spain |
| Dick Quartel | Telematica Instituut, The Netherlands |
| Sven-Volker Rehm | University of Stuttgart and the Otto Beisheim School of Management, Germany |
| Joachim Schelp | University of St. Gallen, Switzerland |
| Pierre-Yves Schobbens | University of Namur, Belgium |
| Marten Schönherr | Technische Universität Berlin, Germany |
| Markus Strohmaier | Graz University of Technology, Austria |
| Bruno Vallespir | Université Bordeaux 1, France |
| Alain Wegmann | Ecole Polytechnique Federal de Lausanne, Switzerland |
| Xiaofei Xu | Harbin Institute of Technology, China |

## Supporting Organizations and Projects

IFIP TC5 WG5.2


InterOP - VLab


IEEE Computer Society


CTIT - Centre for Telematics and Information Technology


KTH Royal Institute of Technology


Freeband A-MUSE project - Architectural Modeling Utility for Service Enabling

# Preface

One of the trends in the global market is the increasing collaboration among enterprises. Constant changes in inter- and intra-organizational environment will persist in the future. Organizations have to flexibly and continuously react to (imminent) changes in markets and trading partners. Large companies but also SMEs have to cope with internal changes from both a technical (e.g. new information, communication, software and hardware technologies) and an organizational point of view (e.g. merging, re-organization, virtual organizations, etc.). In this context, the competitiveness of an enterprise depends not only on its internal performance to produce products and services but also on its ability to seamlessly interoperate with other enterprises. External and internal collaborative work needs more interoperable solutions.

The International Workshop on Enterprise Interoperability, IWEI, aims at identifying and discussing challenges and solutions with respect to enterprise interoperability, both at the business and the technical level. The workshop promotes the development of a scientific foundation for specifying, analyzing and validating interoperability solutions; an architectural framework for addressing interoperability problems from different viewpoints and at different levels of abstraction; a maturity model to evaluate and rank interoperability solutions with respect to distinguished quality criteria; and a working set of practical solutions and tools that can be applied to interoperability problems to date.

IWEI organized by the IFIP Working Group 5.2 on Enterprise Interoperability. The aim of IFIP WG5.2 is to progress and disseminate research and development results in the area of enterprise interoperability. The IWEI workshop is therefore also a platform where ideas emerged from IFIP WG5.2 meetings can be discussed, or reversely, where issues raised at the workshop can be taken to the IFIP community for further contemplation and investigation.

This volume contains the proceedings of the first edition of the workshop, IWEI 2008, held on September 18, 2008, in Munich, Germany, in conjunction with the 12th IEEE International EDOC Conference – The Enterprise Computing Conference (EDOC 2008). Nine papers were selected for oral presentation and publication, based on a thorough review process, in which each paper was reviewed by several experts in the field. The papers are representative for the current research activities in the area of enterprise interoperability. For convenience, the papers were grouped in 4 sessions, reflecting some of the major topics related to enterprise interoperability, namely Session 1 - Ontologies and the semantic web; Session 2 - Service-orientation; Session 3 - Inter-organizational interoperability; and Session 4 - Maturity models.

We would like to take this opportunity to express our gratitude to all people who contributed to the IWEI 2008 workshop. We thank the authors for submitting content, which resulted in valuable information exchange and stimulating discussions; we thank the reviewers for providing useful feedback to the submitted content, which undoubtedly helped the authors to improve their work; and we thank the attendants for expressing interest in the content and initiating relevant discussions. We are indebted to IFIP TC5 for recognizing the importance of enterprise interoperability as a research area with high economic impact, and acting accordingly with the establishment of WG5.2. Finally, we appreciate the possibility to have 3M4EC being held in conjunction with the EDOC 2008 conference, and we are grateful for the support we received from the EDOC 2008 organization.

Munich, Germany, September 2008

Marten van Sinderen, Pontus Johnson, Lea Kutvonen
IWEI Organizers

# A Framework for Interoperability Analysis on the Semantic Web using Architecture Models

Johan Ullberg, Robert Lagerström, Mathias Ekstedt
*Department of Industrial Information and Control Systems*
*Royal Institute of Technology (KTH)*
*{johanu, robertl, mathiase}@ics.kth.se*

## Abstract

*IT decision making requires analysis of possible future scenarios. The quality of the decisions can be enhanced by the use of architecture models that increase the understanding of the components of the system scenario. It is desirable that the created models support the needed analysis effectively since creation of architecture models often is a demanding and time consuming task. This paper suggests a framework for assessing interoperability on the systems communicating over the semantic web as well as a metamodel suitable for this assessment. Extended influence diagrams are used in the framework to capture the relations between various interoperability factors and enable aggregation of these into a holistic interoperability measure. The paper is concluded with an example using the framework and metamodel to create models and perform interoperability analysis.*

## 1. Introduction

The hopes on the semantic web as a solution to many of the problems with information systems interoperability are currently growing. Foremost are these hopes related to a future where information systems will start interacting in a more autonomous and intelligent way without humans first specifying the interaction in detail. However, the semantic web is not an unambiguous product that is ready to use off-the-shelf. Rather, the future users of the semantic web face a number of design decisions that they need to consider when integrating their information systems with this mechanism.

Architecture modeling is today a state of the art approach to information systems development and management. Essentially the main idea is that the architecture models should predict the behavior of the information system, acting in its environment, before the information system is developed and is being taken into operation. The architecture models allow reasoning about the consequences of various potential scenarios and thereby support decision making. Using models also increases the understanding of complex systems and enables reuse of information.

In order to predict which architecture scenario is preferable, three things are needed. Firstly, models over the scenarios need to be created. Secondly, it is necessary to define what is desirable; the goal. In this article the goal is to achieve high information system service interoperability. Thirdly, we need to understand the causal chains from the scenario choice to the goal. Suppose that scenario A features services described in a semantic web language using an ontology that is very suitable for expressing the current service as such, but the language and the ontology is not widely spread. In scenario B on the other hand, the service is described in a semantic web language that is wide spread but has several unambiguous interpretations. To decide which scenario is preferable is often difficult, particularly without a formal analysis.

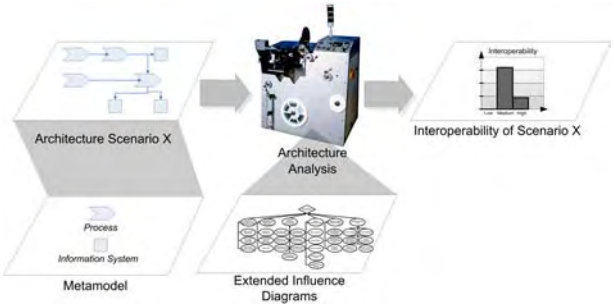



**Figure 1. The relation between metamodels, architecture scenarios, analysis, formal specification of analysis, and the result of the analysis**

In order to perform this kind of analysis, the architecture models firstly need to contain the proper information. In the above example, where the decision maker is interested in service interoperability over the semantic web, the models need to contain information regarding which languages and ontologies that are used for describing the service, how expressive those languages and ontologies are, how easy it is to identify

the service descriptions, etc. The kind of information contained in a model is given by its metamodel, so it is important that architecture metamodels are properly designed.

In order to determine if a metamodel is amenable to the analysis of a certain quality attribute, such as interoperability, it would be helpful with a structured account of that analysis. Which of all the modeled aspects are most important and which aspects are depending on each other and how? We will use a notation called Extended Influence Diagrams (EID) [1] in order to formalize the analysis of interoperability.

Figure 1 depicts the relation between an architecture scenario, modeled using a metamodel, the analysis of the scenario, the formal specification of the analysis through an extended influence diagram and finally the output: the interoperability level of the analyzed scenario.

The main contribution of this paper is an extended influence diagram and a metamodel that, through the creation of architecture models, supports interoperability assessments of information system services using the semantic web.

The remainder of this paper is delineated as follows; section 2 introduces some concepts of the semantic web. Extended influence diagrams are introduced in section 3. Section 4 presents the framework for semantic web service interoperability analysis in the form of an extended influence diagram. Section 5 evaluates the usefulness of a number of common architecture metamodels. Section 6 proceeds to detail the content of the metamodel that supports the interoperability analysis. The applicability of the metamodel is demonstrated in the subsequent section 7. Finally, section 8 concludes the paper.

## 2. Semantic Web Concepts

This section briefly presents the different concepts related to the semantic web. These concepts are later introduced as entities for the semantic web metamodel, see Figure 5.

The overall use case for the semantic web is that *Information Providers* publish information and the *Agents* task is to find the information that is correct with respect to the agents *Goal*. In other words the information provider and the agent does in the successful case perform what we here label *Stakeholder Collaboration.* The information provider publishes semantic information about *Things* in the real world in a *Formal Denotational Description*, i.e. the semantic part of a web webpage. This might e.g. be goods for sale or services provided. The agent has the goals encoded in a *Requirement Description.* We thus here differentiate the real-world objects and phenomena Things and Goals from those belonging to the technical semantic web solution. Both the requirement description and the formal denotational description are written in a *Semantic Language* such as RDF [2].

Semantic descriptions use *Ontologies*, explicit specifications of a conceptualization [3], to define the meaning of terms in their descriptions. Well known ontologies available today include for instance Dublin Core [4] for documents, SUMO[5] as an upper ontology describing more abstract concepts and numerous domain specific ontologies such as for instance the Gene Ontology project [6]. These ontologies are also written in a semantic language, e.g. DAML+OIL [7] and OWL [8], where nowadays perhaps OWL is the most well known and endorsed by W3C. In order to relate different ontologies to each other *Ontology Gateways* are used for this mapping alignment or merging. [9][10]

In the search for relevant information the agent can use *Information Retrieval Applications*, i.e. search engines that crawls the web and index the content it find. Swoogle [11] is one example of such an application. The information retrieval applications will be queried using a *query language* such as Corese [36], RQL [12] or SquishQL [13].

## 3. Bayesian Networks and Extended Influence Diagrams

Friedman describes a Bayesian network, $B=(G, P)$, as a representation of a joint probability distribution, where $G=(V, E)$ is a directed acyclic graph consisting of vertices, $V$, and edges, $E$ [14]. The vertices denote a domain of random variables $X_1,\ldots, X_n$, also denoted chance nodes. Each chance node, $X_i$, may take on a value $x_i$ from the finite domain $Val(X_i)$. The edges denote causal dependencies between the nodes, i.e. how the nodes relate to each other. The second component, $P$, of the network $B$, describes a conditional probability distribution for each chance node, $P(X_i)$, given its parents $Pa(X_i)$ in $G$. It is possible to write the joint probability distribution of the domain $X_1,\ldots, X_n$ using the chain rule of probability, in the product form:

$$P(X_1,...,X_n) = \prod_{i=1}^{n} P(X_i \mid Pa(X_i))$$

In order to specify the joint distribution, the respective conditional probabilities that appear in the product form must be found. The second component $P$ describes distributions for each possible value $x_i$ of $X_i$, and $pa(X_i)$ of $Pa(X_i)$, where $pa(X_i)$ is the set of values of $Pa(x_i)$. These conditional probabilities are represented in matrices, here on called conditional probability matrices (CPMs).

If the probabilities of the source variables are known, it is possible to infer a value for the target variable using the law of total probability,

$$P(X_1) = \sum_i P(X_1 | Pa(X_i)) P(Pa(X_i)) \cdot$$

Also, using Bayes' rule,

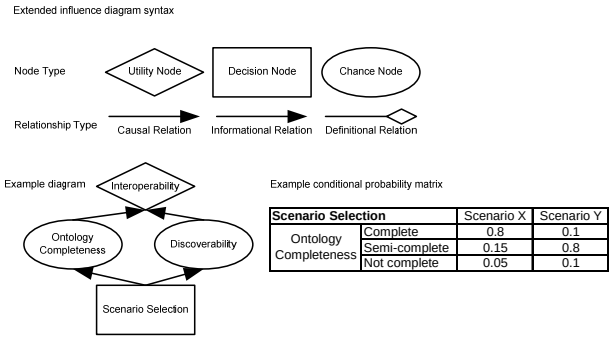
$$P(X_1 | Pa(X_1)) = \frac{P(Pa(x_1) | X_1) P(X_1)}{P(Pa(x_1))}$$

makes it possible to calculate the values of source variables based on the probabilities of a target variables.

Extended influence diagrams are an extension of influence diagrams [15][16], which in turn are an enhancement of the above mentioned Bayesian networks [17][18]. Thus, extended influence diagrams support probabilistic inference in the same manner as Bayesian networks do; given the value of one node, the values of related nodes can be calculated. The different relations that can be used in an extended influence diagram are either causal, as in Bayesian networks, informational, or definitional. In extended influence diagrams there are three different types of nodes; decision nodes, utility nodes, and like in Bayesian networks chance nodes. Decision nodes represent the decisions that can be made, e.g. selecting between different architecture scenarios. Utility nodes represent the goals, e.g. semantic web interoperability. Chance nodes could typically be ontology completeness or discoverability. The syntax for the graphical representation different relations and nodes is presented in Figure 2.

An example extended influence diagram with a conditional probability matrix is presented in Figure 2 below the extended influence diagram syntax. In this example the probability matrix represents the probabilities of the attribute ontology completeness to be complete, semi-complete, or not complete if a scenario x or a Scenario Y is selected. As can be seen in the figure the ontology completeness would be semi-complete with a probability 0.15 (15 %) if scenario x is selected.

For more comprehensive treatments on influence diagrams and extended influence diagrams see [1], [15], [16], [17], [18], [19], [20] and [21]. The extended influence diagrams here serve the purpose of formally specifying architecture analyses.




| Scenario Selection | | Scenario X | Scenario Y |
|---|---|---|---|
| Ontology Completeness | Complete | 0.8 | 0.1 |
| | Semi-complete | 0.15 | 0.8 |
| | Not complete | 0.05 | 0.1 |

**Figure 2. An extended influence diagram and a simple example. With a chosen scenario in the decision node, the chance nodes will assume different values, thereby influencing the utility node [22].**

## 4. A Framework for Interoperability Analysis on the Semantic Web

This section presents an extended influence diagram that captures theory regarding interoperability from the field of the semantic web. The design of the extended influence diagram is mainly influenced by [23][24][25][26][27][28][29][30].

Several methods for assessing interoperability on a general scope have previously been suggested. The assessment methods include LISI[31], SoSI[32], LCIM[33] and i-Score[34]. Few of these are focused on applying mathematical models and not all have a numerical measurement of interoperability [34]. The work presented in this paper incorporates many of the proposed interoperability measures and uses the mathematics of Bayesian networks as a means of aggregation.

Interoperability is the ability of two or more systems or components to exchange information and to use that information [35]. Adopted to the domain of the semantic web, *Semantic Web Interoperability* is defined as the probability for successful retrieval of information on the semantic web. Semantic web interoperability is influenced by five concepts: firstly, *Transmission protocol compatibility*, meaning that the transmission protocols of the agent and the information provider are compatible (generally http); secondly, *Discoverability*, meaning how difficult it is to find the appropriate information provider; thirdly, *Ontology completeness*, i.e. that the Ontologies have enough coverage and expressiveness; fourthly, *Quality of formal denotational description markup*, concerned with the markup of the provided information that is to be found; finally, *Quality of requirement description markup*, meaning that the

specification of what is sought after also is marked up in a sufficient way, cf. Figure 3.

### 4.1 Discoverability

Discoverability in the semantic web is mainly concerned with two tasks [11]: firstly, finding appropriate ontologies while performing markup, *Ontology discoverability*, and secondly, finding the instance data on pages containing semantic markup, captured by the concept *Quality of information retrieval application*.

The quality of the information retrieval application is dependent on its ability to perform indexing, *Quality of indexing*, and the *Semantic expressiveness of query language*. Query languages such as Corese[36], RQL[12] and SquishQL[13] have different expressiveness in terms of formally specifying the sought after information.

### 4.2 Ontology Completeness

The *Ontology completeness* is concerned with coverage of the ontologies, that they cover all aspects needed to create a rich formal semantic description of the objects in the real world as well as the objectives for someone searching for information. The general ontology completeness is defined in terms of the completeness of the ontologies related to the requirement description, the formal denotational description and possible ontology gateways respectively.

The *Requirement description's ontology completeness with respect to real world goal* is the matter of ensuring that the ontology that the user, seeking information, applies when creating a requirement description for the agent is complete with respect to the goal the user has. In the same manner someone publishing information relate to one or many ontologies that also have to be complete with respect to the thing in the real world that is to be described. This concept is named *Formal denotational description's completeness with respect to real-world entity* in the extended influence diagram. Generally, the ontologies of the requirement description do not match those used in the formal denotational description. Ontology gateways can mitigate this problem by relating ontologies to each other. It is however of importance that the gateways are complete with respect to the requirement description, *Ontology gateway completeness with respect to requirement description.*

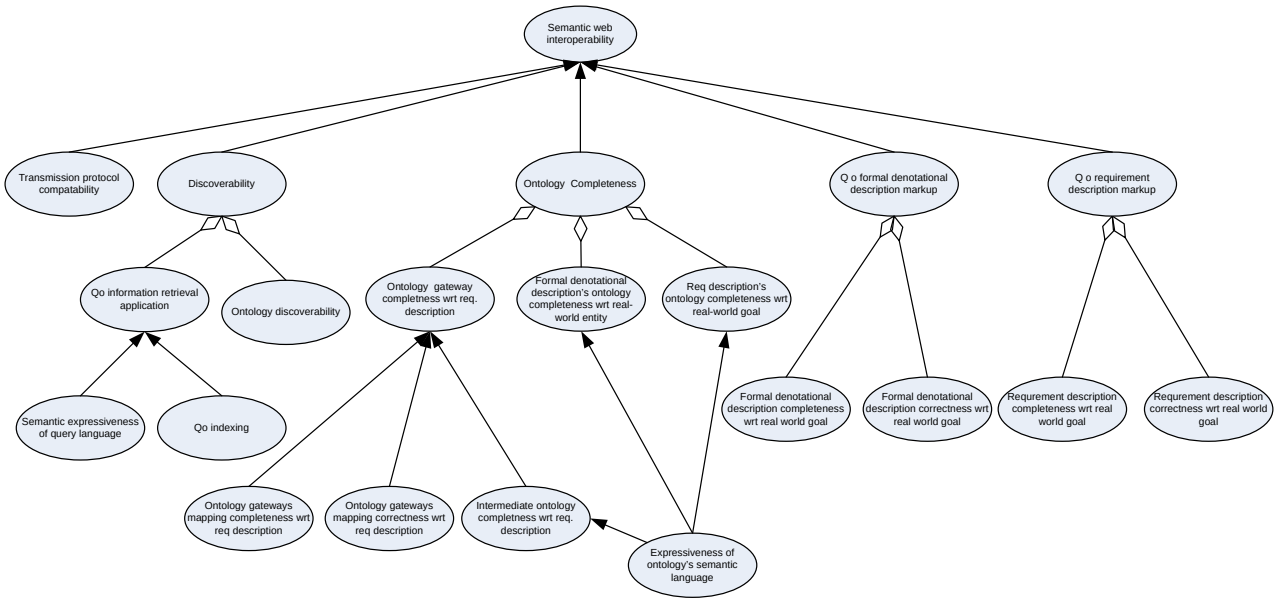



**Figure 3. The extended influence diagram containing factors influencing semantic web interoperability and thereby of interest when performing analysis of such**

The *Expressiveness of ontology's semantic language* influences the ontology completeness, so that all wanted relations between concepts can be expressed in the language. The ontology gateway completeness is also dependent on: the *Ontology gateway's mapping completeness with respect to requirement description*, i.e. that all concepts in the requirement description are mapped in the gateway; the *Ontology gateway's mapping correctness with respect to requirement description*, i.e. that all mapped concepts are correctly mapped; and finally the *Intermediary ontology completeness with respect to requirement description*. Sometimes the ontologies aren't mapped directly to each other but rather through one or more intermediary ontologies. If so it's of importance that these ontologies are complete in coverage of the important concepts.

### 4.3 Quality of Formal Denotational Description Markup

The basis for the semantic web is that information on the web is marked up in a formal denotational description. The quality of this markup is thus of importance to the interoperability. This quality is defined in terms of *Formal denotational description completeness with respect to real world goal*, the all (relevant) information is semantically marked up, and *Formal denotational description correctness with respect to real world goal*, that the markup is correctly performed.

### 4.4 Quality of Requirement Specification Markup

The requirement description, the coding of the goal the agent tries to fulfill, should also be marked up semantically in order to achieve interoperability. As in the case with the formal denotational description this quality is defined as *requirement description completeness with respect to real world goal*, the all information is semantically marked up, and *requirement description correctness with respect to real world goal*, that the markup is correctly performed.

## 5. Architecture Frameworks for Analysis

With the requirement on architecture models to support architecture analysis follows a specific requirement on architecture metamodels. Specifically, all entities and attributes that are required for a complete analysis as specified in an extended influence diagram must be found in the architecture metamodel, in order for the corresponding model to be amenable to analysis. See Figure 4.

There exist many architecture modeling frameworks and languages. The number one software system modeling language is UML [37]. UML provides a metamodel divided into a number of diagram types that can be used for system design and analysis. There also exist extensions to UML such as SysML [38] which adopts hardware aspects of systems to the models. There also exist a substantial number of enterprise architecture frameworks that also takes business and the usage of systems into account in the models. Two enterprise architecture frameworks that are explicitly focused on metamodels are the Department of Defense Architecture Framework (DoDAF) [39] and Archimate [40] Finally there the framework specifically for interoperability such as ATHENA Interoperability Framework (AIF) [41], Levels of Systems Interoperability (LISI) [31] and the European Interoperability Framework (EIF) [41].

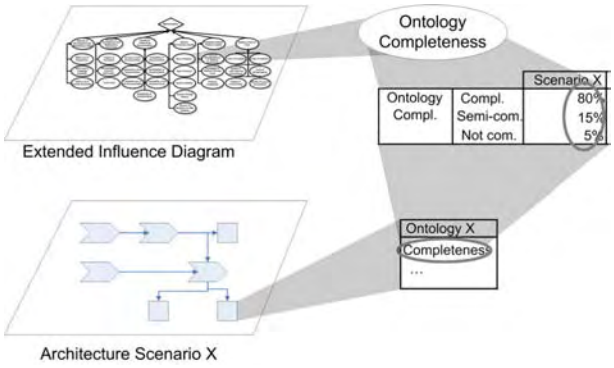



**Figure 4. The properties found in an extended influence diagram determine what entities and attributes should be present in an architecture metamodel.**

When considering the suitability of the metamodels related to these frameworks to the architecture analysis considered in preceding sections, we have found significant difficulties. The metamodels are not detailed enough to provide the information required for the analysis. We are interested in information such as for instance the ontologies how easy it is to identify them. However, the concept of "ontology" is not found in the metamodels. In addition we are interested in a specific attribute of the ontology and some metamodels do not even systematically propose attributes at all.

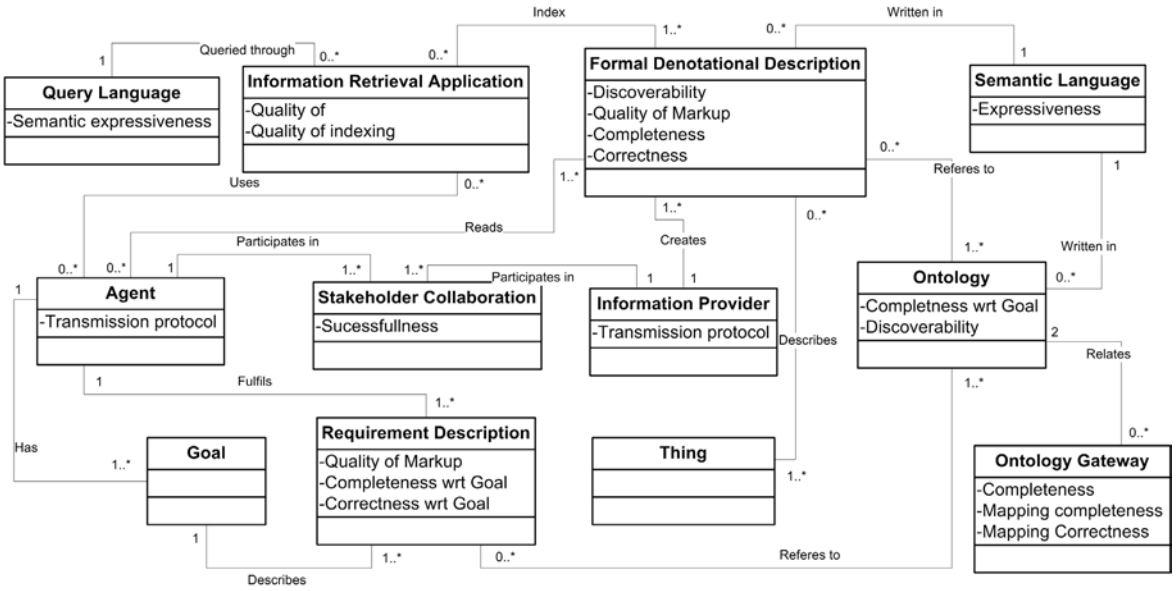



**Figure 5. The metamodel for sematic web interoperability analysis with its entities, attributes, and relations.**

## 6. The Metamodel for Interoperability Analysis on the Semantic Web

In this section, the metamodel suggested for semantic web interoperability analysis is presented. For the sake of interoperability analysis, only the extended influence diagram of section 4 is needed, but as discussed in the introduction, models can be useful in the decision making process. The metamodel is constructed to satisfy the requirements of the preceding section, i.e. it contains all of the entities and attributes necessary to conduct analysis of interoperability.

### 6.1 Entities of the Metamodel

The entities of the metamodel presented in Figure 5 have all been introduced in section 2 as concepts of the semantic web and will here only be listed with their relationships; *Information Providers* create *Formal Denotational Descriptions* that describes *Things* in the real world. The *Agent* performs a *Stakeholder Collaboration* with the information provider and fulfils a *Requirement Description* which describes a *Goal*. To be able to do this the agent must read formal denotational descriptions and understand its' content. Both the requirement description and the formal denotational description are written in a *Semantic Language* and relate to an *Ontology*. Ontologies can be related to other ontologies using *Ontology Gateways*. The agent can use *Information Retrieval Applications* as an aid. These applications are queried through a *Query Language*

### 6.2 Attributes of the Metamodel

For the purpose of semantic web interoperability analysis, a metamodel without attributes would be inadequate. In an architecture model, many important concepts are best captured as entity attributes. As seen in Figure 5, some entities have attributes that correspond to the extended influence diagram of section 4.

Firstly the sought after attribute interoperability, defined as successful collaboration on the semantic web, can be found as the attribute *successfulness* in the entity stakeholder collaboration. The formal denotational description contains four attributes, *discoverability* matching the node with the same name in the extended influence diagram, *quality of markup* and the two attributes affecting this quality, *completeness* and *correctness*. In similar manner the requirement description also contains the attributes *quality of markup*, *completeness* and *correctness*.

In order to match the extended influence diagram the other attributes of the metamodel are the *discoverability* and *completeness* of the ontology, the *expressiveness* of the semantic language, the ontology gateways *mapping completeness* and *correctness* and the *semantic expressiveness* of the query language.

There is one variable in the extended influence diagram not directly related to one attribute in the metamodel, namely the transmission protocol compatibility, this is evaluated by a comparison of the attributes *transmission protocol* in the entities agent and information provider.

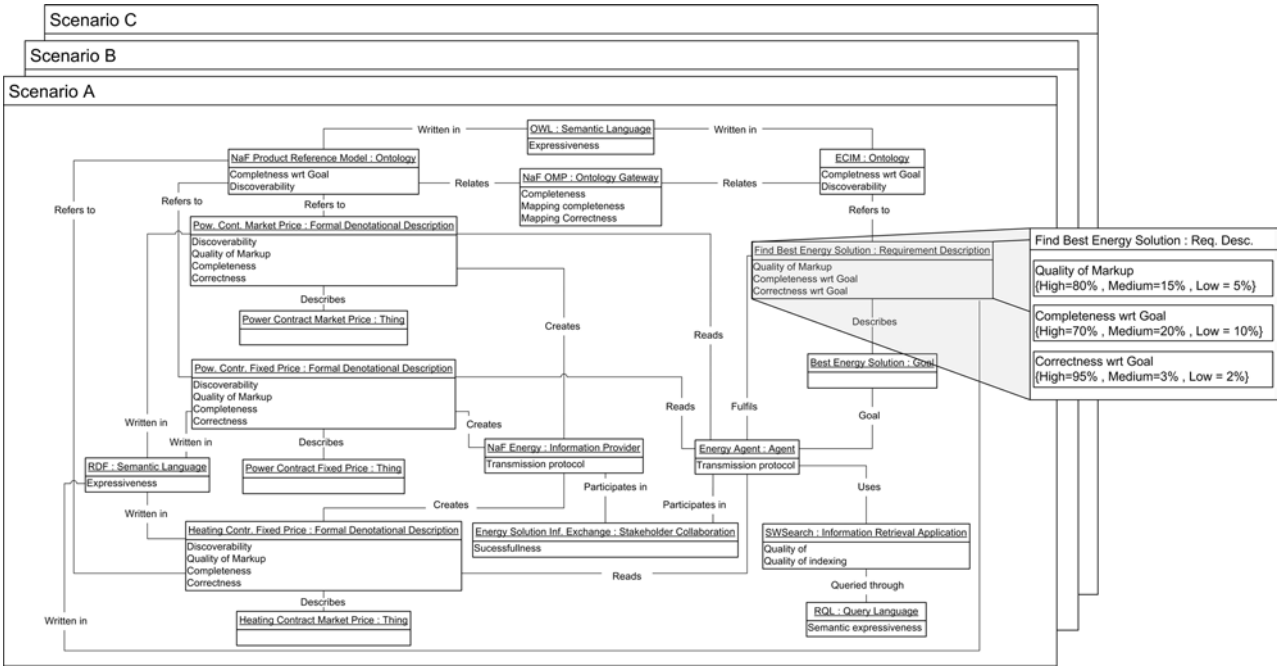



**Figure 6. The architecture model of scenario A, one out of three scenarios in the example. The requirement description entity is magnified to show the probability distributions related to each attribute.**

## 7 Modeling and Analyzing Using the Metamodel – An Example

This section presents an example of a semantic web interoperability analysis used as decision support in a project at the power company NaF Energy. The management of NaF Energy wants to increase the number of customers and therefore marketing campaigns have been initiated. As a part of this campaign a project has been started in order to improve the services offered by NaF and how these services are published on the web. The chief architect in the project suggested that the product portfolio should be published on the semantic web. The product portfolio consists of three offerings; one power contract at market price, one power contract at fixed price, and one heating contract at market price. There are several architectural options for our architect to consider in order to achieve the goal, i.e. that agents will find information regarding the services offered by NaF. There is for instance the choice of which ontology to use, some are more suitable for describing NaF's product portfolio, others are less appropriate but more widely used by the public. NaF can also chose on how much effort, e.g. time, to put into the project of marking the description of their products.

Several possible scenarios are therefore considered and the architect decides that a formal evaluation of the candidate scenarios is to be performed. Based on the metamodel of Section 6 information on the entities and their attributes are collected. Figure 6 describes one scenario, scenario A, in which an in-house and therefore very well suited but not well known ontology is used as well as an ontology gateway for the mapping towards more established ontologies.

Information collection can be done in several ways. In this example the expressiveness of the languages used was assessed by an expert. The completeness of the ontologies was found by looking in the actual ontology and comparing it to what was to be described. The completeness of the mappings towards the ontologies was assessed by interviewing the developers that should implement the solution and asking them how complete the mapping would be given the budget.

All collected variable values were then translated into discrete states, such as Low, Medium, or High. These were then used as input to the semantic web interoperability analysis employing the Bayesian theory in extended influence diagrams, as described in Section 3. When collecting information for the models, there is an issue of credibility [42]. Low credibility may lead to a large uncertainty in the analysis, making it difficult for

the architect to make a rational decision. For instance, studying the actual ontologies to find their completeness is a tedious work but, if done well, this will provide the architect with high credibility of the gathered information. Whereas, interviewing personnel, e.g. developers, to find the completeness in the mapping is less credible and also dependent on the experience of the personnel and the bias of the interviewer. Oftentimes it is very expensive to collect the information needed for a perfectly credible analysis. Since the analysis is based on the formalism of extended influence diagrams this credibility variation can be handled, thus the presented method of analysis provides the architect with an uncertainty degree in the result, shown in Figure 7 as bars indicating the range of values the result may assume.

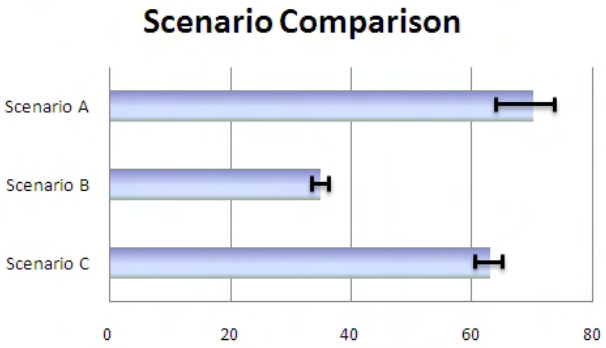



**Figure 7. The comparison between the service interoperability of the different scenarios, the black I-bars indicate the uncertainty of the assessments.**

The final result of the analysis is shown in Figure 7. As can be seen from the figure, scenario A achieved the highest interoperability rating whereas scenario B have a considerably lower degree of interoperability due to the use of an ontology not well known and not mapped to other, more well known, ontologies. Even though not detailed in this paper, using extended influence diagrams for analysis allows for assessments of subcomponents and it is therefore possible to discover that scenario A achieves its' high interoperability score due to a high degree of mark-up of both the requirement description and the formal denotational description, while the ontology completeness scores slightly lower due to incorrectness of the ontology mappings. Scenario C on the other hand has a lower degree of mark-up and discoverability but a higher degree of ontology completeness. The architect can now, based on the interoperability score, make a rational decision, choosing an architecture providing the degree of interoperability needed by the enterprise.

## 8 Conclusion

This paper has presented a framework for interoperability analysis on the semantic web and a metamodel supporting the analysis. The metamodel consists of entities with accompanying attributes that can be used to create architecture models from which it is possible to extract precisely the information that is needed for quantitative semantic web interoperability analysis. An example was provided illustrating the use of the metamodel and the extended influence diagram for analysis.

## 9. References


[1] Johnson, P., et al., "Enterprise Architecture Analysis with Extended Influence Diagrams" *Information System Frontiers, vol 9(2)*, Springer, The Netherlands, 2007.

[2] S. Decker, P. Mitra and S. Melnik, "Framework for the Semantic Web: An RDF Tutorial," *IEEE Internet Computing*, 2000, pp. 68-73

[3] T.R. Gruber, "A Translation Approach to Portable Ontology Specifications," *Knowledge Acquisition*, 1993, pp. 199-220.

[4] Dublin Core, Dublin Core Metadata Initiative, available at http://dublincore.org/, 2005

[5] I. Niles and A. Pease, "Toward a Standard Upper Ontology", *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)*, 2001.

[6] B. Smith, J. Williams, and S. Schulze-Kremer. "The Ontology of the Gene Ontology" *Proceedings of AMIA Symposium*, 2003

[7] A. Gomez-Perez and O. Corcho, *"Ontology Languages for the Semantic Web"*, IEEE Intelligent Systems, 2002, pp. 54-60.

[8] I Horrocks, P.F. Patel-Schneider, and F. van Harmelen."From SHIQ and RDF to OWL: the making of a web ontology language" Journal of Web Semantics, 2003, pp 7–26.

[9] Jurafsky, D., Martin, J.H., *Speech and Language Processing*, Prentice Hall, New Jersey, 2000.

[10] Cardoso, J., Sheth, A., "Semantic E-Workflow Composition", *Journal of Intelligent Information Systems, 21:3*, Kluwer, The Netherlands, 2003, pp 191-225.

[11] L. Ding et. al. "Swoogle: A Search and Metadata Engine for the semantic web", *Proceeding of the thirteenth ACM international conference on Information and knowledge management (CIKM)*, Washington, DC, 2004, pp 652-659

[12] G. Karvounarakis, S. Alexaki, V. Christophides, D. Plexousakis, M. Scholl, "RQL: a declarative query language for RDF", *Proceedings of the 11th international conference on World Wide Web*, Honolulu, Hawaii, USA, 2002, pp. 592-603

[13] L. Miller, A. Seaborne and A. Reggiori, "Three Implementations of SquishQL, a Simple RDF Query Language," *Proceedings of 1st International Semantic Web Conf. (ISWC 2002)*, Springer-Verlag, 2002, pp. 423–435.

[14] N. Friedman, M. Linial, I Nachman and D. Pe'er, "Using Bayesian Networks to Analyze Expression Data." *Journal of Computational Biology*, Mary Ann Liebert, Inc., publishers., 2000, pp. 601-620.

[15] Shachter, R., "Evaluating influence diagrams" *Operations Research, 34(6)* Institute for Operations Research and the Management Sciences, Hanover Maryland, 1986, pp. 871-882.
[16] Howard, R.A., Matheson, J.E., "Influence Diagrams" *Decision Analysis Vol. 2(3)*, Institute for Operations Research and the Management Sciences, Hanover Maryland, 2005, pp. 127–143.
[17] Neapolitan, R., *Learning Bayesian Networks*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 2003.
[18] Jensen, F.V., *Bayesian Networks and Decision Graphs*, Springer New York, Secaucus, NJ, USA, 2001.
[19] Johnson, P., Lagerström, R., Närman, P., "Extended Influence Diagram Generation" *Enterprise Interoperability II – New Challenges and Approaches*, Springer, London, 2007, pp. 599-602.
[20] Shachter, R., "Probabilistic inference and influence diagrams" *Operations Research, 36(4)*, 1998, pp 36-40.
[21] Johnson, P., Ekstedt, M.: *Enterprise Architecture – Models and Analyses for Information System Decision Making.* Studentlitteratur, Lund, Sweden, 2007.
[22] Lagerström, R., "Analyzing System Maintainability Using Enterprise Architecture Models" *Proceedings of the 2nd Workshop on Trends in Enterprise Architecture Research (TEAR'07)* ,St Gallen, Switzerland, 2007, pp. 31-39,
[23] T. Berners-Lee, J. Hendler and O. Lassila, "The Semantic Web" *Scientific American*, Scientific American Inc., 2001, pp. 35-43
[24] J.Heflin and J.Hendler. "A portrait of the Semantic Web in action", *IEEE Intelligent System*, 2001, pp 54–59.
[25] M. Uschold, "Where are the Semantics in the Semantic Web?" *AI Magazine*, American Assocation for Artificial Intelligence, pp 25-36
[26] A. Sheth, C. Ramakrishnan, and C. Thomas, "Semantics for the Semantic Web: The Implicit, the Formal and the Powerful," *Int'l J.Semantic Web and Information Systems, vol. 1, no. 1*, 2005, pp 1-18
[27] J.Hendler. "Agents and the Semantic Web", *IEEE Intelligent System*, 2001, pp 30–37.
[28] N. Shadbolt, W. Hall, T. Berners-Lee, "The Semantic Web Revisited", *IEEE Intelligent System*, 2006, pp 96-101.
[29] E.P. Bontas, L. Nixon, and R. Tolksdorf, "A Conceptual Model for Semantic Web Spaces," *Technical Report B 05-14*, AG Netzbasierte Informationssysteme, Freie Universität Berlin, Germany, 2005.
[30] Linthicum, D., *Enterprise Application Integration*, Addison-Wesley, New Jersey, 2000.
[31] Kasunic, M., Anderson, W., "Measuring Systems Interoperability: Challenges and Opportunities" *Technical Note, CMU/SEI-2004-TN-003*, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, 2004.
[32] Morris, R., Levine, L., Meyers, C., Place, P., Plakosh, D., "Systems of Systems Interoperability: Final Report" *Technical Report CMU/SEI-2004-TR-004* Software Engineering Institute, Carnegie Mellon University, Pittsburgh, 2004.
[33] Tolk, A., Muguira, J., "The Levels of Conceptual Interoperability Model", *Proceeding s of the 2003 Fall Simulation Interoperability Workshop*, Orlando, FL, 2003
[34] Ford, T., Colombi, J., Graham, S., Jacques, D., "The Interoperability Score", *Proceedings of the Fifth Annual Conference on Systems Engineering Research*, Hoboken, NJ, 2007
[35] IEEE, *Standard Glossary of Software Engineering Terminology. Std 610.12-1990*, The Institute of Electrical and Electronics Engineers, New York, 1990.
[36] O. Corby, R. Dieng-Kuntz and C. Faron-Zucker, "Querying the Semantic Web with the Corese Search Engine," *Proc. 16th European Conf. Artificial Intelligence (ECAI 04)*, IOS Press, 2004, pp. 705–709
[37] OMG, *Unified Modeling Language: Superstructure*, version 2.1.2, November 2007
[38] OMG, *OMG Systems Modeling Language (OMG SysML™), V1.0*, available at http://www.sysml.org
[39] Department of Defense Architecture Framework Working Group, *DoD Architecture Framework, version 1.0*. Department of Defense, USA, 2004.
[40] Lankhorst, M. et al., *Enterprise Architecture at Work*, Springer-Verlag, Berlin, 2005
[41] D. Chen, G. Doumeingts and F. Vernadat, "Architectures for enterprise integration and interoperability: Past, present and future", *Computers in Industry*, Elsevier B.V., 2008
[42] E. Johansson and P. Johnson, "Assessment of Enterprise Information Security – Estimating the Credibility of the Results", *Proceedings of the 9th IEEE international Annual Enterprise Distributed Object Computing Conference*, The Netherlands, 2005.

# Semantic Enrichment of Enterprise Models by Ontologies-based Semantic Annotations

N. Zouggar, B. Vallespir, D. Chen

*IMS - LAPS/GRAI, Université de Bordeaux, CNRS*
*351, cours de la Libération 33405 Talence cedex, France*
*nabila.zouggar@laps.ims-bordeaux.fr*

## Abstract

*Enterprise modelling process can be seen as a knowledge-creating process. In this process the semantic conflicts in enterprise modelling is an important issue.*

*Enterprise models are used during the system life cycle by other stakeholders rather than those who developed it. They do not necessarily know the context in which the model was built and quite often are not familiar with the language used for modelling. This situation makes the model to loose its semantics during its exploitation and creates ambiguities and difficulties in its use.*

*This paper will show at firs where the semantic conflicts stand within the enterprise model creation process. Then we propose a methodological approach to follow for the elaboration of enterprise model with the aim of keeping their semantics during their life cycle, by the use of ontologies-based semantic annotations.*

## 1. Introduction

Enterprise models aim at representing the whole or part of an enterprise. They can be informal, semi formal or formal. These models are to be understood, used and re-used by different persons with different knowledge backgrounds. For that the semantic of the concepts used must be clear and without any ambiguity. This paper presents the latest development of semantic enrichment of enterprise modelling using ontologies.

First we will define where the semantic problems related to enterprise model creation process is, on the basis of the SECI model (Socialization, Externalization, Internalization, Combination) which focused on the knowledge creation and transformation. Then we will propose an approach to solve the semantic problem by using ontologies which have an interesting property: the formal capacity of concepts representation. This property will help us to semantically enrich enterprise models. Future works and perspectives are discussed as part of conclusion.

## 2. Semantic in enterprise modelling

In this section, we will define what enterprise modelling is and we will present the enterprise modelling process. Before identifying the semantic problem in enterprise modelling, we will define what semantics is in the context of our research and will present the semantic conflicts which we can meet within enterprise modelling.

### 2.1 Enterprise modelling

Enterprise modelling aims to construct a model of whole or part of the enterprise, and generally of any organization, considered as a system, to explain the structure and the organization or to analyze their behaviour. The model must also be able to represent the particular point of view of an actor.

Several languages of enterprise modelling allow the construction and the exploitation of model according to steps of the system life cycle which are often characterized by a level of abstraction (conceptual, organizational or technical).

Formalization degree of the models varies according to the languages used, it can be informal (such as natural language), semi-formal (such as language with graphic formalism) or formal (mathematical language). Most of time, the models based on informal language are used to describe an existing situation while the models based on a formal

language allow properties fixed in a given project to be checked [1].

The enterprise modelling process aim is:
- The obtention of the necessary information to build the model: this information is the result of a trade-off between stake-holders implied in the study, mainly analysts and persons of the enterprise).
- The construction and review of the preliminary model: a first model based on an enterprise modelling method will be proposed. This model may be related to the actual situation of the system modelled. The review will be done by the modelling team by taking the rules of modelling language..
- The formalization of the final model: at this stage the model is elaborated in its final form according to the formalism used.
- The explanation and justification: it is necessary to present the model to the members of the enterprise, explain the followed steps and the content of the model.

## 2.2 Semantics and semantic conflicts types

Semantics is the study of meaning. The word derives from Greek: semantikos. In linguistics it is the study of interpretation of signs as used by agents or communities within particular circumstances and contexts.

The sign is all components which describe an entity of a given system; it is represented according to 3 axes:
- Semantic: Relation between signs and the things they refer to.
- Syntactic: Relation of signs to each other in formal structures.
- Pragmatic: Relation of signs to their impacts on those who use them.

| | $T_1=T_2$ | $T_1\neq T_2$ |
|---|---|---|
| $D_1=D_2$ | Equivalence **No Conflict** | Synonymy **Low Conflict** |
| $D_1\cap D_2=D_2$ | Additional **Medium Conflict** | IS-A **Medium Conflict** |
| $D_1\cap D_2\neq 0$ $D_1\cap D_2\neq D_2$ $D_1\cap D_2\neq D_1$ | Overlap **Major conflict** | Overlap **Major conflict** |
| $D_1\cap D_2=0$ | Homonymy **Low Conflict** | Disjointness **No Conflict** |

**Table 1:** Semantic conflicts types (adapted from [2])

In the context of our research, we are interested in the semantic axis for any sign. Semantic is the meaning that each entity of an enterprise model during its life cycle can carry. This meaning must be the same on the time to avoid a problem of understanding of the model what can induce a poor exploitation and use of the model.

If we consider that the conceptualisation of an entity of an enterprise model is given by a term T and a definition D, then a concept C can be represented by C=(T, D).

Different combinations can result from this couple (T, D) which translate a set of possible semantic relations between similar concepts [2]. All these combinations give place to semantic conflicts which we can encounter in enterprise modelling as shown in table 1.

Ushold and Gruninger [3] presented the semantic continuum that fixed semantics of entities according to the degree of formalism:
-Tacit semantic which exists only in people mental;
-Semi-informal semantic (explicit and abstract), it is explicit but is often represented in an abstract way by generally using natural languages such as English or French;
-Semi-formal semantic indicates an explicit and relatively formal semantics which is intended mainly for human by using generally graphic formalisms such as the semantic models, UML diagrams, etc.;
-Formal semantics is based on rigorous mathematical formalisms (such as the description logic, first order logic, etc.) which enable to treat it in an automatic way.

The combination of semantic conflicts and the semantic continuum gives us the following matrix (Table 2). This table shows that the risk of conflicts increases when the formalism of the semantics decreases.

| Semantic / Conflict | Tacit | Semi-informal | Semi-formal | Formal |
|---|---|---|---|---|
| No | | | | * |
| Low | | | * | |
| Medium | | * | | |
| Major | * | | | |

**Table 2:** Semantic conflicts compared to semantic continuum

### 2.3 Model-creating process as knowledge-creating process

Nonaka, Toyama and Konno present in [4] a knowledge conversion model which it called SECI (Socialisation, Externalisation, Combination and Internalisation). For them, an organisation creates knowledge through the interactions between explicit knowledge and tacit knowledge. There are four modes of knowledge conversion:
-*Socialisation* enables the conversion of tacit knowledge into a new tacit knowledge through interaction between individuals. Semantic of the concepts used in this mode of conversion is tacit.
-*Externalisation* is the process of articulating tacit knowledge into explicit knowledge. It is the action of filling information using language. At this level semantics can be semi-informal or semi-formal.
-*Combination* involves the conversion of explicit knowledge into more complex sets of explicit knowledge. Semantic is formal in this mode of conversion.
-*Internalisation* of newly created explicit knowledge using tacit knowledge. It is shared across the organization. At this level semantics may be semi-informal or semi-formal.

The modes where we can meet semantic conflicts in the knowledge creating process are: socialisation where the conflict is “major”; the externalisation and internalisation where the conflicts can be “low” or “medium”.

If we compare the four modes of knowledge conversion and the enterprise modelling process we can propose the following correspondences:
- *“To obtain the necessary information to model”* can be associated to *“Socialization”*, since the analysts will cooperate with the people of the enterprise in order to collect information.
-*“Construction and review of the preliminary model”* can be associated with the *“Externalisation”*, since the collected information in the first step will be transformed into a model so, in explicit knowledge.
-*“Formalisation of the final model”* can be associated with the *“Combination”*, since we change or correct the model designed at the previous step in another model, therefore we go from explicit knowledge towards another explicit knowledge.
-*“Explanation and justification of the model”* can be associated with the *“Internalisation”*, since we expose the model to a group who will create new tacit knowledge by interpreting it.

If we carry on the comparison, we meet semantic conflicts in the enterprise modelling process on the same levels as for the knowledge conversion (figure 1).

### 2.4 Conclusion

We found semantic problems in enterprise modelling on three levels of modelling process which are: obtain the information, construct the model and explain and justify the model. These semantic problems handicap enterprise modelling and make it difficult to use on the ground, considering the need to bring a solution of semantic enrichment of enterprise models which we will see in the following section.

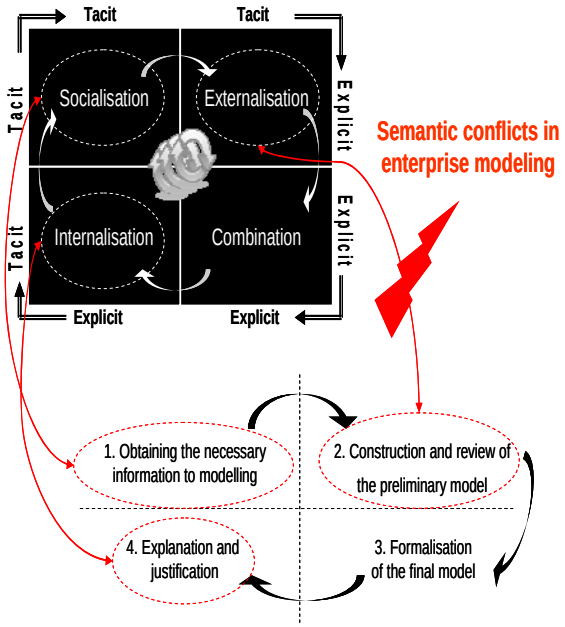



**Figure 1:** Semantic conflicts in enterprise modelling

## 3. Semantic enrichment of enterprise models

As seen previously, semantic conflicts exist in different step of the modelling process, it is due to poor conceptualization of entities used. The resulting model will surely include concepts bringing semantic conflicts. We must add to these concepts an additional component that will enable them to overcome these semantic conflicts. This is known as semantic enrichment of enterprise models.

The component we need for semantic enrichment must allow each entity involved in the model to carry an explicit semantics during its life cycle. We propose

to do that using an ontology-based semantic annotation.

Ontology is an explicit and formal specification of a conceptualisation of a domain of interest [5]. This definition stresses two key points: the conceptualisation is formal and hence permits reasoning by computer and a practical ontology is designed for some particular domain of interest.

Before explaining how to use ontology-based semantic annotations to enrich enterprise models, we present the complementarities and mappings existing between enterprise modelling and ontologies.

### 3.1 Complementarities and mapping [6]

The definitions given previously enable us to say that there is a link between enterprise modelling and ontologies. First we can affirm that both have as purpose to support the modelling of an enterprise. More particularly, research in ontology in enterprise domain mainly focuses on enterprise concepts identification and description; while research in enterprise modelling deals also for a part with the concept definition (for example GRAI[1] conceptual model) but mainly focuses on modelling languages and model construction using languages. Thus we can tentatively say that a possible overlapping is the concepts identification (Figure.2).

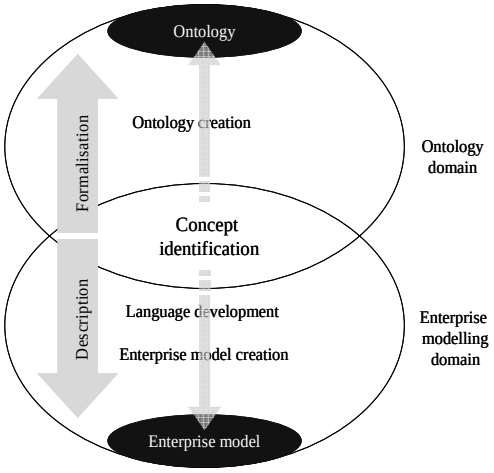



**Figure 2.** Common elements to enterprise modelling and ontologies

However a deeper analysis leads to conclude that the conceptual models developed in the enterprise modelling research are mainly informal and do not allow to capture precisely the semantics of the concepts. In the contrary, ontology techniques used to describe enterprise concepts are more formal and thus allow a better definition of the semantics.

The difference stands also in the content of the models. The enterprise model represents the structure or the operation of the enterprise whereas ontology organizes only the concepts used and the relations between them. In other words, ontologies in the domain of enterprise modelling such as TOVE [7], can be considered as enterprise ontology rather than enterprise model in the sense that there is no associate modelling language in ontology to allow building enterprise mode. Ontology techniques are useful to elaborate enterprise meta-models rather than developing enterprise modelling techniques and models. Thus the two approaches are complementary.

This difference can be beneficial for enterprise modelling; indeed the use of ontologies can mitigate the semantic deficit of the languages and the models that are primarily presented under their syntactic component. This situation is particularly highlighted when we seek to exchange a model or to federate distinct modelling languages. The users face often a problem of understanding due to the fact that the analysis suggested is based primarily on a syntactic analysis of the components; the semantics of the latter being not very explicit.

### 3.2 Ontology-based semantic annotation for enterprise model

As shown Figure 3, semantic enrichment of enterprise model will be done by the association of each entity model to a concept of an ontology, called Reference Ontology (RO). This association will be done by using the semantic annotations which carries information about the link between the model entity and the concept that goes with it in the reference ontology.

We will see what ontology to use for semantic enrichment of enterprise models and how semantic annotations are presented to that.

#### 3.2.1 Reference Ontology

Ontologies consist of concepts (also known as classes), relations (properties), instances and axioms. A more succinct definition of an ontology is as a 4-tuple <C, R, I, A>, where C is a set of concepts, R a set

[1] GRAI is a set of methodological modules, which contribute to the improvement of enterprise performances through enterprise modelling [18].

of relations, I a set of instances and A a set of axioms [8].

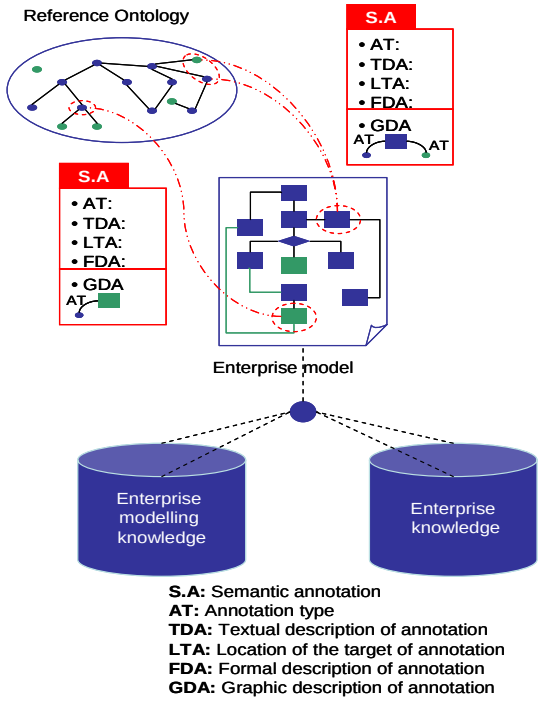



**Figure 3.** Semantic enrichment of enterprise model

The languages used for the construction of ontology may be classed as for the enterprise models: informal (understandable for the user but difficult to check the absence of redundancy or contradiction); semi formal (increased clarity and reduced ambiguity) and formal (possibility to check redundancy and consistency) [9].

The choice of the formalization degree is done according to the use of ontology. Indeed, if the aim is the support of communication between people, then the representation of ontology may be informal since it is precise enough to capture the semantic of each one. If now ontology must be used by software tools then the semantics must be formal.

In our research, we want to avoid the ambiguities that may occur in the enterprise model caused by semantic conflicts. Therefore, we choose between two types of ontology: informal and semi-formal. The first is not interesting, since we can not verify or validate it. A semi-formal ontology would be in our case more appropriate. As observed by Gruber in [10], currently the ontologies that are semi-formal have demonstrated very high practical value. Ontology development effort for semi-formal ontologies can be significantly smaller compared to that required for developing formal ontologies or ontologies with more expressive representations.

The characterizing feature of languages belonging to this category is a diagrammatic approach to knowledge representation [11]. This class of languages, usually, represents concepts as (labeled) nodes of a graph (/net). For what the relations concern, they are represented either as arcs of the graph or by using nodes, as well as concepts, but with a different shape. The languages which are mainly characterized by a graphic notation are: Conceptual graphs, Semantic networks, UML, Topic Maps and Concept Maps.

The steps to follow to create ontology are [12]:
- Determine the field and scope of the ontology. Determine the degree of formalisation needed, choose an appropriate ontology language.
- Study the possibility of using existing ontologies, to extend and refine them. Reuse existing ontologies can even constitute a requirement if our system needs to interact with other applications which already use specific ontologies or controlled vocabularies.
- Enumerate the significant terms in ontology. Indeed, it is useful to note in a list form all the terms to be treated or explained to a user, and the properties related to these terms.
- Define the classes and their hierarchy.
- Define the class properties (attributes) and their facets (values types, authorized values, number of values, etc.).
-Create class instances in the hierarchy.

### 3.2.2 Semantic Annotation

The annotation is one of the most common forms of meta-data in the Web context, it is also graphic or textual information attached to a document and generally placed in this document.

The semantic annotation is a particular case of annotation because it refers to ontology. It can be made in the form of comments, of explanations note, questions or another type of external remark which can be attached to a document or a selected part of this document [13].

To perform an annotation it is necessary to proceed through the three following phases which are [14]:
- The location which consists in placing in the document the ontology concepts references that it contains. These elements are considered as meta-data,
- The instantiation which allows to give attributes values of the concepts using information present in the document,

- The enrichment which aims at adding information by means of concepts attributes which could not be given values in the previous phase.

We note that in the first two steps, there are not information addition but rather localization and characterization of information already present. They are insertion steps. At the last step, the document is enriched by information which did not exist; it is a step of annotation formalized by meta-data.

## 3.3 Structured approach for semantic enrichment of models

The approach that we propose for semantic enrichment of enterprise models follows 6 steps, which are (figure 4): Constitute the team, Define reference ontology (RO); Delineated enterprise model (EM) concept; Match Enterprise model concept to reference ontology concept; Construct annotation; Deploy enrich model.

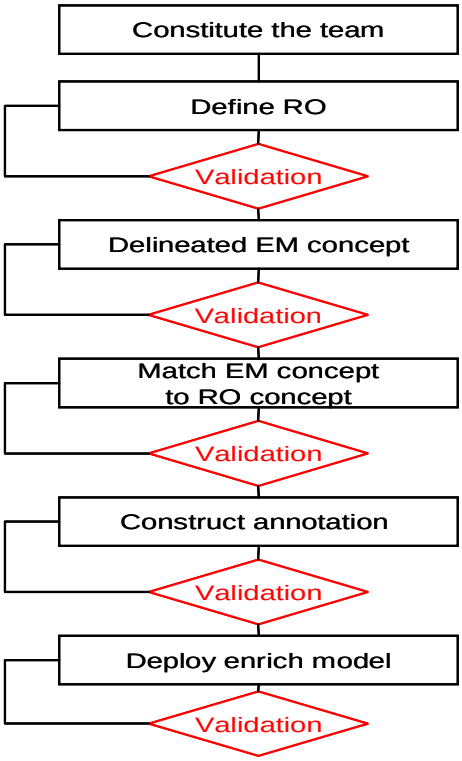



**Figure 4.** Methodological approach to enrich model

### 3.3.1 Constitute the team

Two teams must be constituted to carry out semantic enrichment of model, one to handle the enrichment tasks (execution team), another to deals with validation at each step of enrichment (supervision team). In both teams skills required are: enterprise domain, enterprise modelling, ontologies and semantic annotations.

### 3.3.2 Define reference ontology

The enterprise model contains concepts directly related to the field of business and others concepts which are related to enterprise modelling language used. It leads to find one or several ontologies bringing together all this knowledge. Otherwise we have to build an adequate ontology.

The validation is done by the supervision team to check if the ontology choice is suitable for model enrichment.

### 3.3.3 Delineated enterprise model concept

The execution team will bring together the concepts of the model to enrich to decide on definitions to give to the concepts for not having latest ambiguities on its. This step is very important since it will enable us in the next step, to involve model concepts to correct reference ontology concepts.

To validate this stage, we must ensure that all concepts have been well defined and that these definitions are going in the right direction.

### 3.3.4 Match enterprise model concept to reference ontology concept

Enterprise model concepts must be linked to those of the ontology that correspond in a most adequate way. This correspondence is based on the definition that we gave to each model concept in the previous step.

There are methods that can measure the similarity or semantic distance between two concepts. The most interesting is the one proposed by D'Amato *et al.* [15] who propose a measure of similarity based on the interpretation of concepts. They are described with the logic of descriptions [16] defined as follows:

$$s : l \times l \rightarrow [0..1]$$

$$s(C,D) = \frac{|C \cap D|^I}{|C|^I + |D|^I - |(C \cap D)|^I} \times \max\left( \frac{|(C \cap D)|^I}{|C|^I}, \frac{|(C \cap D)|^I}{|D|^I} \right)$$

Where $(.)^I$ is an interpreting function and $|.|$ represents the cardinal of set.

The problem with these methods of measurement is that ontologies and models should be formally represented, which is not the case with our study. So, this measure of similarity between concepts will be done manually.

The validation is done ensuring that the similarity between the concepts is real.

### 3.3.5 Construct annotation

After finding the corresponding concept in reference ontology, execution team will build the annotation to clarify what is the relationship that exists between the two concepts (ontology and model).

The structure of a semantic annotation can be represented as in figure 5:

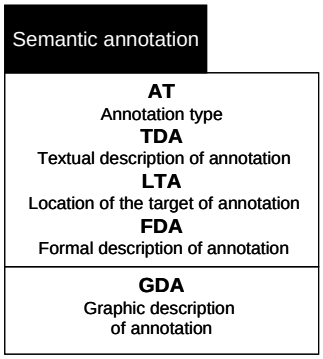



**Figure 5.** Semantic annotation schema

This schema includes the following elements [17]:

- Annotation type:
  - Decoration: annotations are comments associated with the resource;
  - Link: annotations are links;
  - Instance Identification: the annotated object (U#X) is an instance of a given class and the annotation content (Ref2Ontology) may be a link to that class (uri);
  - Aboutness: no assertion is made about the existence of an instance of the concept, but there is a loose association with the concept;
  - Pertinence: the target of the annotation may be of interest for the annotated object;
- Textual description of the annotation: Human readable of annotation content;
- Location of the target of the annotation: link to a reference ontology; this link is assumed to be the URI of its target.
- Formal definition of the annotation: expression of complementary information, like the type of relationship that holds between the annotated object and the target of the annotation (exact/partial match, more/less general, etc.). The value of this formal definition depends on the type of the annotation. For instance, in the information model perspective, this part may contain the definition of integrity constraints, while in other perspectives, it may contain the definition of the relevant business rules, etc. Moreover, this part of the annotation scheme is intended to be machine readable and interpretable. Therefore, its content is preferably expressed using a formal language. In the current experiment, the formal definition (called constraints in the annotation scheme) was expressed using UML/OCL.
- Graphic description of the annotation: graphic representation of the annotated concept is given related to the reference ontology.

The validation of this step ensures annotations to be correctly made.

### 3.3.6 Deploy enrich model

In this section, we collect all the information built in the previous steps to form the enriched model with all the annotations that connect each model concept to its equivalent in the reference ontology.

The validation is done by checking all these annotations and validating the meaning that the team wanted to give to the model.

## 4. Conclusion

Semantic problems in enterprise modelling were presented in this paper by the connection that we made between enterprise modelling process and the conversion knowledge model (SECI). The relationship that we have found, allows saying that the modelling process is also a knowledge-creating process. But if the semantics of such knowledge is not formally defined, the models will be confronted with problems of understanding. That is why we propose to use ontologies.

Complementarities between enterprise modelling and ontology have been identified. Ontologies have the interest to bring a degree of formalization missing in enterprise modelling. That is initially possible by using an ontology based semantic annotation that binds the concepts used in a model with an ontology that already exists or that should be built. We proposed in this paper a 6 step approach to carry out semantic enrichment of enterprise models.

Currently, we are investigating the problems of understanding of enterprise model. This is done by evaluating how individuals understand a model when first this model is natural and, second when it is accompanied by an ontology-based semantic annotations.

The results of these studies will be presented in a coming publication.

## 5. References


[1] V. Chapurlat, M. Larnac, E. Lamine and J. Magnier, "Definition of a formal analysis framework for existing enterprise modelling approaches", Proceeding of ICIMS-NOE conference, Life cycle approaches to production systems: management, control, supervision (ASI), Louvain, Belgique, 1999.

[2] M. Kavouras, "A unified ontological framwork for semantic Integration". International Workshop on Next Generation Geospatial Information, Cambridge, UK, 2003.

[3] M. Uschold and M. Gruninger, "Creating Semantically Integrated Communities on the World Wide Web", Invited Talk Semantic Web Workshop Co-located with WWW 2002, Honolulu, HI, 2002.

[4] I. Nonaka, R. Toyama and N. Konno, "SECI, Ba and Leadership: a Unifed Model of Dynamic Knowledge Creation", International Journal of Strategic Management, Long Range Planning, 33 (1), pp.5-34, 2000.

[5] T. Gruber, "What is an ontology? Summary statement of Gruber's defintion of ontology". http://www-ksl.stanford.edu/kst/what-is-an-ontology.html, 2001.

[6] N. Zouggar, D. Chen and B. Vallespir, "Enterprise modelling and Ontology", Proceeding of 17th IFAC World Congress, Seoul, 2008.

[7] M. Gruninger, K. Atefi, and M.S. Fox, "Ontologies to support process integration in enterprise engineering" Computational and Mathematical Organization Theory, Volume 6, Number 4, pp. 381-394, 2000.

[8] J. Davies, R. Studer And P. Warren, "Semantic web technologies", Trends and research in ontology-based systems, Wiley edition, England, 2006.

[9] M. Uschold and M. Gruninger, "Ontologies: principles, methods and applications", The Knowledge Engineering Review, Volume 11, Number 2, pp. 93-155, 1996.

[10] T. Gruber, "It is what it does: The pragmatics of ontology", Invited talk at Sharing the Knowledge International CIDOC CRM Symposium, http://tomgruber.org/writing/cidoc-ontology.htm, 2003

[11] ATHENA, "SoA on Ontologies and the Ontology Authoring and Management System, with Ontology Modelling Language. Part 1 – State of the Art on ontologies: languages, tools and contents". Deliverable D.A3.1, 2004.

[12] N. Fridman Noy and D-L. McGuinness, "Ontology development 101: A guide to creating your first ontology", Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, 2001.

[13] D. Taniar, J. W. Rahayu, "Web semantics and ontology", Idea Group, pp. 165-187 2006

[14] E. Desmontils and C. Jacquin, "Annotation sur le web: notes de lecture. Journées de l'AS Web sémantique" http://www.lalic.paris4.sorbonne.fr/stic/, 2002.

[15] C. D'Amato, N. Fanizzi and F.Esposito, "A dissimilarity measure for ALC concept descriptions" , SAC'06: Proceedings of the 2006 ACM symposium on Applied computing, ACM Press, pp. 1695–1699, New York, 2006.

[16] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider, "The Description Logic Handbook", Cambridge University Press, Cambridge, UK, 2003.

[17] N. Boudjlida et al., "A practical experiment on semantic enrichment of enterprise models in a homogeneous environment", INTEROP Deliverable DTG4.1, 2006.

[18] B. Vallespir, C. Merle et G. Doumeingts, "The GRAI Integrated Method : a technico-economical methodology to design manufacturing systems", In actes of 1st IFAC workshop on "A cost effective use of computer aided technologies and integration methods in small and medium sized companies", CIM'92, Vienne, Autriche, 1992.

# Towards Enterprise Interoperability Service Utilities

Brian Elvesæter
*SINTEF ICT, P.O. Box 124, Blindern, N-0314*
*Oslo, Norway*
*brian.elvesater@sintef.no*

Francesco Taglino
*CNR-IASI, 00185 Roma, Italy*
*francesco.taglino@iasi.cnr.it*

Enrico Del Grosso
*TXT e-Solutions, 20126 Milano, Italy*
*enrico.delgrosso@txt.it*

Gorka Benguria
*ESI-Tecnalia, 48170 Zamudio, Spain*
*gorka.benguria@esi.es*

Alberto Capellini
*ATOS, 28037 Madrid, Spain*
*alberto.capellini@atosresearch.eu*

## Abstract

*This position paper presents the vision and initial results of the COIN (FP7-216256) European project for the development of open source Enterprise Interoperability (EI) services following the Software-as-a-Service Utility (SaaS-U) paradigm.*

## 1. Introduction

COIN (FP7-216256) is an integrated project [1] in the European Commission Seventh Framework Programme. The mission of the COIN project is to study, design, develop and prototype an open, self-adaptive, generic ICT solution where Enterprise Collaboration (EC) and Enterprise Interoperability (EI) services will be an invisible, pervasive and self-adaptive knowledge and business utility at the disposal of the European networked enterprises.

In this paper we position the COIN initial results for the development of open source services, which will be integrated according to the Enterprise Interoperability Research Roadmap grand challenge of Interoperability Service Utility (ISU) [1] into a coherent pool of EI services as a contribution to the Software-as-a-Service Utility (SaaS-U) vision.

The rest of this paper is structured as follows: In section 2 we give a short overview of related work. Section 3 describes the identified candidates for implementing a first set of EI services. In section 4 we describe how the EI services relate to the SaaS-U paradigm. Conclusions and future work are presented in section 5.

## 2. Related work

The COIN project is founded on the vision of an innovative integration and improvement of previous project results related to Enterprise Interoperability. Enterprise Interoperability [1] is a relatively recent term that describes a field of activity with the aim to improve the manner in which enterprises, by means of information and communications technology (ICT), interoperate with other enterprises, organisations, or with other business units of the same enterprise, in order to conduct their business.

One aim of the COIN project is to provide a foundation for EI services based on the principles from existing interoperability frameworks and the results from previous projects, which are to be integrated with the new and open source COIN service platform supporting the SaaS-U provisioning paradigm.

Results from previous European projects on interoperability were collected, analysed and consolidated into a set of baseline EI services according to a state-of-the-art analysis. Due to the requirement of an open source COIN service platform the state-of-the-art analysis focused primarily on available open source solutions, in particular from the following projects:

- ABILITIES (FP6-027306) [2] addressed Enterprise Interoperability among SME in Enlarged Europe. The project released tools, services and Web portals to solve communication and interoperability issues related to business documents exchange.
- ATHENA (FP6-507849) [3] was the integrated project focusing on Enterprise Interoperability in FP6. The project developed an interoperability framework baseline comprising of a set of models, tools, services and methods to solve interoperability issues.
- INTEROP (FP6-508011) [4] was a network of excellence which joined 50 of Europe's most prominent research institutes focused on interoperability research. The project coordinated research results and development of interoperability solutions within Europe.
- SUPER (FP6-026850) [5] addresses the use of semantic techniques in the context of business process modelling and management. The project developed several tools for semantic business processes modelling, management, monitoring and analysis.

## 3. Baseline EI services

In the COIN context, Enterprise Interoperability services provide functionality for applying IT solutions that overcome interoperability gaps between two or more enterprises and thus enabling them to set-up and run collaborations. The main goal of the EI services is to improve interoperability, mainly for SMEs, which means to reduce the costs of data reconciliation, systems integration and business processes synchronization and harmonization. Typical indicators will be in the cost of service composition and of data mediation and reconciliation.

The EI services will be realised according to the interoperability dimensions in the **COIN EI Services Framework,** which is based on the ATHENA Interoperability Framework (AIF) [6] and harmonised with the European Interoperability Framework (EIF) [7] in the areas of technical, semantic and organisational interoperability. The AIF defines an interoperability reference model (see Figure 1) that focuses on the alignment and exchange of provided and required artefacts of collaborating enterprises. Interoperations can take place at various levels, namely *enterprise*, *business process*es, *services* and *information/data*.

For each of these levels specific EI services may be developed to support *collaborative enterprise modelling*, *cross-organisational business processes*, *flexible execution and composition of services* and *information/data interoperability*. The *model-driven interoperability* approach cuts across all levels and provides model-based solutions to formalise and exchange the provided and required artefacts that must be negotiated and agreed upon. The *semantic mediation interoperability* approach concerns the application of ontology-based techniques for semantic reconciliation of the models expressed on the different levels.

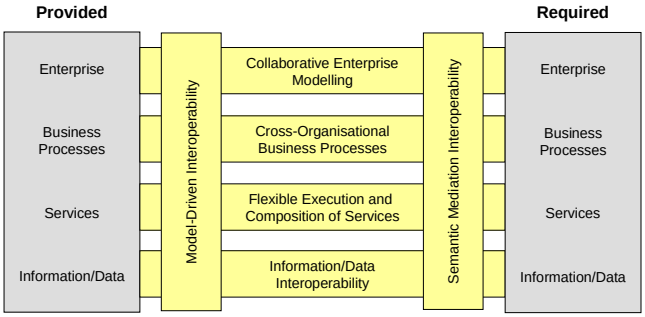



**Figure 1. Interoperability reference model**

Based on the state-of-the-art analysis a set of baseline EI services were specified [8]. These services will be implemented as Semantic Web Services and tested in relevant industry pilots.

### 3.1 Model-driven interoperability

Model-driven interoperability services support enterprises to formalise, exchange and align models that are relevant to set up collaborations. Model-driven interoperability is influenced by the ongoing standardisation activities around the OMG Model Driven Architecture (MDA) [9]. MDA is a technology framework that defines an approach in which visual modelling languages and visual models can be used to integrate the huge diversity of models used in the development of software systems.

The Eclipse community provides open source implementations of MDA specifications for metamodelling [10] and model transformations [11, 12] which can be used to align different models. The model transformation engines was developed as part of the MODELWARE project [13].

These technologies will be the basis for the development of the **COIN Model Transformation Service Engine** which will provide functionality for storing, searching and executing model-to-model and model-to-text transformations, in order to overcome

the incompatibilities between different modelling formalisms.

### 3.2. Enterprise modelling interoperability

Enterprise modelling interoperability services support enterprises to factually co-operate with other, external organisations in spite of e.g., different working practices, legislations, cultures and commercial approaches. Enterprise modelling is a discipline devoted to the understanding and improvement of the organisations through the development of enterprise models.

The COIN project will provide two baseline enterprise modelling interoperability services:

- **Enterprise Model Interchange Service** based on the POP* metamodel [14] developed in ATHENA. The POP* metamodel defines a core set of enterprise language constructs in the modelling dimensions Process, Organisation, Product and other dimensions like System and Decision to be defined in an enterprise model. The POP* metamodel is a flexible intermediate language that facilitates model exchange between different enterprise modelling tools. The POP* metamodel was included as an annex to the ISO19440 standard [15].
- **Enterprise Interoperability Maturity Assessment Service** to assess and improve the level of interoperability. An SME needs to establish new relationships with other organisations to respond to the multiple changes in their environment. The readiness to interoperate is a key attribute to efficiently respond to those changes. In that sense there are some models that guide organisations in the identification of good practices that improve the capability of the organisation to interoperate with others.

### 3.3. Business process interoperability

Business process interoperability services support enterprises to make proper external views of enterprise internal processes synchronised by a collaborative inter-enterprise business process.

Process models play an important part in Business Process Management (BPM) environments. As a result, business process modelling tools are an integral part of today's software designing process. In this area, significant improvements are expected especially concerning semantic interoperability. From the point of view of supporting cross-organisational business processes, all the life cycle of business processes should be taken in account. This means that not only business process modelling support, but management and monitoring services must be provided. Finally, we should consider also analysis services as a way of making business people more independent of IT people when performing business analysis within and across enterprises.

Potential candidates for the development of COIN baseline business process interoperability services are:

- **Cross-Organisational Business Process (CBP) Modelling Service** based on the CBP metamodel [16] developed in ATHENA. The CBP metamodel defines language constructs for modelling cross-organisational business processes using the concepts of view process and private process. A CBP defines the interactions between two or more business entities which links together view processes. A view process combine different (internal) private processes to an abstract level that enables companies to hide critical information from unauthorized partners.
- **Semantic Business Process Modelling Service** based on projects results from SUPER. The services deals with enrichment of existing business process models with semantic annotations and are expected to enable semantic interoperability by creating a common understanding of the business semantics.
- **Semantic Business Process Management Service** based on projects results from SUPER. The service will manage the life-cycle of deployed business process models independently of the underlying process engines actually executing the model.

### 3.4. Service interoperability

Service interoperability will be supported by the **COIN Baseline Service Platform**. The service platform will be based on a SESA (Semantically-Enabled Service-oriented Architecture) as the founding block enriched with scalability, trust and security, and intelligent negation capabilities.

The Web Service Execution Environment (WSMX) [17], which is the most complete and functional implementation of a SESA as of today, will be used for implementing the COIN platform. WSMX supports semantically-enabled functionalities such as dynamic discovery, selection and mediation, as well as

semantically-enabled control and connection functions such as service invocation and interoperation thus directly contributing to the exposition, integration, composition and invocation of services advertised by the platform.

Services, requester inquiries and related data model are going to be formalized using the Web Service Modeling Ontology (WSMO) formalism on top of which WSMX has been designed.

### 3.5. Semantic mediation interoperability

Semantic mediation interoperability services support enterprises to apply ontology-based techniques for semantic mediation such as semantic reconciliation of business documents in order to support interoperability among heterogeneous software applications. In literature, several platforms and frameworks addressing this issue exist. Among these initiatives, we here recall the ATHENA Semantic Reconciliation Suite [18], the Artemis Message Exchange Framework [19] and the MAFRA Mapping Framework [20].

In general, such complex solutions need the synergy of different tools and services that will be provided by COIN:

- **Ontology Management and Engineering.** Ontologies are a requirement for semantic mediation, and tools and methods for ontology building (e.g., Protégé, ATHOS), maintenance and evolution (e.g., METHONTOLOGY, UPON) are needed.
- **Semantic Mapping** allows definition of correspondences between documents or any kind of resource, and ontology, in order to describe in an unambiguous way such resources. Two kinds of mapping services have been analyzed: **Semantic annotation**, which works at conceptual level: on unstructured documents (e.g., text, audio, video); on structured documents (e.g., business document schema). Tools of the former case (e.g., MnM, Cohse) are mainly for supporting search and retrieval purposes. Tools of the latter case are mainly for semantic mediation (e.g., ASTAR, MWSAF). **Semantic Transformation rules:** respect to semantic annotation, semantic transformation rules have an operational purpose. They are applied to instance documents to drive the document mediation and reconciliation. Examples of such tools for building Semantic transformation rules are ARGOS and OWLmt.
- **Semantic Reconciliation Execution** is in charge to interpret semantic transformation rules, and to apply them against document instances, in order to actually transform such documents from an internal representation to another. Examples of Semantic Reconciliation Engines are ARES, are WSMX Semantic Mediation module.
- **Semantic Mapping Assessment** to understand how much two information systems can interoperate between each other;
- **Reconciliation Execution Monitoring** to check the effectiveness of the document mediation.

### 3.6. Information and data interoperability

Information and data interoperability services support enterprises to exchange and share business documents among organisations, by filling interoperability gaps related to the payload (format and content) and to the messages and/or structures to be exchanged between different and heterogeneous data sources. The common problem faced by these services is extraction of information from a specific format of a data source, and then make this information available to another data source in another format.

The COIN project will provide two baseline data interoperability services:

- **Transactional Data Interoperability Service** which concerns the exchange of information between two distinct actors (for example a seller and a supplier) and provides functionality for the creation and application of business document format and content rules to be applied in the context of format and content changes.
- **Massive Data Interoperability Service** which concerns the exchange of information among multiple actors. The main functionality of the service is to map the data provided by data providers to a schema that represent the kind of information required by consumers.

## 4. Towards EI service utilities

The research projects on interoperability in the European Commission Sixth Framework Programme have developed a vast set of standalone software products and tools, as well as some Web-based

services to address interoperability issues. However, some of these solutions are difficult to integrate and use for SMEs.

Experiences from piloting activities in the ATHENA project suggested that Enterprise Interoperability is very challenging and that the expected gains from interoperability research will consist in finding technologies and methods that will fasten interconnection of applications through standardised Web infrastructure for software application communication and for collaboration [21].

Within the context of an enterprise, flexibility, fast development and re-configuration are important properties for software applications. This implies to avoid as much as possible the intervention of software engineers and developers, and to prefer direct parameterisation and configuration by software users. This also implies accurate ways to architecture enterprise applications that facilitates publication of information managed by the application, publication of services made available by the application for other applications and finally publication of services made available for the human users. Finally such architectures should make it possible to manage coherency of the application systems despite numerous existing interfaces and adaptation of the application systems within the whole enterprise.

The Enterprise Interoperability Research Roadmap [1] envision interoperability support as utility-like capabilities that needs to be supported by an enabling system of services for delivering basic interoperability to enterprises, independent of particular IT deployment. The term Interoperability Service Utility (ISU) is used to denote this overall system. The ISU is envisaged to provide interoperability as a technical, commoditised functionality, delivered as services.

The ISU challenge is addressed by COIN by providing a service infrastructure for Enterprise Interoperability in the business context of Enterprise Collaboration. This will not just create a service platform, but mainly a new business concept – the Software-as-a-Service Utility (SaaS-U) model. The baseline EI services will be offered as Semantic Web Services on top of the WSMX environment [17].

The SaaS-U paradigm fits well fit the ISU concepts and can be seen as a software application delivery model where a software vendor develops Web-native software services and hosts and operates them for use by its customers over the Internet. Customers do not pay for owning the software itself any longer but rather for using it on-demand. They use it through an API accessible over the Web and often written using Web services.

## 5. Conclusions and future work

In this paper we position the COIN service platform as a technology enabler for developing and offering Enterprise Interoperability services according to the Service-as-a-Software Utility (SaaS-U) paradigm.

We give a short description of potential candidates for the development of baseline EI services based on state-of-the-art analysis of previous FP6 research projects. The development of these services will follow two phases. The first phase involves wrapping and potentially re-implementing parts of the solutions as Web services. The second phase involves enriching the service descriptions with semantics in order to make them Semantic Web Services.

After the implementation of the baseline EI services, which are scheduled to be delivered by the end of 2008, focus will be directed on developing new and innovative EI services for (1) information interoperability, which will explore new service communication and coordination in business collaborations, (2) knowledge interoperability, which will stem upon the concept of semantic profiles and will develop the environment to expose, compare and semantically mediate such profiles, and (3) business interoperability, which will go beyond the cross-enterprise business process coordination and provide new formalisms and languages for interactive and collaborative BPM as well as tools for process and workflow mining and knowledge extraction.

## 6. References


[1] M.-S. Li, R. Cabral, G. Doumeingts, and K. Popplewell, "Enterprise Interoperability Research Roadmap, Final Version, Version 4.0", July 2006.

[2] ABILITIES, "ABILITIES Home Page", ABILITIES STREP. http://www.viewzone.org/abilities/ (last visited 2008).

[3] ATHENA, "ATHENA Home Page", ATHENA IP. http://www.athena-ip.org/ (last visited 2007).

[4] INTEROP, "INTEROP Home Page", INTEROP NoE. http://www.interop-noe.org/ (last visited 2008).

[5] SUPER, "SUPER Home Page", SUPER IP. http://www.ip-super.org/ (last visited 2008).

[6] A.-J. Berre, B. Elvesæter, N. Figay, C. Guglielmina, S. G. Johnsen, D. Karlsen, and S. Lippe, "The ATHENA Interoperability Framework", in Proc. of the 3rd International Conference on Interoperability for Enterprise Software and Applications (I-ESA'07), Madeira, Portugal, 2007, Enterprise Interoperability II, Springer, pp. 569-580.

[7] IDABC, "European Interoperability Framework for Pan-European eGovernment Services, Version 1.0", IDABC, 2004. http://europa.eu.int/idabc/en/document/3761

[8] COIN, "D5.1.1: State-of-the-Art and Baseline EI Services Specifications", COIN IP, Deliverable D5.1.1, July 2008. http://www.coin-ip.eu/
[9] OMG, "OMG Model Driven Architecture", Object Management Group (OMG). http://www.omg.org/mda/ (last visited 2008).
[10] eclipse.org, "Eclipse Modeling Framework (EMF)". http://www.eclipse.org/emf (last visited 2008).
[11] eclipse.org, "Model-to-model (M2M)". http://www.eclipse.org/m2m/ (last visited 2008).
[12] eclipse.org, "Generative Modeling Technologies (GMT)". http://www.eclipse.org/gmt/ (last visited 2008).
[13] MODELWARE, "D3.4 Model Transformation Tool Suite", MODELWARE IP, Deliverable D3.4, 11 September 2006. http:///www.modelware-ist.org
[14] ATHENA A1, "D.A1.3.1: Report on Methodology description and guidelines definition, Version 1.0", ATHENA IP, Deliverable D.A1.3.1, March 2005. http://interop-vlab.eu/ei_public_deliverables/athena-deliverables/
[15] ISO, "Enterprise integration - Constructs for enterprise modelling", International Organization for Standardization (ISO), ISO 19440, 2007.
[16] ATHENA A2, "D.A2.2: Specification of a Cross-Organisational Business Process Model, Version 1.0", ATHENA IP, Deliverable D.A2.2, June 2005. http://interop-vlab.eu/ei_public_deliverables/athena-deliverables/
[17] WSMX, "WSMX (Web Service Modelling eXecution environment)". http://www.wsmx.org:8080/wsmxsite/ (last visited 2008).
[18] ATHENA A3, "D.A3.3: Semantic Annotation language and tool for Information and Business Processes, Version 1.0", ATHENA IP, Deliverable D.A3.3, February 2006. http://interop-vlab.eu/ei_public_deliverables/athena-deliverables/
[19] V. Bicer, G. Laleci, and A. D. Y. Kabak, "Artemis Message Exchange Framework: Semantic Interoperability of Exchanged Messages in the Healthcare Domain", ACM Sigmod Record, vol. 34, no. 2, 2005.
[20] A. Maedche, B. Motik, N. Silva, and R. Volz, "MAFRA - A MApping FRAmework for Distributed Ontologies", in Proc. of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web (EKAW '02), 2002, Springer-Verlag.
[21] ATHENA A4, "D.A4.2: Specification of Interoperability Framework and Profiles, Guidelines and Best Practices", ATHENA IP, Deliverable D.A4.2, 2007. http://interop-vlab.eu/ei_public_deliverables/athena-deliverables/

# An Interoperability Service Utility Platform for Automobile Supply Chain Management

Yong Zhang, Shijun Liu, Yuchang Jiao
*School of Computer Science and Technology, Shan Dong University，Jinan 250101，P.R.China*
*evelyn0330@163.com, lsj@sdu.edu.cn, jiaoyuchang@gmail.com*

## Abstract

*The competitiveness of an enterprise depends not only on its internal performance to produce products and services but also on its ability to seamlessly interoperate with other enterprises. External and internal collaborative work needs more interoperable solutions. How to enhance interoperability between stakeholders and improve efficiency of supply chain management is the key issue that needs to be addressed in automoble industry .*

*This paper proposes a methodology that provides a guide on how to establish interoperability between enterprises through a federated approach. According to this methodology, an interoperability service utility platform is designed and implemented for automobile suppy chain management. In the platform, interoperability is considered to be a utility-like capability and delivered in the form of SaaS. This paper introduces the specification of these ISUs and proposes an interactive framwork which is used to establish interoperability between two SaaS applications.*

**Keywords:** Enterprise Interoperability, ISU, SaaS, Supply chain management.

## 1. Introduction

Today an enterprise's competitiveness is to a large extent determined by its ability to seamlessly interoperate with others. Enterprise Interoperability (EI) has therefore become an important area of research to ensure the competitiveness and growth of enterprises [1].

In modern manufacturing field, such as automobile industry, an entire manufacturing process often cooperated by assembly factory and part suppliers. Agile Supply Chain Management increasingly becomes an effective and important measure to enhance competitive advantage of enterprises that needs the support of agile information system to integrate their supply chain more effectively and quickly [2].

As auto manufacturers inexorably move their sourcing of components and low value-added operations offshore, to lower cost countries, so their supply chains increase in both distance and complexity. Many companies are faced with the challenge of providing an agile response to customers and yet operating a lean operation across an extended global supply chain. This is a challenge that needs a solution beyond the abilities of simply judgment, the telephone and spreadsheets [3].

Existing interoperability solutions are suitable only for large enterprises and SMEs lack cheap, easy to integrate and easy to customize solutions. Therefore, SMEs are still far behind in the reform of supply chain management due to their small IT budgets, crude process standards with little visibility data to enable them to share and compare with trading partners.

Therefore, our research work aims at developing a methodology that provides the guide on how to implement an interoperability solution in auto supply chain management through a federated approach. More precisely the methodology allows establishing interoperability by: (1) constructing a virtual enterprise by identifying and involving various actors and stakeholders; (2) dynamically composing available interoperability service utilities according to identified requirements; (3) evaluating and improving the interoperability solution in practice.

Under the guiding of this methodology, we design and develop an ISU platform, which is built on top of existing ICT infrastructure available in most of enterprises. In our platform, the ISUs are delivered in the form of SaaS[4], which are cheap, fast, reliable, and without major integration efforts, so they can be invoked by enterprises on the fly in support of their business activities.

The goal of ISU Platform is to provide a holistic solution enabling the collaborative supply chain management in a flexible and dynamic environment and especially to facilitate SMEs' participation to collaborative supply chain management processes.

## 2. Architecture of ISU Platform

According to the methodology mentioned above, we design and develop an ISU platform, which is used to deliver interoperability service utilities easily and quickly. Both the assembly factory and the supplier should first register to be users of this platform, where they can negotiate for goods and services. After the establishing of a virtual enterprise in our platform, the master of the V.E. could apply for the authorization to use these ISUs, such as SBM and APO.

The design of the platform is based on the SOA principle; all the functions are implemented as services, including web services and software as a service. In the

ISU platform, interoperability is considered to be a "utility-like" capability, and the ISUs are delivered in the form of SaaS.

The architecture of our platform is shown in figure 1, which has 5 different layers.

The bottom of the architecture is the data center of our platform. There are data bases for all kinds of important data needed in the normal execution of the ISU platform.

Portal is the presentation layer of ISU platform, and the use of the platform is organized in a way of "personal work-space". That's to say, when the user logs in our platform, he will work in his self-customized workspace to support the business activities. If the user is from a part supplier, he can get the orders from his assembly factory, and deliver the materials on time.

Support service layer is basic and fundamental layer in the platform. In this layer, it contains many services which are used to support the basic function of our platform, such as user management service, security service, data synchronization service, log service and so on. All the ISU services are based on these support services.

ISU services layer is the most important one in our platform. As the platform is designed to support the auto supply chain management, the interoperability service utilities are the core functions auto manufacturer enterprises will use to enhance the interoperability between stakeholders. In the ISU platform, it provides several widely used services, which are delivered in the form of SaaS. They are SBM service, APO service, SMS service and conference service etc. In real world, SBM service and APO service will directly influence the production of enterprises, while the SMS service and conference service are often used to support daily business activities that are not very important.

To build agile supply chain management for automobile industry, SBM provides a cooperative environment of supply business for assembly factory and its suppliers, where the assembly factory is in the center of the business. SBM will help the assembly factory to deal with businesses related to suppliers such as bill inquiry, inventory management, and payment management, etc.

Advanced Planning and Optimization (APO) is very important in the automobile industry, because it is at the leading edge of manufacturing management technology. The appeal of APO to manufacturers is obvious, because companies can optimize their supply chains to reduce costs, improve product margins, lower inventories, and increase manufacturing throughput. APO necessitates deciding when to build each order, in what operation sequence, and with what machines to meet the required due dates.

In real world, most of the SMEs can't afford the expensive software systems with the same function of SBM and APO. Therefore, the goal of the ISU platform is to facilitate SMEs' participation to collaborative supply chain management processes by invoking SBM service and APO service on the fly.

At present, SBM service and APO service are the most important services in the platform used to improve interoperability between enterprises. They will be introduced in the following sections.

The composite services layer of the architecture is more complex than the ISU services layer. In automobile supply chain management, SBM and APO are often used together to support the important business activities. Dynamically composing available interoperability service utilities according to identified requirements is the key step to construct a new application.

Therefore, a composite services layer is designed to deal with the complicated business requirements. This paper will introduce an interactive framework used to establish interoperability between two ISUs: SBM and APO.

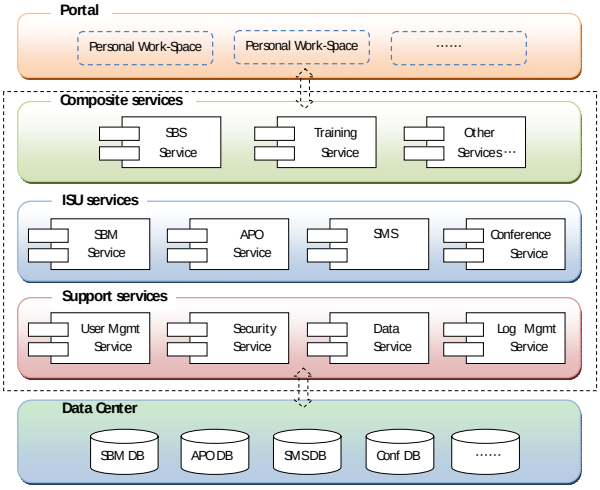



**Figure 1. Architecture of the ISU platform**

## 3. Establishing of Virtual Enterprises

The virtual enterprise (V.E.) concept is one of the most important ways to raise the agility and competitiveness of a manufacturing enterprise [5]. Under this concept a master company develops its products by using the manufacturing resources of external partners. In this paper, the creation of virtual enterprise is executed throw the platform.

All the stakeholders have to register to be users of the platform, if they want to use the interoperability service utilities to support their business activities.

In the ISU platform, there are mainly two kinds of enterprise users: assembly factory users and supplier users. The establishing of virtual enterprise is a prerequisite to use these ISUs. The assembly factory users have authorization to create a virtual enterprise. They can choose appropriate suppliers from all the registered users to supply the parts/materials they need for normal production. After the assembly facotory

gathering all the suppliers it needs, a virtual enterprise is established.

Then, master of the V.E. could apply for ISUs such as SBM service and APO service. After been approved by the platform, the master could use these ISUs with the other members of V.E. to establish interoperability easily and quickly.

It must be addressed that, there exists three different kinds of virtual enterprises in the platform, and the platform can serve all of them well.

The first kind of V.E. is the most common and standard one, as is shown in figure 2. In this kind of V.E., an assembly factory is the unique master, and a number of suppliers serve it. The relationship among them is very simple because there is no intersection in the organization.

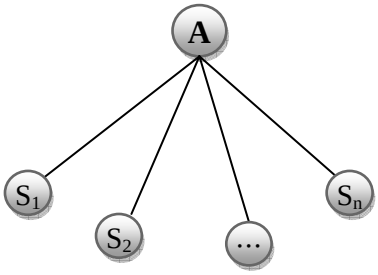



**Figure 2. Standard model of V.E.**

In the second kind of V.E., the relationship is more complicated, because intersections appear among V.E.'s boundaries. As is shown in figure 3, one supplier may serve several masters. As a result, they will exchange information with different assembly factories.

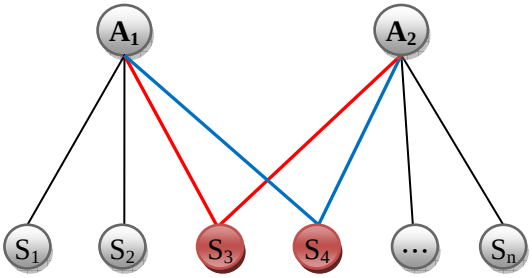



**Figure 3. Intersectional model of V.E.**

The third kind of V.E., is the most complicated one. In ISU platform, all the enterprise users could apply for ISU services, so it potentially allows a "double-role" situation. In other words, the master of a V.E., may be a supplier in another V.E.. As is shown in figure 4, $A_1$, $A_2$ are both masters in their organization of V.E., however, they are parts suppliers of $A_3$ at the same time. Just take an engine manufacture enterprise for example, it can be master of a V.E., and get outsourcing parts to produce engines; meanwhile, it may play the role of a supplier to serve other assembly factories.

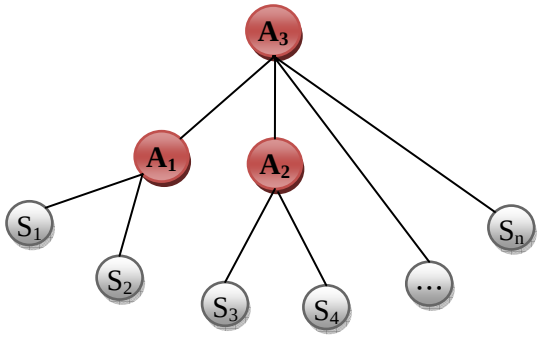



**Figure 4. Double-role model of V.E.**

According to the analysis of these possible cases, we design and develop the ISU platform in a flexible way to deal with various requirements.

## 4. SBM service

To build agile supply chain for automobile industry, a software named supply business management (SBM) was developed. SBM is an interoperability service utility which is delivered in the form of SaaS. The SaaS vision focuses on separating the possession and ownership of software from its use. Delivering software's functionality as a set of distributed services that can be configured and bound at delivery time can overcome many current limitations constraining software use, deployment, and evolution.

At present, the SBM service is widely used by many assembly factories that have their powerful inner software systems but have no information platform to share all the important data with their hundreds of suppliers.

### 4.1 Business scenario

An automobile manufacturer (named F) has its own CAPP software system which is used to support its daily business activities and improve efficiency of the supply chain management.

During execution, production department makes daily production plans, and these plans will be imported into CAPP system. Then CAPP system decomposes these plans and gets concrete parts/materials needed according to the BOM. Compared with the inventory, CAPP will generate daily procurement plans.

According to these procurement plans, workers in procurement department will call the corresponding suppliers to deliver parts/materials on time to ensure normal production. If some of these suppliers can't supply on time, the workers will get some alternative suppliers to fill the gap.

When suppliers deliver parts/materials to company F, they have to go to procurement department first to print procurement orders through CAPP system, and then deliver goods to warehouse together with the printed

orders. As a result, there is usually a long queue in the office of procurement department waiting to print procurement orders. When suppliers go to finance department to get some financial information, they also have to wait a long time.

In this scenario, efficiency of the supply chain management needs to be improved immediately, as all the information can be read from the inner software system, but they can't be shared with suppliers. In a word, company F needs an information platform to share all the important data with its hundreds of suppliers to enhance the interoperability with these stakeholders.

At last, company F tries to enhance efficiency of their supply chain management by importing the third-party SBM service to decrease its IT investment and operational cost, instead of developing a new solution from scratch.

## 4.2 Functions

SBM provides a cooperative environment of supply business for assembly factory and suppliers.

At the beginning, company F will use a data synchronization service to synchronize all the important data it wants to share with these suppliers. Data synchronization service is one of the support services provided in our platform, which is used to synchronize important data from inner systems of an enterprise to the ISU platform. During execution, we have to synchronize the production data to the database of SBM. This is a precondition to use SBM service.

With the help of SBM service, assembly factory can share production related information easily with its hundreds of suppliers. SBM provides several services to support the business activities, such as plan service, order service, finance service, quality service, notice service, inventory service etc.

Using the plan service, assembly factory could publish its procurement plans to suppliers to guide their production and shipping. There are mainly three kinds of purchase plans, including monthly plans, weekly plans, and daily plans.

Order service is the most important function in SBM. Assembly factory allocates orders to its suppliers according to the pre-defined quota standard. The decomposed orders are published immediately to corresponding suppliers. Then, the suppliers will print orders online and shipping the parts/materials directly to warehouse. They will never wait a long time in procurement department to print the orders.

Finance service publishes all the financial information to suppliers, such as general ledger arrearages, quality compensation etc.

Quality service gives all the quality related data, such as quality inspection, sampling inspection etc.

Inventory service allows suppliers to get the accurate inventory in assembly factory. According to the inventory, suppliers can get the information of already used materials, and carry out the distribution.

In order to serve all kinds of assembly factories, the SBM service is designed and realized flexibly, which provides the function of customization. That is to say, users of SBM could customize the service and use it in a DIY way.

## 4.3 Customization

In real world, business rules of automobile assembly factories are complicated and different from each other. Therefore, SBM service may not be suitable for all of them. So we provide the function of customization to deal with various conditions. All the customized information will be written in corresponding configure files in the form of XML.

Actually, there are three kinds of configure files in SBM service: UI config file, authorization config file and the business logic config file.

The logo and menus can be configured in the UI config file; roles of users and authorization of all kinds of operations are configured in the authorization config file. What's more, all the business rules are configured in the business logic config file. With these config files, an SBM instance can be customized by the master of a V.E.

The data flow of SBM service is shown in figure 5.

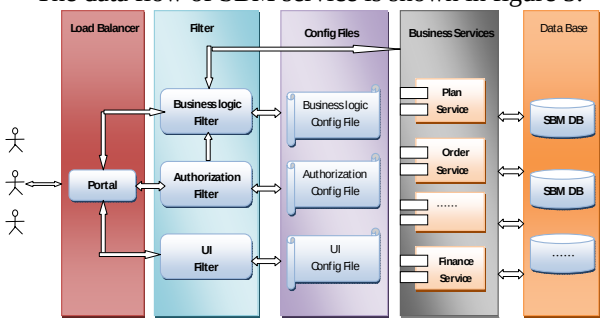



**Figure 5. Data flow of SBM service**

As is shown in figure 5, when supplier users request information from SBM service, they will first pass a load balancer which is used to deal with concurrent request from a large number of users.

Then, the filter layer is used to filter the requests with the help of UI filter, authorization filter and business logic filter. The UI filter is used to get the logo, menus and other UI configure details customized by the assembly factories. The authorization filter is used to identify whether the operation of the user is authorized in the authorization file. If the operation is authorized, the request will pass the business logic filter, to get business rules pre-defined by master of V.E. According to these business rules, corresponding business services will be invoked to get the needed information from the SBM data base.

It has to be addressed that, before the master of V.E. is approved to use SBM service, SLA (service level

agreements) should be contracted, such as the maximal concurrent users on line and the response time of querying 1 million records etc.

With the support of SBM service, assembly factory could enhance and improve interoperability with its suppliers with little IT investment and operational cost, instead of developing a new solution from scratch.

## 5. Interoperability between SBM and APO

In the ISU platform, SBM and APO are delivered in the form of SaaS. However, SBM and APO usually have to be used together to fulfill some special requirements. An interactive framework is proposed to establish the interoperability between two SaaS applications.

### 5.1 Improved business scenario

In the scenario of section 4, with the help of SBM service, company F could share the important information synchronously with its suppliers.

In SBM service, information flow is only from company F to its suppliers: company F publishes its procurement plans to its suppliers, who then determine whether they have enough capacities to satisfy such demands. However, company F knows nothing about the capacities of its suppliers. Actually, almost none of its suppliers have such kinds of software to support their calculations on production planning and scheduling, so they have to deal with such heavy tasks manually, which are not only time-consuming, but also lead to imprecise results.

To address such issues, company F plans to enhance efficiency of its supply chain management by compositing SBM and APO together to help suppliers automatically calculate their replenishment capacities. APO is a set of packaged services that provides strong capabilities for advanced production planning and optimization through explosion or implosion.

After the composition, suppliers can get the procurement plans through SBM service and directly invoke the APO service to calculate the production planning and optimization. What's more, with the help of APO, suppliers can feedback on line to the assembly factory whether they can supply the needed parts/materials on time.

In this improved scenario, the interoperability is established by the following steps:

1) Production department of company F makes daily production planning and imports these plans into the CAPP system;

2) Then CAPP system will decompose these plans and get concrete parts/materials needed according to the BOM. Compared with the inventory, CAPP will generate the daily procurement plans;

3) The suppliers get corresponding procurement plans published throw services in SBM, and then calculate the advanced planning and optimization through the APO service which is invoked in SBM;

4) APO returns the results to the suppliers, who then make decisions based on the implosion results and send a response to procurement department to feedback whether they can supply on time. Then the suppliers prepare to produce the required parts/materials;

5) The procurement department collects responses from all the suppliers. If there are some suppliers that can't completely fulfill the initial procurement plans, it will then try to find some alternative suppliers to fill the gap;

6) In some special situations where no alternative suppliers could be found, production department of company F has to adjust its production plans.

With the help of SBM service and APO service, suppliers can share the information of company F easily; moreover, they can calculate their production planning and optimization automatically based on the interoperability between SBM and APO.

### 5.2 Interactive framework

In order to invoke functions of APO directly in SBM, we have to expose some related functions to web services, as is shown in figure 6. To ensure the interoperability between SBM and APO, three technological issues have to be addressed:

1) How to embed the original web pages of APO into the pages of SBM, and directly use them to invoke the web services exposed by APO;

2) How to connect the web services exposed by SBM and APO to execute business process and get the needed data;

3) How to coordinate the web services exposed by SBM and APO;

To address such issues, this paper proposes an interactive framework to establish interoperability between SBM and APO. As is shown in figure 6, with this framework, the interoperability can be established in presentation layer, function layer, and data layer. And finally, the original APO service will act as a module of SBM. The users won't realize that their business activities are executed in a third-party application.

In the following sections, the paper will introduce solutions to the three technological issues mentioned above.

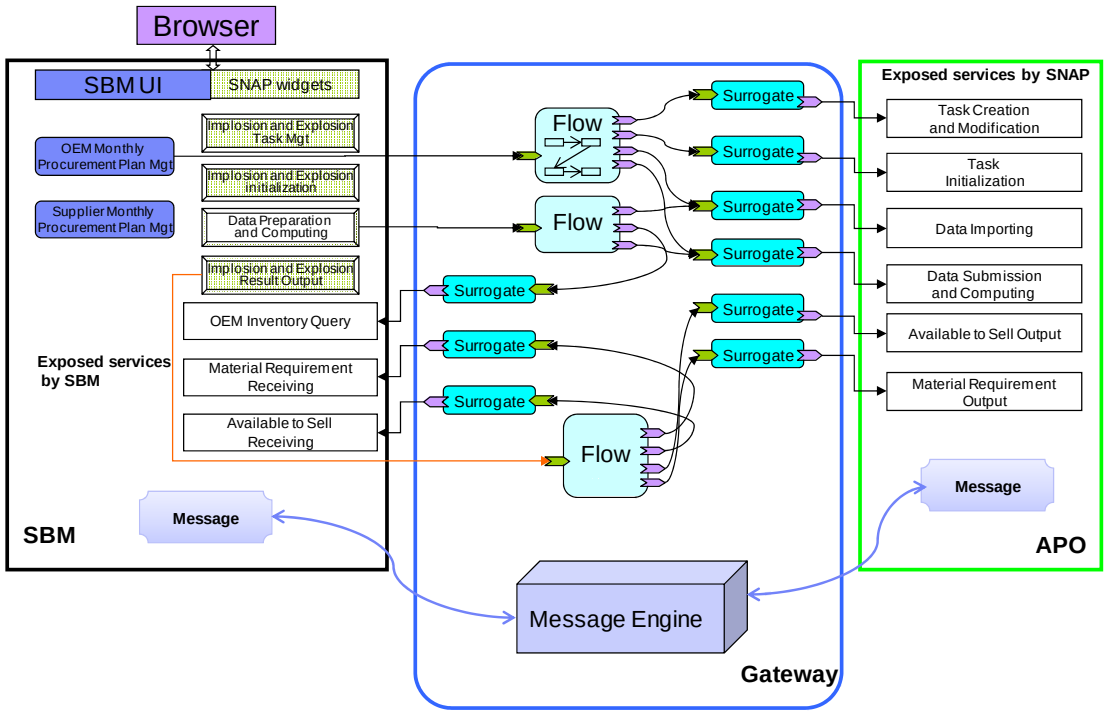



**Figure 6. Interactive framework**

## 5.3 Architecture of EWidget model

To establish the interoperability in presentation layer, an extended UI widget model—EWidget is designed. EWidget is a kind of utility which is used to access information and services. Widget is an instance of EWidget. Different widgets usually implement different functions, and can be composed together to construct a new application.

Figure 7 shows the architecture of EWidget model, which lies in both presentation layer and business logic layer. The size, layout and initial variables of a EWidget can be defined in its properties which are used in the exchanging of information. EWidget can support the invoking of remote services through DELEGATE. {op1, op2,⋯, opn} defines the operations which can be customized by application designers. EWidget model extends Dojo toolkit [6], which is written in JavaScript and XML.

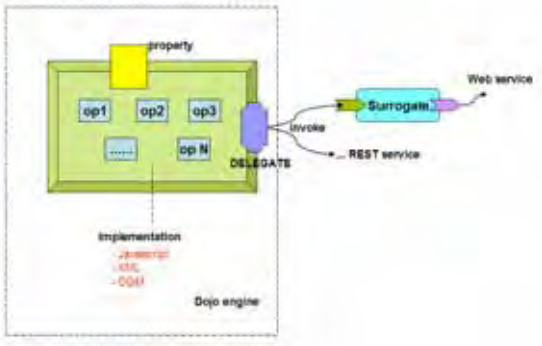



**Figure 7. Architecture of EWidget model**

In EWidget model, the invoking of remote services is needed to execute business process. However, most of the popular browsers, such as IE and Firefox, prohibit the cross-domain invoking of web services. Therefore, we import the DELEGATE module to solve this problem. According to the EWidget model, original web pages of APO could be converted into widgets that can be embedded into pages of SBM.

Each widget stands for a unique function. The whole process will be fulfilled by traversing from one widget to another according to the business rules. At the same time, each widget deals with a certain kind of data structure, and the business data will be transferred from one widget to another, from APO to SBM.

## 5.4 Service connection

Generally speaking, there are various services in a heterogeneous environment; web service is just one of them. How to identify them and make them work together is an issue need to be addressed. This paper provides solutions including service authentication, service authorization and service connection according to different business requirements.

Service surrogate extended from SCA [7] provides a programming model which is used to construct applications and solutions based on SOA. SCA (Service Component Architecture) is a model that aims to encompass a wide range of technologies for service components and for the access methods used to connect them.

Component is the basic unit of Service Component Architecture, as shown in Figure 8. Component is a paragraph of codes that provide special functions. As to access methods, SCA compositions support various communication and service access technologies such as web service, messaging system and RPC (Remote Procedure Call). The original definition of SCA Component defines the implementation, exposure and invoking mechanism of services. However, it cannot fulfill our requirements for interoperability, such as service authorization and service authentication.

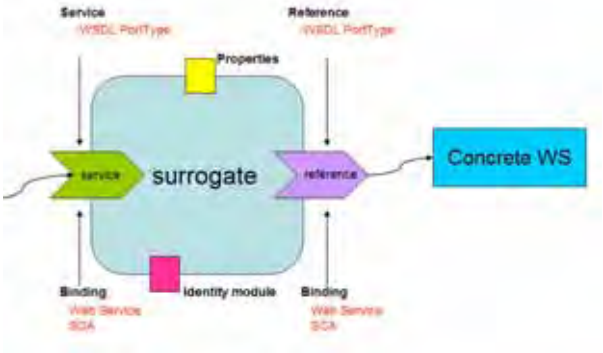



**Figure 8. Definition of surrogate based on SCA Component**

SCA provides service flow definition mechanism to finish complex business automatically. It is the same as the extended SCA component. SCA describes the content and linkage of an application in assemblies called composites. Composites can contain components, services, references, property declarations, plus the wiring that describes the connections between these elements. Composites can group and link components built from different implementation technologies, allowing appropriate technologies to be used for each business task [8].

This paper extends SCA Component by adding the Identity Module to monitor and process SOAP messages based on WS-Security. As is shown in Figure 8, the extended SCA Component is named "surrogate". In common situation, SBM and APO manage their users separately. So the detailed solution of service connection is:

1) Encapsulating the related web services with SCA Component.

2) Process the identity information in the identity module. A mapping table for users should be maintained in the ISU platform. The table stores the mapping relationship of SBM users and APO users. The role and authorization will not be changed and still be managed by SBM and APO. When an authorized user, such as Bob logs in SBM and invokes web services provided by APO, the authentication process will be activated. Firstly, Identity Module will insert Bob's identity information into SOAP head; then the gateway will capture the message and find the corresponding user of Bob in the mapping table. If succeed, the new user_id certified by APO will be inserted into SOAP head to invoke the services exposed by APO. The return process is the same.

## 5.5 Service coordination

Service coordination is often used to support a number of applications, including those that need to reach consistent agreement on the outcome of distributed transactions.

In order to ensure the interoperability between SBM service and APO service, this paper proposes a Message Notification based service coordination technology, which is driven by business rules. We design and implement a tool named Message Engine in dependence on Apache Kandula2 [9] project, which provides an open-source implementation of WS-Coordination, WS-Atomic Transaction and WS-Business Activity.

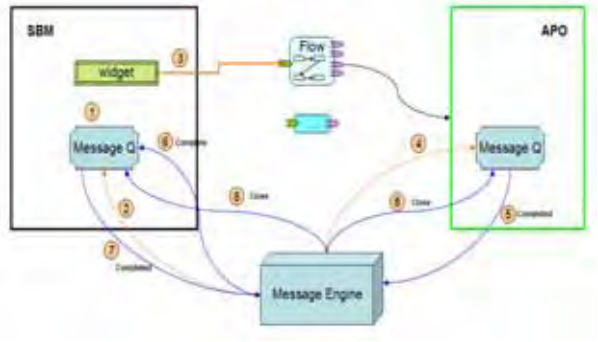



**Figure 9. Service Coordination based on Message Notification**

Message Engine not only enables an application service to create a context needed to propagate an activity to other services but also enables the coordination of transactions among web services.

As shown in Figure 9, with the help of the Message Engine, the whole process of service coordination based on Message Notification is carried out by the following steps:

1) Configuring the Message Q of SBM service and APO service;

2) Registering the Message Q of SBM to Message Engine;

3) Users log in SBM service and then activate a business process;

4) Web services involved in the process are invoked and Message Q of APO is registered to Message Engine;

5) Message Q in APO invokes the completed () function automatically after the execution;

6) Message Engine invokes the complete () function of SBM to notify the end of the process;

7) On receiving the notification, the whole process ends;

8) Notifying APO the whole process ends.

With the help of these technologies, interoperability between SBM and APO can be established to fulfill the

requirement of making them work together to enhance the efficiency of supply chain management.

## 6. Application of the ISU platform

The ISU platform was named SDMSP in real world. At present, there are more than 600 enterprises users in SDMSP platform. They are using the ISUs provided by our platform to establish interoperability with their partners and enhance efficiency of their business activities. More importantly, the SDMSP platform facilitates SMEs' participation to collaborative supply chain management processes.

## 7. Conclusion and future work

This paper proposed a methodology that provides a guide on how to establish interoperability between enterprises through a federated approach. Under the guide of this methodology, the paper designed and realized an ISU platform to enhance the interoperability and efficiency of the supply chain management in automobile industry.

The paper introduced the architecture of ISU platform and the application of SBM service and APO service. Moreover, the paper proposed an interactive framework to establish the interoperability between SBM service and APO service.

However, there are only several interoperability service utilities in the platform and not enough to support the whole process of supply chain management. In the future, we will design and develop more ISUs for automobile industry. What's more, there are many grand challenges in the research of enterprise interoperability, such as "Future Internet and Enterprise Systems", "Knowledge-Oriented Collaboration and Semantic Interoperability" etc. We will focus on these key issues.

## 8. Acknowledgement

The authors would like to acknowledge the support provided for the project by the Science &Technology Development Projects of Shandong Province (2006GG1104045), the National Natural Science Foundation of China(90612021, 60703027), the National High Technology Research and Development Program of China(No.2006AA01A113), the National Key Technologies R&D Program(2006BAF01A24) and the program for Changjiang Scholars and Innovative Research Team in University.

## References

[1] Yannis Charalabidis, George Gionis, Karl Moritz Hermann, Cristina Martinez; "Enterprise Interoperability Research Roadmap";Feb.2008

[2] Tarokh, M. J.; Ghahremanloo, Hoda; Karami, Mahnaz, Agility in Auto Dealers SCM, Service Operations and Logistics, and Informatics, 2007. SOLI 2007. IEEE International Conference on Volume, Issue , 27-29 Aug. 2007 Page(s):1 – 6

[3] Ross, A. Creating agile supply chains, Manufacturing Engineer, Dec. 2003, Volume: 82, Issue: 6, pp: 18-21

[4] Abhijit Dubey and Dilip Wagle, "Delivering software as a service", The McKinsey Quarterly, Web exclusive, May 2007

[5] D. Wang, K.L. Yung, W.H. Ip, "A heuristic genetic algorithm for subcontractor selection in a global manufacturing environment", IEEE Transactions on Systems, Man and Cybernetics—Part C: Applications and Reviews, 2001, 31 (2), pp.189–198.

[6] http://dojotoolkit.org/

[7] SCA Service Component Architecture – Assembly ModelSpecification.http://www.osoa.org/display/Main/Service+Component+Architecture+Home

[8] http://incubator.apache.org/tuscany/

[9] http://ws.apache.org/kandula/2/index.html

# Enterprise Interoperability with SOA: a Survey of Service Composition Approaches

Rodrigo Mantovaneli Pessoa[1], Eduardo Silva[1], Marten van Sinderen[1],
Dick A. C. Quartel[2], Luís Ferreira Pires[1]
[1] *University of Twente, Enschede, The Netherlands*
[2] *Telematica Instituut, Enschede, The Netherlands*
*{mantovanelir, e.m.g.silva, m.j.vansinderen, l.ferreirapires}@ewi.utwente.nl*
*{Dick.Quartel}@telin.nl*

## Abstract

*Service-Oriented Architecture (SOA) claims to facilitate the construction of flexible and loosely coupled business applications, and therefore is seen as an enabling factor for enterprise interoperability. The concept of service, which is central to SOA, is very convenient to address the matching of needs and capabilities in enterprise collaborations. In order to satisfy more demanding needs or to rapidly adapt to changing needs it is possible to perform service composition in order to combine the capabilities provided through several available services. This paper presents a survey on recent approaches for service composition. To perform this study a conceptual framework for service composition is proposed. This framework allows studying how different approaches deal with the service composition life-cycle and provides basic guidelines for their analysis, evaluation and comparison. The proposed framework is used to analyse five representative service composition approaches.*

## 1. Introduction

One promising benefit of Service-Oriented Architecture (SOA) is to facilitate the construction of flexible and loosely coupled business applications. SOA-based businesses applications can span several networked enterprises, with services that encapsulate and externalize various corporate applications and data collections. Service composition is an essential ingredient of SOA, as it is concerned with aggregating interoperable services such that the goals of (enterprises in) a collaboration endeavour can be satisfied. Many individual service composition approaches and solutions have been proposed and developed in recent years. However, more effort has to be spent on their evaluation and comparison. In this paper we investigate recent approaches and technologies to support service composition.

Traditionally, research surveys on service composition tend to either focus on one particular emerging technology (such as workflow-based, AI planning-based, ontology-based, etc) [1, 2, 3, 4] or be domain-specific [5, 6]. In this paper, we adopt a different approach to this classification and organize our study around the concept of *service composition* life-cycle. We define the different phases of the service composition life-cycle, and based on this we create a comparison framework. The proposed framework establishes a set of evaluation criteria that provides basic guidelines for analysis, evaluation and comparison. We argue that our framework enables a more comprehensive understanding of existing service composition approaches, and allows us to recognize opportunities for combining approaches and identifying open issues and research challenges.

The remaining of this paper is structured as follows: Section 2 discusses enterprise interoperability issues related to service composition. Section 3 describes the service composition life-cycle. Section 4 introduces our comparison framework. Section 5 describes five different service composition approaches. Section 6 compares the studied approaches. Finally, Section 7 presents our conclusions and defines some future research directions.

## 2. Enterprise Interoperability and Service Composition

Enterprise interoperability denotes the ability of organizational entities (businesses, government, companies and parts thereof), hereafter called *enterprises*, to interoperate in order to achieve certain business goals [7]. Since enterprises are subject to constant internal changes and must continuously react to ongoing or imminent changes in markets and trading partners, interoperability solutions cannot be static. In addition, since enterprises increasingly use ICT to support their business activities, interoperability solutions cannot be restricted to the organizational level alone. Therefore, in order to develop practical solutions for modern enterprises, interoperability should be addressed both from organizational and technical points of view, and flexibility (next to other "ilities") should be a major concern.

The following aspects of interoperability have been distinguished [8]: (i) businesses; (ii) processes; (iii) services; and (iv) data interoperability. Businesses and processes interoperability are considered mainly at the organizational level, whereas services and data interoperability require focus on (information) technology issues. The term service can be used to denote a business function as well as a function of a computer-based application. In the context of this paper, we limit ourselves to the latter denotation, however, being aware that a comprehensive treatment of enterprise interoperability would require consideration of both denotations within a single framework.

It has now been widely recognized that SOA can bring significant benefits for enterprise interoperability [9]. A perceived value of SOA is that the concept of service, which is central to SOA, allows one to address the matching of needs and capabilities in enterprise collaborations at the proper level of abstraction. SOA is based on the assumption that enterprise systems may be under the control of different ownership domains. Therefore, the focus is on services that these systems can provide (capabilities) or that these systems want to use (needs), and ownership issues are not visible except for restrictions imposed on the use of these services. Services are self-contained units of functionality, which are described, published and discovered [10]. These properties form the basis for fulfilling the desirable flexibility mentioned above: service providers can make themselves and their services instantly known. Moreover, user needs can also be matched to a combination of multiple provider capabilities, corresponding to multiple services, through a mechanism where discovered services are composed with the aim of fully satisfying the user needs.

This brings us to the focus of this paper, namely *service composition*. We perform a survey of different service composition approaches, analysing them according to the phases of the service composition life-cycle.

## 3. Composite Service Life-cycle

A composite service consists of a composition of existing services to achieve some functionality that typically is not provided by a single available service. The composite service life-cycle provides an integrated view of the phases and artefacts produced for service development and execution. The phases cover design-time and runtime aspects of the composite service life-cycle and allow services to be created, operated and maintained. The life-cycle phases may vary according to the granularity chosen for the description of the different activities involved in the service composition process. Figure 1 shows the life-cycle we propose for the development and execution of composite services.

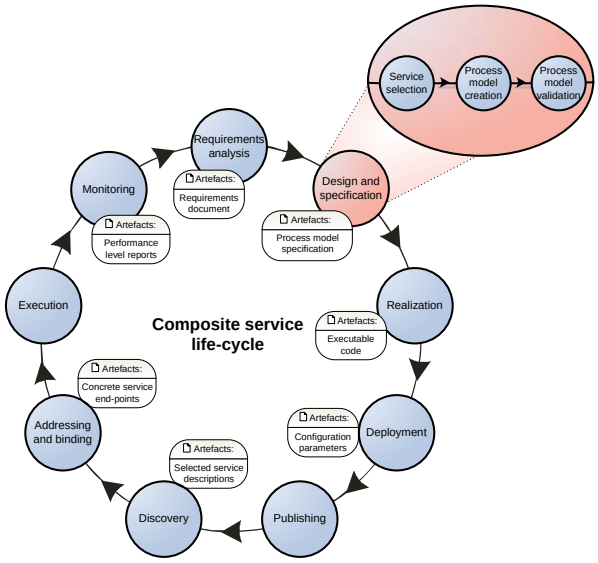



**Figure 1. Composite service life-cycle**

We have aimed at specifying the common phases and artefacts of a service composition process in a single and comprehensive model. The circles in Figure 1 represent phases of the composite service life-cycle, and the round rectangles represent the artefacts produced in each phase.

***Requirements analysis.*** The first phase of the life-cycle identifies and prioritizes business and customers requirements. In this phase the scope is set, resources are planned, and the business context in which the new services will operate is determined. Possible needs for new services are identified based on an analysis of

business and customers requirements. The artefact produced in the requirements analysis phase is a software requirements document, which typically details the system's functional and non-functional requirements in a structured form.

***Design and specification.*** In this phase, the composite service is designed to fulfil the business and customers requirements identified in the previous phase. In this phase the services needed to realize the requested service composition are selected and a process model specifying how the services are coordinated is built and validated. This involves the following sub-phases:

- ***Service selection*:** performs the selection and ranking of suitable services to fulfil the composition requirements. The output of the selection and ranking algorithm is the list of services that fulfil all functional and quality requirements of the user, ordered by some criteria.
- ***Process model creation*:** relates to the creation of a service composition. We should be able to determine what component services are executed and by whom, at what moment in time, what are the components dependencies and what are the expected results. The process model can be specified in terms of a single party view or a multi-party view. The first is usually defined as a service orchestration, while the second is defined as a service choreography. The resulting artefact is a specification of how to coordinate the discovered and selected services to meet the user needs. This specification can be produced at design-time or runtime, using manual, semi-automated or automated composition methods.
- ***Process model validation*:** services may have the same or similar functionality. Therefore, it is possible that more than one composite service is generated to fulfil the service requirements. In this case, the composite services should be evaluated and ranked on their overall utility. This evaluation is usually performed based on the composite service functional and non-functional properties. The most commonly used evaluation method is by using utility functions that rank the created composite services according to weights specified by the service requester for each non-functional property. The validation of the composite service correctness is another aspect of the service composition. It consists of verifying if the requested requirements are achieved by the composite service, if there are no deadlocks, etc.

***Realization.*** This phase focuses on the service technical implementation details. In this phase the service identified and designed in the previous phase is built and tested. The service implementation can be coded in different computer languages and the source code can be obtained through actual programming, wrapping existing legacy applications or by model-to-code transformations. The materialization of a service is completed when some executable (code) specification is produced as artefact.

***Deployment.*** The service deployment phase addresses the problem of installing, configuring and managing services and service instances in the service execution environment.

***Publishing.*** In this phase the service description information is published. Usually this information is specified in a service description document. The published information allows one to advertise the service, and so that the service can be discovered by potential consumers. The service description document describes what the service does, where it can be found, and how it can be invoked.

***Discovery.*** The discovery phase concerns the service consumers' ability to find (either at design-time or at runtime) the service descriptions published in service registries in the Publishing phase. Once the services are published in the registry, users can search and find the services that meet their requirements.

***Service binding.*** A service may have different implementations, and each implementation may have multiple deployments in different service end-points. The service binding phase focuses on how service endpoints are discovered and instantiated. The binding phase may be performed either at design-time or at runtime, and can be static or dynamic. Static service bindings are defined at design-time and define a tightly coupled interaction between service user and service provider. Dynamic service bindings allow a dynamic binding of service user and service provider's service at runtime, given a selection and discovery mechanism, usually defined at design-time.

***Execution.*** The execution phase involves the invocation of all participating services, possibly hosted in different provider domains. The execution of a composite service must be consistent with the specified process model and for this, a coordination mechanism is required. During composite service execution, this coordination mechanism is responsible for invoking the participating services, receiving notifications of completion from each participating service, transferring and transforming (when required) the input/output parameters among the participating services, and evaluating pre-conditions that must be satisfied prior to the invocation of a participating service and post-processing actions that must be

performed after the execution of a participating service.

***Monitoring.*** In this phase, the service provider constantly monitors the composite service execution, evaluating its performance and verifying if the agreed performance levels are met. The goal of the service monitoring phase is to rapidly respond to indications of service degradation or failure, and to ensure that the service level agreements are fulfilled. To achieve these objectives, in general, service monitoring requires a set of QoS metrics to be gathered and interpreted.

## 4. Comparison Framework

In this section, we present our framework for analysis and comparison of service composition approaches. The proposed framework is derived from our service life-cycle. We specially focus on the service discovery and composition phases, or *Process model creation*, but also include other phases, such as process model verification and service execution. For each considered life-cycle phase we developed a set of evaluation criteria. Based on the framework, one may evaluate different service composition approaches, having a common ground to compare them. The characterization of service composition approaches is generally based on literature review. Table 1 presents our framework.

**Table 1. Comparison framework**

| Life-cycle Phase | Evaluation Criterion |
|---|---|
| Service Discovery | Service description |
| | Service matching and selection |
| Process model creation | Behaviour specification |
| | Information specification |
| | Level of automation |
| | Composition time |
| | Coordination distribution |
| Process model verification | Composition correctness |
| Execution | Service binding |

In the following, we present in more detail each of the considered evaluation criterion of our framework:

***Service description:*** aims at evaluating how a service is described. Different aspects can be considered, but we may reduce them to two groups: functional and non-functional aspects. The service functional description focus on the functionality supported by the service, which is usually expressed in terms of inputs, outputs, preconditions and effects (IOPEs). The non-functional service description aspects describe other properties the service such as: performance, availability, cost, etc.

***Service matching and selection:*** refers to the task of discovering and selecting services that can be used in the composition. Services are discovered based on a set of requirements specified in a service request. The discovery and matching may be based on different information, such as service goals, or IOPEs. At the end of the service discovery process a list of services are retrieved. This set can be further filtered based on selection criteria. Different approaches may be taken to perform service selection. For example, non-functional properties, such as service cost, may be used to rank candidate services.

***Behaviour specification:*** deals with the design of composite service behaviour. Here we focus on describing the languages and formalisms used by approaches for modelling the behaviour of a service composition (combined behaviour of the component services) and for defining constraints between the service operations that determine allowed invocation orders.

***Information specification:*** since services handle data in the form of input and output parameters, it is necessary to model the data (types) that a service can handle, the data flow of the composite service and possible data transformation between the component services of a composite service.

***Level of automation:*** given a set of available services and a user service request description, the problem of service composition synthesis is concerned with the creation of a new composite service, thus producing a specification of how to coordinate the available services to realize the client request. Such a specification can be obtained either automatically, i.e., using a tool that implements a composition algorithm, semi-automatically, i.e., in case the user makes choices during the composition phase aided by an interactive tool, or manually by the user. Depending on the intended purpose of a given approach, different (and specific) requirements may arise and be imposed.

***Composition time:*** refers to the moment at which the approaches perform the service composition synthesis. Two distinct moments may exist, namely design-time and runtime. However, an initial composition plan can be defined at design-time, which can be adapted dynamically at runtime.

***Coordination distribution:*** the coordination of a composite service requires that the service is completely specified, in terms of both the specification of how various services are linked, and the internal process model of the composite Two main kinds of coordination have been identified in [11]:

- ***Centralised:*** centralised coordination is based on a hub-and-spoke topology, in which one service is given the role of process mediator/delegator, and all the interactions pass through such a service. This mechanism is usually defined as orchestration.
- ***Peer-to-peer:*** in decentralized coordination, there are multiple coordination entities, placed at distributed locations, each executing a composite service specification (which is a portion of the original composite service specification). The coordination entities communicate directly with each other rather than through a central coordinator, in order to transfer data and control when necessary in an asynchronous way. This mechanism is usually defined as choreography.

***Composition correctness:*** refers to the capability of checking the correctness and reliability of the composite service with respect to the service requirements. Composition correctness requires verification of the composed service's properties, such as reachability, liveness or safety.

***Service binding:*** in order to enhance the flexibility of a composition, services are usually not hard-coded into the composition model but bound into it at different times (i.e., runtime and design time). During execution, a composition engine has to target messages to specific services, which are defined in the composition schema. The service selection model deals with static and dynamic binding, i.e., how a service is selected and bound statically at design-time or dynamically at runtime. Alonso et al. [12] describe four different service binding models:

- Static binding: service endpoint URL is hard-coded;
- Dynamic binding by reference: service URL is computed and stored into a variable;
- Dynamic binding by lookup: before each service invocation a query is sent to a registry to locate a suitable implementation;
- Dynamic operation selection: no assumptions are made about the signature of the arbitrary service to be invoked.

## 5. Service Composition Approaches

Recently, several techniques and methodologies for modelling and specifying different aspects of service composition have been proposed. In this section, we discuss some representative approaches with respect to the evaluation criteria defined in our framework, in order to compare them.

### 5.1. METEOR-S

The METEOR (Managing End-To-End OpeRations) project at the Large Scale Distributed Information Systems (LSDIS) Lab at the University of Georgia focused on workflow management techniques for large-scale transactional workflows. Its follow-up project, which incorporates workflow management for semantic web services, is called METEOR-S (METEOR for Semantic web services) [13]. A key feature in this project is the usage of semantics for the complete life-cycle of semantic web services. Its annotation framework is an approach to add semantics to current industry standards such as WSDL. Finding an appropriate service for the composition is realized by a discovery engine that queries an enhanced UDDI registry.

***Service description.*** The service descriptions are semantically augmented, which resulted on the SAWSDL (Semantic Annotations for WSDL) language. SAWSDL is based on WSDL-S, which was a joint specification developed by IBM and LSDIS Lab for adding semantic annotation to WSDL. SAWSDL is a simple extension of WSDL using the extensibility elements. It provides a mechanism to annotate the capabilities and requirements of web services (described using WSDL) with semantic concepts defined in an external domain model (e.g., ontology). Externalizing the domain models allows SAWSDL to take an agnostic view towards semantic representation languages. This allows developers to build domain models in their preferred language or reuse existing domain models. It has two basic types of annotations: the model reference and the schema mapping. Additionally, the METEOR-S framework extends the SAWSDL annotations with preconditions and effects, used to describe the conditions that must be met before an operation can be invoked and the result that the invocation of the operation will have.

The approach uses QoS ontologies to represent the semantics of service non-functional properties and has described generic QoS metrics based on four dimensions: time, cost, reliability, and fidelity. Each metric specification consists of a quadruple. *QoSq(s,o) = <name, comparisonOp, val, unit>*, where *'name'* is the parameter name, *'comparisonOp'* is a comparison operator, *'val'* is a numerical value, and *'unit'* is the metric unit. The approach also presents a mathematical model that formally describes the formulae to compute QoS metrics among workflow tasks and an algorithm to automatically compute the overall QoS of a workflow.

***Service matching and selection.*** METEOR-S has developed a three-phase algorithm for service selection that requires the users to enter service requirements as

templates constructed using ontological concepts. In the first phase, the algorithm matches services (operations in different WSDL files) based on the functionality they provide. In the second phase, the result set from the first phase is ranked based on semantic similarity between the input and output concepts of the selected operations and the input and output concepts of the template, respectively. The optional third phase involves ranking based on the semantic similarity between the precondition and effect concepts of the selected operations and preconditions and effect concepts of the template. The semantic matching on the semantic template of the activity is done against the operations, inputs, outputs, preconditions and effects of the services available. The ranking on semantic matching is based on the weights assigned by the process creator to the individual semantic parts of the activity, namely operations, inputs, outputs, preconditions and effects. The assigned weights are normalized before calculation

***Information specification.*** The information specification is based on manual specification of the data semantics. The model reference annotation is used to specify the association between a WSDL element and a concept in some semantic model (ontology). The schema mapping annotations are used by the METEOR-S framework to deal with further mismatches in the structure of the inputs and outputs of the web services, particularly transforming one data representation into another, such that it can be used in another web service. Mappings are created between the web service message element and the ontology concept with which the message element is semantically associated. In addition to a mapping from the web service message element to the ontology concept, also called the *liftingSchemaMapping*, an additional mapping from the ontology concept to the message element, called the *loweringSchemaMapping*, is specified. Once the mappings are defined, two web services can interoperate by reusing these mappings and the ontologies now become a vehicle through which web services resolve their message level heterogeneity.

***Behaviour specification.*** METEOR-S specifies the process model describing the behaviour of services by capturing semantics of the activities in the process template during the design phase. The activities are not bound to web service implementations, but defined using semantic descriptions. Such templates are independent of the service description and process definition standards. The process template is a collection of activities, which can be linked using control flow constructs. The process templates in METEOR-S have a BPEL-like syntax. For representing control flow, the template uses the BPEL constructs. The template has also some additional constructs, like invoke activity, criteria, semantic-spec, discovery-spec, etc. which are prescribed in the BPEL specification, i.e., they are METEOR-S specific constructs independent of any process specification standard that can be used to generate executable processes.

***Level of automation.*** The development module provides a GUI-based tool for creating semantic web services using SAWSDL. The tool provides support for semi-automatic and manual annotation of existing web services or source code with concepts from domain ontologies.

***Composition time.*** METEOR-S offers support for two types of service composition, namely Static Composition (services to be composed are decided at design-time, static binding) and Dynamic Composition (services to be composed are decided at runtime, dynamic binding).

***Coordination distribution.*** The coordination of the composite service is based on a BPEL-like centralised process engine, or an orchestration.

***Composition correctness.*** The constraint analysis and optimization sub-modules deal with correctness and optimization of the process based on quality of service constraints. There is also support for state machine based verification of BPEL process.

***Service binding.*** The current prototype supports three kinds of service binding: static binding, deployment-time binding and dynamic binding. In static binding, a set of services is permanently bound to the composition. Deployment-time and runtime binding are achieved by using a proxy-based approach to bind a set of services that realise the service composition. In deployment-time binding, configuration is performed before the process starts executing. In runtime binding, configuration is performed after the process starts executing. Both deployment-time and runtime binding support reconfiguration. The configuration module has the ability to change the service bound to the proxies by simply changing a field in a shared data structure. This data structure is synchronised and accessed by each proxy before each service invocation. During reconfiguration, the process manager locks the data structure, thus making all proxies wait while the process is being reconfigured..

## 5.2. SODIUM

SODIUM (Service-Oriented Development In a Unified fraMework) was an international project, involving research, technological and industrial partners, dedicated to tackling interoperability

challenges that companies face at the data, services and business levels [14]. The project has developed a Generic Service Model, containing the common concepts of heterogeneous services from multiple points of view. The special characteristics of individual service technologies (such as Web services, Grid services or P2P services) are then dealt with as extensions to the core.

The SODIUM methodology adopts a model-driven and iterative approach for service composition and evolves in four phases: the user starts by defining the details of the complex task at a high abstraction level (phase 1); the user uses this abstract description to generate queries used for service discovery (phase 2); the discovered services are used to populate each of the abstract process tasks, hence transforming it to a concrete process description, and both the abstract and concrete descriptions are stored in the service composition (phase 3); and the concrete process description is transformed into executable descriptions and publishable documents about the composite service which has been built (phase 4).

***Service description.*** The service is represented in a UML model, according to an UML profile that can be used to model semantic aspects of web services. Activities are stereotyped in order to represent web service operations. Parameters of this activity element represent its inputs and outputs. A web service activity has a set of tagged values. The web service provider is defined by the tagged value *provider*. The URL to the WSDL file is registered in the tagged value *wsdl*. The exact service operation to invoke is given by the three tagged values *service*, *portType* and *operation*. To represent a p2p service operation, activities are stereotyped as *P2PServiceOperation*. The five tagged values type defined for a peer-to-peer service operation are *PSDL*, *Operation*, *Service* and *Pipe*. To represent a grid service operation, activities are stereotyped as a *GridServiceOperation*. The six tagged values type defined for a grid service operation are *gwsdl*, *ServiceLocation*, *Service*, *Operation*, *PortType* and *ResourceInstance*. The service may optionally have semantic references for the data types used and QoS offered, which are described depending on what kind of information we can retrieve about the service.

The approach makes use of OMG's QoS profile to represent collections of QoS properties with precise semantic meaning in the UML service model. Each QoS property contains a set of QoS dimensions with a name, its allowed value domain, an ordering function (whether higher or lower values are considered better) and its relationship to other QoS properties. The ordering direction of a property is defined as either increasing or decreasing, where increasing means that higher values are preferred. All the QoS properties to be used elsewhere shall be defined as a QoS property either within the model itself or as in imported model. Since SODIUM adopts the OMG profile, it already has generic capabilities to define QoS ontologies. In this sense, SODIUM does not suggest any specific QoS dimensions as part of the language, but is capable of defining any needed QoS dimension.

***Service matching and selection.*** The Behavioural Service Discovery Framework (BSDF) presented a novel approach in service discovery, which enables modelling queries with behavioural constraints in a visual manner. It comprises three main components: (i) a visual query modeller that models behavioural service queries by means of UML behavioural diagrams; (ii) a translator that transforms the raw XMI output of the modeller to a generic XML-based query language, namely USQL; and (iii) a query engine capable of processing and executing USQL queries in various types of target registries, repositories, networks, etc.

The framework is able to match the query against various types of service choreography advertisements, independently of their format and protocols. The basic idea of the USQL engine lies in the logical grouping of heterogeneous registries, depending on the domain their advertised services belong to. Having the registries organized in this way, the engine sets the service requestor one step closer to his/her specific requirements and narrows the range of the returning results, making them more relevant to the initial request.

***Information specification.*** Data objects are used to represent the data content that is created in the composition and that may be passed along to different activities. A data object has a specific *ObjectType* (optional) and may also represent the input and output parameters of the whole composition.

If the outputs of one service do not perfectly match the required input of the next service, there is a need to introduce intermediate data transformation steps between the services. This requires manual adjustments by the developer when specifying transformation nodes for defining data transformations as expressions in QVT (Queries/Views/Transformations). The transformation node is used for one-to-many, many-to-one and many-to-many data transformations. In a many-to-one transformation, the information from many source data objects is used to produce the content of a single target data object.

***Behaviour specification.*** The Visual Service Composition Language (VSCL) is a graphical composition language for defining service

compositions containing heterogeneous (web, grid, p2p) services. The main concepts of the languages are the tasks and the flow of data and control between tasks. The task-graph node consists of a task part and an entire sub-graph. Thus, the language has a construct that can be repeated at arbitrary levels to create a recursive decomposition structure. A task may be executed by different kinds of services, namely P2P, Web or Grid services. A service composition consists of Nodes and Flows/Edges. The Nodes are *TaskNodes* , which represent the invocation of a remote service, *ControlNodes*, which represent specific crossings for control flow, *ObjectNodes*, which are used for data transfer between the tasks, *EventNodes*, which represent an expression node that passes control through its outgoing arcs upon the occurrence of a predetermined event or *TransformationNodes*, which are used for defining data transformations. Two different kinds of Flow are used to specify flow of control and data between nodes. A Flow indicates a directed flow of either flow of control (*ControlFlow* or *EventFlow*) or flow of data objects (*ObjectFlow*). A Flow has one source node and one target node.

***Level of automation.*** The aim of the SODIUM approach is semi-automated tool support for service composition. In order to do so, the approach has automated large parts of the steps needed in the process of developing composite services. Many of the proposed steps for automation are model-driven transformations that transform between models and lexical descriptions about the services, both forward and reverse engineering.

***Composition time.*** Though runtime service selection is discussed, the primary focus has been design-time service discovery and composition.

***Coordination distribution.*** The coordination distribution runs on a central execution engine. This component deals with the execution of compositions by invoking services and orchestrating the control and data flow across the different steps of the composition.

***Composition correctness.*** The analyser is a component used for validating a composition against a set of rules. This allows one to check the model for syntactical errors. A dialog box allows the user to select which rules to validate, to execute and present the analysis result. Rules can be defined for each of the main classes in the model. Some constraints should be checked during editing, while others should be checked before the generation of lexical executable representation of the composition. However, no formal proof of correctness is given.

***Service binding.*** SODIUM refines the concept of binding beyond the basic distinction of static and dynamic binding. Service binding can be defined at the design, the compilation, the deployment, the beginning of the execution of a composition, or just before the actual service invocation takes place.

### 5.3. MoSCoE

MoSCoE (Modeling Web Service Composition and Execution) [15] is a project coordinated by Iowa State University. MoSCoE aims at the creation of a framework for modelling service composition and execution. The composition process in MoSCoE is divided in three-steps: abstraction, composition and refinement. Abstraction is provided to the framework users, allowing them to request a service using a high-level specification. The service providers advertise their services using common standard service description languages, namely OWL-S and WSDL descriptions. Given a service request, the framework's composition engine creates a suitable composition, from the existing services, if possible; otherwise it starts the refinement phase, guiding the user through a service request refinement procedure to create a service composition. The refinement process is iterative, stopping when a suitable service composition is found, or when the user decides to end the composition process. If a service composition is obtained, it is translated into a concrete BPEL workflow, which can be executed. The service composition defines rules for non-functional properties. At the execution time these rules are monitored, and in case of some specified event takes place, the appropriate actions are taken. Furthermore, while executing the service composition, various data and control flow transformations are carried out by referring to the pre-defined ontologies used in the service descriptions, and to specified inter-ontology mappings.

***Service description.*** Existing services are represented in OWL-S and WSDL specifications. OWL-S is used to semantically describe existing services, and specifying functional and non-functional aspects of the services, mainly for discovery. The MoSCoE project claims that other languages for service description can be supported. In fact, the framework translates all the service request and service descriptions to State Machine representations. This means that the service composition process is independent of the service description languages.

***Service matching and selection.*** MoSCoE performs service discovery based on semantic descriptions. It assumes that existing services are semantically described in OWL-S. Services are discovered based on the specified user request functional aspects, the so called IOPEs (Inputs, Outputs, Preconditions and Effects). The service request is specified in a visual

form, using a UML State Machines representation of the desired service. This information is interpreted to perform service discovery. The discovered services are organised in terms of degree of *semantic match*, which can be *Exact*, *Plug-in*, *Subsumption*, *Intersection* or *Disjoint*. The degree of match is computed through semantic reasoning on the requested properties and the existing services' semantic descriptions. Services that have intersection and disjoint matches are not considered as valid matches.

From the set of valid matches, a selection based on non-functional properties is performed. Non-functional properties are also described in an ontology. MoSCoE defines a quality vector which allows the user to specify which non-functional properties are of interest. Based on the quality vector, a quality matrix can be constructed, where lines represent the quality vector and columns the candidate services values for the considered non-functional properties. Furthermore, it is possible to define weight values for the different considered non-functional properties, which allows computing an additive value function, and consequently selecting the best suited service.

***Information specification.*** MoSCoE allows one to semantically describe services, in terms of both functional and non-functional properties. However, the necessary supporting ontologies may be defined by different parties. This may cause problems of semantic interoperability of the services used in a composition.

***Behaviour specification.*** In MoSCoE, service requests are specified using a UML State Machine representation. The discovered services are also translated to a State Machine representation. However, the MoSCoE service composition is described using a Transition System representation, more specifically, a Symbolic Transition System (STS). Therefore, transformations have to be performed on the service request and candidate services, to obtain Symbolic Transition Systems from state machine representations.

Once the candidate services are translated to STS, they are combined to reach the service goal specified by the user. This composition is obtained automatically, consisting on sequential and/or parallel compositions of STS service representations to meet the service goal specified by the user. If a composition is possible, the framework proceeds with the translation of the resulting STS representation to executable code, namely BPEL executable code. In case no composition is possible, the *Refinement* phase is triggered.

The Refinement phase consists of performing a new iteration with the user, asking for a service request refinement (using the UML State Machine representation). At the moment a refinement request is issued, concrete information about the problems/issues encountered during the service composition creation is provided. Given this, the user is guided on the refinement process, being asked to give more detailed information on concrete problems found on the service composition. After delivering the more detailed information, the UML State Machine is interpreted again, a new service discovery is performed, and a new set of services is retrieved. The service request and the set of discovered services are translated to State Machines and to STSs. Based on the STSs, the framework performs a new attempt to create a service composition that matches the *refined* user service request. If this is possible, the framework stops the service composition process, generating the executable code; otherwise it asks the user for a new refinement. This cycle may happen indefinitely if no service composition is constructed, unless the user decides to stop the process.

***Level of automation.*** The MoSCoE approach to service composition can be classified as semi-automatic. The process of composing the existing services is automatic, but the user is asked for refinements in case a service composition matching the user initial service request cannot be found. Furthermore, the user is expected to specify a UML State Machine representing the service request. This process may be extremely complex for non-technical users.

***Composition time.*** MoSCoE is mainly targeted to design-time service composition. The runtime service composition is not emphasized in the framework documentation.

***Coordination topology.*** MoSCoE creates a service composition strategy, or orchestration, which defines the behaviour of a mediator, consisting of a plan that allows the management of interactions with the different service composition components.

***Composition correctness.*** MoSCoE uses Symbolic Transitions System (STS) to represent services and service compositions. By using this formal representation, the framework has mechanisms to formally verify the created service compositions. MoSCoE checks soundness and completeness of the composition against the provided set of restrictions on the service composition. The service composition is also checked at the moment a new refinement is issued. This checking is used to provide specific information concerning the problems found at the composition time to meet the specified service user goals.

***Addressing and binding.*** MoSCoE defines static service bindings at design-time. However, constraints and rules are also defined for non-functional properties

of the service composition. These properties are monitored at runtime. If some specified event takes place, the engine stops the service composition execution and an alternative service composition is selected, if available.

### 5.4. SeCSE

SeCSE (Service-Centric Systems Engineering) [16] was a European project from the 6th Framework Programme for Research and Development. The SeCSE project focused on the creation of new methods, tools and techniques for requirements analysis, system integrators and service providers to support cost-effective development and use of dependable services and service-centric applications.

The four main research areas of the project were Service engineering; Service discovery; Service-centric systems engineering; and Service delivery. To address these areas, SeCSE has defined a methodology consisting of the following phases: i) Business requirements definition and service discovery; ii) Composition creation; iii) Instrumentation and monitoring rule definition; iv) Service regression testing; v) Deployment of service composition; vi) Service-centric system description; vii) Service-centric system publication.

***Service description.*** Faceted Service Specification [17] is used to perform service description. The proposed Facet Specification structure includes a stable element and variable elements. The principal stable element is that the XML file structure used to represent a Service Specification and associated Facets will not change, since these are independent of Facet types. Instances of any new Facet type can be listed in a Service Specification provided a type name has been defined and made public. Any new Facet Specification language can be used simply by embedding specifications written in the language within a Facet Specification file. The variable elements that the SeCSE runtime architecture needs to accommodate are: i) new Facet types, as service consumers' requirements evolve; ii) service specification languages, since new service specification languages are emerging and existing ones are still evolving. However, service consumers wishing to evaluate a service specification need to be able to establish whether their tool (if any) is capable of interpreting the specification. Hence, an indicator of the language used is needed, but the mechanism used to interpret it needs to accommodate potentially arbitrary choices of language and their versions.

***Service matching and selection.*** The service discovery phase is performed based on a user (or Service Integrator as used in SeCSE) service request. A service request is defined using UML Use Case specifications. Additionally the service request specifies also functional requirements, using VOLERE. Use cases and requirements are expressed in structured natural language, using a SeCSE tool called UCaRE.

Once the use cases and requirements are specified, they can be used to construct a service request in UCaRE, allowing the user to select the information to be used in the service discovery query. The service request query is then passed to EDDiE, the SeSCE service discovery engine. After some manipulation on the natural language service request query, a two steps matching is performed: i) XQuery text-searching functions to discover an initial set of services descriptions that satisfy global search constraints; ii) traditional vector-space model information retrieval, enhanced with WordNet to further refine and assess the quality of candidate services set. The resulting matches are then presented to the user.

***Information specification.*** SeCSE uses Faceted Service Specification to deal with information specification, and service description.

***Behaviour specification.*** The process model specification is based on the activities identified in the requirements analysis phase. The user identifies the different required activities, and based on the set of discovered services he realises how the service composition can be made. This is done through a workflow definition. Given this workflow, the user proceeds with the actual service composition, using the Composition Designer (CD) tool provided by the SeSCE project. CD allows manipulating the WSDL descriptions of the discovered services and creating the necessary BPEL service description. From the resulting data, CD can generate a UML model that is recognized by the SeSCE Architecture-time Service Discovery (ASD) tool, which allows the publication and discovery of the composed service.

The CD tool also allows the user to further enrich the service composition description with binding and monitoring rules. These rules enable runtime dynamic adaptation of the service composition in case specific events take place. The definitions of the rules consist of Event-Condition-Action (ECA) expressions.

***Level of automation.*** The service composition process can be considered as semi-automatic. The definition of the service composition is obtained based on a workflow specified by the user of the system. However, the user is supported in this task by the high level specification of the requirements provided in the requirements analysis phase. Furthermore, the actual service composition, or instantiation of the defined workflow, is performed based on the list of services

discovered automatically from the user's service request. The user also specifies binding rules, which allow a dynamic adaptation of the service composition according the defined constraints.

***Composition time.*** The composition process proposed by SeSCE can be considered as hybrid. The service composition is specified at design-time, as the dynamic binding rules. However, dynamic adaptations of the service composition take place at runtime, in case some pre-defined event takes place, such as the unavailability of a given service.

***Coordination topology.*** The coordination topology consists on a centralised topology or orchestration, where a single party manages the service composition execution, monitoring and reconfiguration.

***Composition correctness.*** SeCSE focuses mainly on QoS requirements verification, to perform runtime adaptation of the service composition. A technique based on a genetic algorithm is used for this purpose.

***Addressing and binding.*** Binding is defined at design-time. However, the SeCSE approach also defines binding rules at design-time, which allow one to specify ECA rules to be interpreted at runtime and guide possible dynamic binding changes at runtime, as some events are observed, leading to reconfiguration actions on the service composition.

### 5.5. WSMF

The Web Service Modelling Framework (WSMF) [18] is a framework for describing the various aspects related to web services composition. The framework provides a Web Services Execution Environment (WSMX), which is a reference implementation of the Web Services Modelling Ontology (WSMO) and operates using the Web Services Modelling Language (WSML).

The WSMF conceptual model incorporates four core elements that are essential to represent semantic web services and related issues, namely ontologies, which provide the common terminology used by other WSMO elements, services, which are requested, provided, and agreed upon by requesters and providers, goals, which provide means to characterize user requests in terms of functional and non-functional requirements, and mediators, which deal with interoperability problems between different WSMO elements. In addition to these core elements, WSMO introduces a set of core non-functional properties that are globally defined and may be used by all its modelling elements.

***Service description.*** In WSMO, requestors of a service express their objectives as goals, which are high level descriptions of concrete tasks. A WSMO goal description consists of a requested capability and requested interfaces. The former specifies the objective to be achieved in terms of a capability from the user perspective, while the latter specifies the communication behaviour for automated web service usage supported and required by the client. Analogously, a WSMO service description consists of a capability, which is a functional description of a web service describing constraints on the input and output of a service through the notions of preconditions, assumptions, post-conditions, and effects, and choreography interfaces that specify how the service behaves in order to achieve its functionality, i.e., the interaction behaviour supported by a service.

***Service matching and selection.*** In the context of WSMF, service discovery is based on matching abstract request goal descriptions with semantic annotations of services. Hence, in order to precisely express user goals with respect to discovery, WSMO goals carry an additional non-functional property *typeOfMatch*, which denotes the matchmaking notion to be applied. The simplest approach uses an algorithm that matches keywords from the goal description with keywords from the service descriptions. For the lightweight semantic discovery, the capabilities of goals and candidate services are transformed into a variant of WSML and expressed as *taxon* concepts. Once this transformation has taken place, an appropriate reasoning engine is used to determine if there is an overlap in the set of concepts resulting from the transformation of the goal and service capabilities, respectively. The QoS-based discovery mechanism can be used to filter services from a large set of candidates or to order services that match a goal according to some specific QoS characteristic.

***Information specification.*** In WSMF, ontologies are used to define a common agreed terminology, by providing concepts and relationships between these concepts. Only WSML is used as WSMX internal data representation. By reusing standard terminologies, different elements can be either linked directly or indirectly via predefined mapping and alignments, in order to achieve interoperability between services. The concept of mediation in WSMO has been introduced to handle heterogeneities that may exist between elements that should interoperate, by resolving data and behavioural mismatches. A WSMO mediator connects the WSMO elements in a loosely coupled manner, and provides mediation facilities for resolving mismatches that might arise in the process of connecting different elements defined by WSMO. More specifically, WSMO defines four types of mediators: OOMediators, which mediate heterogeneous ontologies, GGMediators, which connect Goals, WGMediators, which link web services to Goals, and WWMediators,

which connect interoperating web services, resolving their mismatches.

***Behaviour specification.*** The behaviour specification is based on Abstract State Machines (ASM), consisting of states and guarded transitions. A state is described by the WSMO ontology and the guarded transitions are used to express changes of states by means of transition rules. The domain ontology constitutes the underlying knowledge representation for the ASM, and transition rules specified in terms of logic formulas describe how state changes when a transition is executed. WSML defines a syntax and semantics for ontology descriptions and comprises different formalisms, most notably Description Logics and Logic Programming. The underlying formalisms are used to give a formal meaning to ontology descriptions in WSML, resulting in variants of the language that differ in logical expressiveness and in the underlying language paradigms.

***Level of automation.*** The framework aims at an automated, goal-driven service composition that builds on pre- and post-conditions associated to service descriptions.

***Composition time.*** During design-time, the design and implementation of adapters, creation of ontologies and service descriptions, rules for lifting/lowering, and mapping rules between ontologies are carried out. The runtime phase involves discovery, selection and execution of the appropriate services to accomplish a given goal.

***Coordination distribution.*** Information interchange for consumption and cooperation of services happens in a peer-to-peer manner, without the need of a central coordination entity.

***Composition correctness.*** Apparently there is no explicit support for correctness in this framework.

***Service binding.*** WSMO defines a proxy infrastructure for dynamic service binding and invocation. For each invoked service, a proxy has to be declared. The proxy allows referencing a service without knowing at design-time which concrete service is bound to it. This reference may consist of a goal definition or of a name, pre-conditions, post-conditions, input ports, output ports as well as error ports. The binding process happens at runtime and is based on binding rules defined in the proxy. The binding can be fixed to exactly one service, be defined in a registry (e.g., UDDI) or depend on input data coming from an input port. The last case means that the requester has full control over which service to select at runtime, because he can specify the binding criteria through the input ports. The only condition is that the data required to execute the composite service can be provided by the service bound at runtime.

## 6. Discussion

Table 2 summarizes the profiles of the composition approaches described in the previous section, according to multiple evaluation criteria defined within our comparison framework. By contrasting their profiles, it may be concluded that each approach focus on a specific set of phases involved in a composite service lifecycle, while disregarding others.

**Table 2. Comparison of the approaches**

| | METEOR-S | SODIUM | MoSCoE | SeCSE | WSMF |
|---|---|---|---|---|---|
| Service description | Semantically augmented with SAWSDL | UML models enriched with. constraints and OMG's QoS profile | Services are represented in OWL-S and WSDL | Services are represented in Faceted service specification | Capability described in terms of pre- and post-conditions, assumptions, and effects |
| Service matching and selection | three-phase matching algorithm based on semantic similarity | Based on USQL queries with behavioural constraints | Semantic reasoning optimized with non-functional properties | Two-phase matching algorithm based on text-searching functions | Keyword matching, lightweight semantic matching and QoS based |
| Behaviour specification | Based on process templates with BPEL-like syntax | A graphical composition language is used to define data and control flow | Based on UML state machine diagram and Symbolic Transition Systems | BPEL-like service composition creation based on abstract workflow definition | Based on abstract state machines, consisting of states and guarded transitions |
| Information specification | Model reference annotations and schema mapping annotations | Data objects used as internal data representation format and data transformations expressed in QVT | Inter-ontology mappings | Faceted service specification | Ontologies are used as internal data representation format and mediators are defined in case of data mismatch |
| Level of automation | Support for manual and semi-automatic | Semi-automatic support | Semi-automatic support | Semi-automatic support | Automatic |
| Composition time | Design-time and runtime | Design-time | Design-time | Design-time with binding rules for dynamic adaptation | Runtime |
| coordination distribution | Centralized | Centralized | Centralized | Centralized | Peer-2-peer |
| Composition correctness | State machine based verification of BPEL | No support for formal proof of correctness | Symbolic transitions system based | No explicit support | No explicit support |
| Service binding | Static binding, deployment-time binding and dynamic binding | Design-time, compilation-time, deployment-time and runtime | Static binding | Static and dynamic binding | Static and dynamic binding |

When comparing the described approaches, it is possible to notice that services can be described in different ways. They are described according to different existing standards (e.g. WSDL, SAWSDL, OWL-S, WSML), and can be characterized by a set of input and output parameters, QoS parameters,

keywords, pre- and post- conditions, effects, etc. Nevertheless, it has been done, at varying levels of abstraction and each of these levels implies a different description of services, ranging from simple unstructured keywords to detailed characterizations of possible state transitions.

The service discovery phase is directly dependent on the way services are described. Consequently, the achievable accuracy of a result in the discovery phase may vary significantly from one approach to another, since different sorts and amounts of information are available during the discovery phase and since more or less structure and semantic are embedded in the service descriptions, which are used by the matching algorithms. There is, however, an inherent trade off between expressiveness of the service descriptions and computational performance.

The way in which behaviour is specified also varies across the compared approaches. The behaviour of a composite service can be described explicitly, using some language that directly specifies the composition flow control (allowed order of invocations). This principle is currently taken by SODIUM and SeCSE approaches and reflects its primary focus on design-time service composition. Alternatively, such behaviour can also be described indirectly, by specifying the conditions under which the involved services in a composition can be invoked, its inputs and outputs parameters and the effects of such invocations. METEOR-S and MoSCoE allow IOPEs (inputs, outputs, preconditions and effects) to be specified at the level of WSDL operations. It allows AI planning techniques to be used to fully or partially automate the service composition process. The planning algorithms can be executed at design-time or runtime to find a suitable ordering of the operations, based on the initial conditions and the goal of a particular client. However, planning algorithms usually have high computational complexity and require substantial resources. WSMO specifies the conditions and effects using abstract state machines, consisting of states and guarded transitions. A state is described within an ontology and the guarded transitions are used to express changes of states by means of transition rules. However, this implicit behaviour specification may be neither intuitive nor trivial to make sure that the expectations implied by the designed transition rules match the expected operation message exchange patterns in the context of a service composition.

Concerning information specification, there are several converging ideas focusing on schema mappings and data transformations to cope with heterogeneity issues that can exist between the formats of the data exchanged between services. Similar to various existing approaches, developers may specify mappings between each element of an input/output parameter of a service and concepts from different data schemes. Ontologies are used as the information model throughout both WSMO and MoSCoE. Because WSMO heavily emphasizes mediation, mediators are a first class component of the WSMO service model. An example of a WSMO mediator for resolving data mismatches is the *ooMediator*, which links two ontologies, resolving possible mismatch issues between them. In METEOR-S, the data mediator module uses the data semantics of the proxies and the services, to perform XSLT transformations data mediation between the on-the-wire XML data. With this approach, the grounding needs to link the inputs and outputs of the service with the appropriate XSLT transformations. Therefore, mapping and merging of schemas becomes a core question and some (*semi*)automatic support has to be developed to reduce the exhaustive work needed for manual creation and maintenance of these mappings.

Regarding the coordination distribution, more traditional approaches assume a centralized coordination, where a central entity coordinates the invocation of services involved in a composition. In the other hand, WSMF goes beyond this traditional form of central coordination, where a peer-to-peer interaction takes place among equal partners, in terms of their level of control over other entities.

## 7. Conclusions

In this paper, we have presented an analytical framework for analysis and comparison of service composition approaches. The framework was developed following the phases of the composite service life-cycle. Additionally a set of criteria was identified to evaluate each of the considered life-cycle phases.

We claim that service composition plays a major role in enterprise interoperability, and so here we present some state of the art on service composition approaches. According to our framework, an ideal approach would efficiently cover all of the requirements that we have identified across the key phases of the composite service life-cycle. However, for practical reasons, the compared approaches focus on specific phases of the life-cycle, while neglecting others. Not surprisingly, the described approaches widely differ in how they address the above mentioned requirements.

Based on our study we conclude that none of the service composition approaches we investigated covers

all the life-cycle phases and is commonly accepted by the research community. For this reason, we believe there are opportunities to be exploited by combining the benefits of the different approaches. However, there are still some issues that have not been explicitly addressed in our study. An example is the way ontologies are used, since ontologies are currently a major technology for supporting service description and composition. Different organisations define ontologies in different ways, which may generate major problems of interoperability. Some approaches define manual mappings to deal with the ontologies interoperability problem; however these mappings or mediation techniques may be error prone and difficult to apply in realistic applications.

An issue that apparently is not being widely addressed is the support to end-users' service composition at runtime. This is a major research challenge and business opportunity, since the idea of delivering services at runtime on demand to end-users is a natural opportunity and benefit of service-oriented systems. However, as it can be observed from our study, the majority of the approaches mainly focus on design-time service composition, providing support to service developers. On this specific topic we foresee many research challenges as well as many business opportunities in the upcoming years.

## References


[1] F. Lautenbacher, B. Bauer, "A Survey on Workflow Annotation & Composition Approaches", *In Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SemBPM)*, Innsbruck, Austria, June 2007, pp. 12-23.

[2] Seog-Chan Oh, Dongwon Lee and Soundar R. T. Kumara, "A Comparative Illustration of AI Planning-based Web Services Composition," *ACM SIGecom Exchanges*, Vol. 5, No. 5, 2005, pp. 1-10.

[3] N. F. Noy, "Semantic integration: a survey of ontology-based approaches", *ACM SIGMOD Record*, Vol. 33, No. 4, December 2004, pp. 65-70.

[4] M.H. ter Beek, A. Bucchiarone, and S. Gnesi , "Formal Methods for Service Composition", *Annals of Mathematics, Computing & Teleinformatics*, Vol 1, No 5, 2007, pp. 1-10.

[5] D. Griffin, D. Pesch, "A Survey on Web Services in Telecommunications". *IEEE Communications Magazine*, July 2007, Vol. 45, No. 7, pp. 28-35.

[6] J. Brønsted, K. M. Hansen, M. Ingstrup, "A Survey of Service Composition Mechanisms in Ubiquitous Computing", In *Proceedings of UbiComp 2007 Workshop*, June 2007, Vol. 4717, No. 9, pp. 87-92, Innsbruck, Austria.

[7] IFIP TC5 SIG on Enterprise Interoperability, "TC5 SIG EI Aims and Scope", January 2008. Soon available at http://www.ifip.org/.

[8] D. Chen, "Enterprise interoperability framework", In Proceedings of Enterprise Modelling and Ontologies for Interoperability, EMOI - Interop 2006, CEUR Vol. 200, 2006.

[9] M.-S. Le et al. (Eds.), "Enterprise interoperability research roadmap", V4.0, July 2006. Available at ftp://ftp.cordis.europe.eu/pub/ist/doc/directorate_d/ebusiness/ei-roadmap-final_eng.pdf.

[10] M.P. Papazoglou, D. Georgakpoulos, "Introduction to special issue on Service Oriented Computing", Communications of the ACM, Vol. 46, No. 10, 2003, pp. 24-28.

[11] R. Hull, M. Benedikt, V. Christophides, and J. Su. "E-Services: A Look Behind the Curtain". In *Proceedings of the PODS 2003 Conference*, San Diego, CA, USA, 2003.

[12] G. Alonso, F. Casati, H. Kuno, V. Machiraju, "*Web Services. Concepts, Architectures and Applications*", 2004, Springer-Verlag, Berlin Heidelberg.

[13] K. Verma, K. Gomadam, A. P. Sheth, et al.. "The METEOR-S Approach for Configuring and Executing Dynamic Web Processes", Technical Report , 2005.

[14] S. Topouzidou. "SODIUM, Service-Oriented Development In a Unified framework", Final report IST-FP6-004559. http://www.atc.gr/sodium.

[15] J. Pathak, S. Basu, V, Honavar. "Modeling Web Services by Iterative Reformulation of Functional and Non-functional Requirements", In *proceedings the 4th International Conference on Service-Oriented Computing (ICSOC)*, Chicago, USA, December 2006, pp. 314-326.

[16] The SeCSE team. "Designing and Deploying Service-Centric Systems: The SeCSE Way", In *proceedings of Workshop: Service Oriented Computing: a look at the Inside*, ICSOC 2007, Vienna, Austria, September 2007.

[17] SeCSE Project. "Specification Language Definition". *SeCSE project deliverable* (A1D23), 2007.

[18] D. Roman, et al.. "*Web Service Modeling Ontology",* Applied Ontologies, vol. 1, pp. 77-106, 2005.

# A Reference Model for Cross-Organizational Coordination Architectures

Eddy Truyen and Wouter Joosen
Department of Computer Science
K.U.Leuven
Celestijnenlaan 200A
B-3001 Leuven, Belgium
Eddy.Truyen@cs.kuleuven.be

## Abstract

*Dominant trends in distributed systems are increasing network connectivity, business-to-business integration and an evolution from desktop-oriented software to service-oriented software. The size and complexity of networked services has furthermore broadened from intra-organization to cross-organizational service provisioning. In the last decade many coordination architectures have been proposed to coordinate the provisioning of services across organizational boundaries. These approaches involve various types of service contracts and policies, and various coordination activities including negotiation, validation, enactment, monitoring and enforcement of service policies and contracts. In this paper, we propose a reference model for such coordination architectures in order to support the following goals: (i) to facilitate the analysis, comparison and discussion of different coordination architectures and (ii) to allow for constructive proposals about improving the existing coordination architectures.*

## 1. Introduction

Organizations need to integrate their business processes in order to be able to operate and survive in a market. When multiple independent organizations interact with each other, they have to coordinate their activities in order to ensure that their interaction leads to an added-value result that is satisfactory to all parties. Coordination, which can also be defined as "*the managing of dependencies between agents in order to foster harmonious interaction between them*" [21], is indispensable for effective cooperation between autonomous organizations, as well as for safe competition between them[25].

Coordination involves a certain *agreement*, i.e., a set of rules of engagement, that must be complied with by all partipating organizations for the business relationship to be exploited in a safe and harmonious way. The agreement is to be specified using an agreement language. An important assumption is that the organizations involved might not trust each other. So an important requirement for cross-organizational coordination is that it should enforce the regulated interactions (as encoded in the agreement) between two or more mutually distrusting and autonomous organizations.

Information technology is used to speed-up coordination. In this paper a *coordination middleware* is defined as a distributed management system [23] that provides technological support for the efficient organization of providing and consuming services across organizational boundaries [35]. By studying different coordination architectures (i.e., the software architecture of existing coordination middlewares), we can derive *reference models* [6] for cross-organizational coordination. A reference model is defined as a standard decomposition of a problem into parts that cooperatively solve the problem. Reference models arise mostly from experience and as a result are a characteristic of mature domains [6]. In our opinion, the understanding and acceptance of a common reference model for cross-organizational coordination is an important prerequisite in order to begin with the definition of a standardized approach for cross-organizational coordination.

The goal of this paper is to propose a reference model for cross-organizational coordination. The scope of the targetted reference model is focussed towards the technological and infrastructural aspects of cross-organizational coordination. We defer the reader, interested in the socio-organizational aspect of coordination, to reference models defined by researchers from the University of St. Gallen [33, 34].

Figure 1 shows a very simple reference model for cross-organizational coordination. Firstly, agreements must be represented digitally by means of an agreement language that offers the necessary concepts for describing and executing agreements. Second, coordination middleware must be

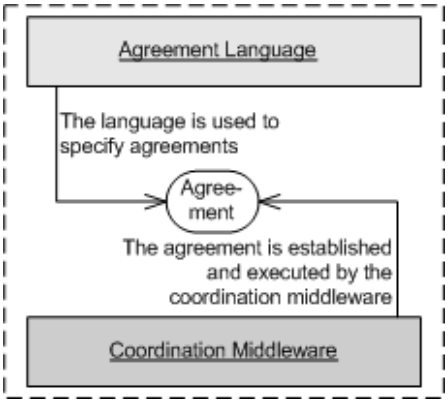



**Figure 1. A simplified reference model for cross-organizational coordination**

developed in order to establish agreements dynamically, and to enforce the agreements or detect violations against these. Although this simple reference model already presents a commonly accepted decomposition of the major problems involved, it is not detailed enough in order to be of any use. An effective reference model should instead enable us to analyze, compare and classify existing coordination architectures and, secondly, it should also enable us to make constructive proposals about improving the existing coordination architectures. To achieve this effectively, we seek a reference model with a finer-grained decomposition that unravels the different concerns of importance such that it becomes possible to assess these concerns in isolation.

The process of creating the reference model was based on the following approach. First we have performed a literature study of existing cross-organizational coordination middlewares. Secondly, based on the literature we have identified commonly recurring concepts and problems. We have performed this process iteratively and incrementally: a prior version of the reference model is validated by mapping it to a new coordination middleware that was not yet included in the literature study. Any found incompletenesses or inconsisties in the current version of the reference model are then amended, leading to a new version of the reference model. Overall, the process of creating the reference model was mostly based on generalization and abstraction, and not on rigor deduction. The set of coordination middlewares that were included in the literature study have been listed in [40].

The rest of this paper is structured as follows. Section 2 introduces and motivates the reference model for cross-organization coordination. Then section 3 compares seven well-known coordination architectures by mapping these to the reference model. Finally section 4 draws important conclusions from these mappings. In particular we identify important gaps in required coordination functionality that is not bridged by current coordination architectures. This allows us to propose some senseful directions for future work on cross-organizational coordination architectures.

## 2. Reference model

There is a vast heterogeneous body of research on cross-organizational coordination. Various types of agreements are supported and different kinds of language and middleware technology are being used, different coordination middlewares vary in their multitude of architectural styles such as Peer-2-Peer and Client-Server. However, to solve the problem of enterprise interoperability, it is of paramount importance that, within each business domains, reference architectures are defined with standard interfaces for establishing and executing agreements. A reference model that defines a standard decomposition of the coordination problem is a first step to obtain such reference architecture. A reference architecture for a particular business domain is derived by mapping the reference model for cross-organizational coordination onto software components using an architectural style [6]. The definition of such a domain-specific reference architecture is out of the scope of this paper. The paper only presents a reference model with the afterthought that a mapping to a reference architecture is one of its possible applications.

As pointed out in section 1, the reference model has been obtained by refining the simplified model in Figure 1 with additional dimensions. The simplified model distinguishes between three dimensions: (i) the type of agreements that are established, (ii) the language for describing agreements, and (iii) the middleware for establishing and executing the agreements. The extended reference model (see Figure 2) is then obtained by refining the two last dimensions into 4 and 2 sub-dimensions, respectively. The resulting reference model can thus be represented as a tree that has 7 subdimensions as its leafs:

1. The type of agreements ($Dim.\ 1$) that are established.
2. The language for describing agreements ($Dim.\ 2$) where we distinguish between four sub-dimensions:
   (a) the conceptual model of the language for describing agreements. ($Dim.\ 2.a$).
   (b) the computational model of the language for making agreements implementable ($Dim.\ 2.b$)
   (c) the management concepts of the language for managing large sets of agreements and dealing with their potential conflicts ($Dim.\ 2.c$).
   (d) the implementation technology of the language so that computers can process it ($Dim.\ 2.d$).

3. The middleware for establishing and executing agreements ($Dim.\ 3$) where we distinguish between two sub-dimensions:

   (a) the functionality that the underlying coordination middleware offers for establishing and executing agreements($Dim.\ 3.a$).

   (b) the distributed systems architecture of the coordination middleware($Dim.\ 3.c$).

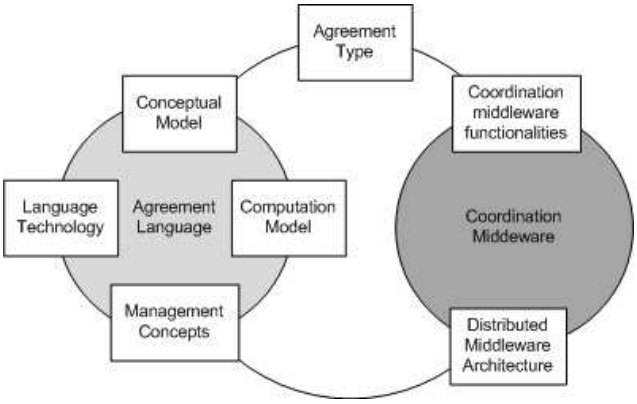



**Figure 2. The refined reference model**

The division of the agreement language and coordination middleware dimensions into multiple subdimensions is motivated by the desire of genericity and separation of concerns. The conceptual model of the language should be as independent as possible from the computational model for example. This independence enables that different organizations can implement the same agreement differently depending on their choice of implementation platform, while adhering to the terms of the agreement. Similarly, establishment and execution of agreements are actually taken care of by two separate middleware layers. The design of the establishment layer should be as independent as possible from the execution layer. This independence enables a single organization to have a uniform execution middleware for multiple agreement domains[24]. Also, in some business domains, this independence is required such that organizations can dynamically form business relationships in a virtual market using a standardized agreement model and matchmaking engines [31, 15], while still using its own proprietary execution middleware.

The following sections provide for each of the 7 subdimensions an overview of common problems and provide a set of elements for comparing existing coordination architectures.

## 2.1. Types of agreements

We distinguish between four types of agreements in the literature:

- Type 1: Agreements about the flow of business domain interactions; How is the cross-organizational relationship going to be exploited? What messages will be exchanged between the service consumer and service provider organizations and which data types are involved?
- Type 2: Agreements about modal constraints, i.e. authorizations, obligations, prohibitions, timings[11] that organizations must agree to and must respect as long as their business relationship holds.
- Type 3: Service-Level Agreements (SLA's) about the deployment and configuration of non-functional requirements of provided services, e.g. must the interactions be performed in a transactional way or not, must certain Quality-of-Service (QoS) requirements be enforced during the interactions?
- Type 4: Agreements at the level of protocols and standards used for implementing the non-functional requirements. For example, what specific transaction protocol must be used?

## 2.2. Language models and technology

In order to regulate the interactions between two or more independent agents, one needs a language for describing how the interactions are to be done. Such a language must (a) offer *a conceptual model* for describing the regulations at a sufficient high-level of abstraction that is independent from the organizations internal processes and data, (b) offer a *computational model* that ensures that regulations are implementable by the underlying system, (c) offer additional modeling concepts for managing large sets of agreements and (d) be *computer-interpretable*.

### 2.2.1 Conceptual model

A conceptual model for specifying cross-organizational agreements provides the *modeling concepts* necessary to describe all information that is needed to establish and exploit cross-organizational service relationships. In order to be successful, the conceptual model must have a clear scope on what is the domain or area where the agreements apply. The domain may be a country, or a professional institute, or a specific market such as financial services. Broadly speaking, we can divide the work on modeling concepts for cross-organizational coordination into management policies (e.g., [25, 9, 11]), electronic contracts (e.g.,

[15, 24, 28, 37, 41]) and conceptual models that offer a combination of contracts and policies.

**Policies** *Policies* are a means of specifying and influencing management behavior within a distributed system[38]. Policies can be broadly classified in *authorization* policies and *obligation* policies, where the former captures access control requirements of the system and the latter is about requirements related to system behavior.

Although policies are originally envisaged to be specified and enforced within a single organization, policies are also useful for regulating inter-organizational interactions. In this context, policies typically support agreement types 1 and 2. For example consider the following four policies from the retail business domain that regulate the interactions between a purchaser and a seller organization:

> [$Authorization\ Policy\ 1$]"The seller has the right to inspect a purchaser's order at any time,..."
>
> [$Prohibition\ Policy\ 2$]"...except when the order is destined for a person with VIP status."
>
> [$Obligation\ Policy\ 3$]"The purchaser is obliged to pay the price of a product no later than 10 days after the date of ordering."
>
> [$Obligation\ Policy\ 4$] "A seller is required to ship an ordered item within 10 days to a Purchaser after the Purchaser has paid the order". [$obligation$]

**Electronic contracts** An *electronic contract* is an agreement between two or more organizations that stipulates how the involved parties are expected to behave. A contract generally consists of a structured set of entries, called clauses.

Contract models exist in the literature for all four agreement types as presented in section 2.1. Electronic contracts for agreement types 1 and 2 are typically electronic versions of conventional contracts on paper between business organizations. Electronic contracts for agreement types 3 and 4 typically correspond with SLA's and QoS contracts that eventually have to be mapped to available computing resources.

### 2.2.2 Computational model

Agreements (either contracts or policies) may be specified at the conceptual level (as goals) and then need to be refined into a *implementable actions* that can be enforced upon the contracted application services.

To achieve implementable actions, the coordination language must also offer a computational model that supports cross-organizational service provisioning across independent organizations. Generally speaking, this computational model offers behavioral concepts that enable coordination architectures (see section 2.3) to observe service behavior by registering for events and control states that relate to the progress with which the service execution fulfils the agreement, and to control the service behavior by allowing, forbidding, or enforcing actions to take place at the service consumer or service provider site.

For example, we will show what behavioral concepts are needed such that the above $Obligation\ Policy\ 4$ can be refined into an implementable policy specification[1]:

```
   +Obligation(Seller)
{onEvent(OrderPaidbyPurchaser):
   action(OrderedItemSentToPurchaser)
within(10 days)}
```

We can distinguish the following behavioral elements in the policy specification: (1) the subject Seller to which the obligation applies, (2) the action of shipping an ordered item, (3) the target Purchaser (4) an additional time constraint of 10 days, and (5) an activation condition stating that the policy should become active after the OrderPaidByPurchaser event has occurred.

All these elements must be mappable to implementable objects and operations available in the underlying software system of the service provider and consumer.

Note that the definition of a policy specification can consist of multiple levels of abstractions. For example, suppose `OrderPaidByPurchaser` corresponds to a high-level business event that is not necessarily directly represented as a message exchange between purchaser and seller. In order to implement or detect such a business event, it must be refined into a pattern of lower-level events that directly come from the internal processes of the seller and/or purchaser organization. How this is implemented, needs to be specified at a lower level of abstraction.

Event-based architectural description languages already support some of these mechanisms. For example Rapide [19, 18] provides *maps* that correlate a complex pattern of multiple event occurrence into a single higher-level business event. A similar mechanism is provided by Finesse [8, 32]. Milosevic [24] and Yildiz [42] both point out the usefulness of these existing complex event processing languages for cross-organizational coordination.

### 2.2.3 Management concepts

Management concepts are necessary for facilitying agreement specification for large-scale systems with millions of objects. This typically involves a concept that provides a

[1]We describe this policy specification using a pseudo language. See section 2.2.4 for more details about what language technologies have been used by existing coordination architectures.

way of grouping objects to which agreements apply [11]. Also concepts are required for hierarchical grouping of agreements in order to help manage the large set of agreements that apply for a given organization.

Another important aspect of agreement management is the detection and resolution of conflicts between multiple agreements[2]. Existing solutions for dealing with conflicts between policies originates from the work of Lupu and Sloman. [20]. They distinguish between two kinds of conflicts: modality conflicts and application-specific conflicts. A *modality conflict* arises for instance when two policies associate both an authorization and a prohibition to the same subject, action, target tuple. Modality conflicts are domain-independent.

*Application-specific conflicts* are conflicts that are related to the specific application for which the policies have been defined. Application-specific conflicts are further classified into conflict of duties, conflict of interests, conflict of priorities for resources, multiple manager conflict and self-management conflict [26, 4]. A conflict of duty, for example, will arise when the same subject is permitted to perform actions that, in the context of the application, are defined to be conflicting. For example, in the context of a financial application, the same subject should not have the right to sign a payment check and authorize that payment at the same time. Lupu and Sloman have worked on various solutions for detecting both kinds of conflicts.

#### 2.2.4 Language technology

The modeling and behavioral concepts offered must eventually be expressed by means of a language format that can be processed by a computer. While it is possible to develop a computer language from scratch for this (e.g. [24]), most coordination architectures employ an existing language technology or specification standard. In particular the following language technologies have already been used: XML – e.g. [14], Ontology Languages and the Semantic Web [7] such as OWL [22] – e.g. [9], WS-Policy and WS-Policyattachment [12] – e.g. [41], Object-oriented programming languages [29] – e.g. [11], the Meta Object Facility (MOF) [30] – e.g. [37], Finite state machines – e.g. [28], Logic programming languages such as Prolog – e.g. [25], reflection as a complementary implementation technology – e.g. [25].

### 2.3. Coordination middleware functionalities

Besides a language for specifying agreements, coordination middleware is needed to support exploitation of the cross-organizational service provisioning across independent organizations. Broadly speaking, we distinguish between two main responsibilities for cross-organizational coordination middleware [23, 15]: (i) *establishment* of agreements between a service consumer and service provider organization and (ii) *execution* of the contracted services in conformance with the agreement.

Based on [23], we decompose the broad responsibilities of the establishment and execution layers into more specific functionalities.

#### 2.3.1 Establishment layer

Establishment involves negotiation and validation of agreements.

**Negotiation** Organizations aim to use matchmaking technology [31] for speeding up agreement negotiation such that contracted business relationships can be achieved dynamically without engaging into lengthy and costly human-to-human negotiations. With this end in view, agreement templates are often used. An *agreement template* is a common form that is used as a standard within the agreement domain. Examples of agreement templates are: sales and order contracts, car and house sale contracts, airline tickets. An agreement template is similar to a regular agreement with predefined properties that need to be filled in the course of agreement negotiation [15]. In most agreement templates, the agreeing parties can negotiate the data to be written in the blanks of the template, but not the clauses.

**Validation** Agreement validation corresponds with validating whether there are no ambiguities in the clauses of an established agreement. An agreement template may help with validation as it may provide rules that constrain how the agreement template may be instantiated such that an agreement is considered valid.

#### 2.3.2 Execution

Execution involves enactment, monitoring and enforcement.

**Enactment** When agreements will be negotiated dynamically, exploitation of the agreements is preferably also performed on the fly. This requires the ability to defer resource allocation to the latest time, such that an organization is freed from the risk of having to pre-allocate resources for long periods of time [15]. *Enactment* is the process that dynamically creates the appropriate infrastructure to the point where the exploitation of the business relationship is imminent. Some systems also offer concepts for automating the mapping of an agreement to necessary resources; this

[2] Note that conflicts between the clauses in the same agreement are considered as ambiguities that should be dealt with as part of validating the agreement.

is however limited to coordination middleware supporting agreement types 3 and 4 involving non-functional requirements and QoS. The execution layer in these middleware platforms relies on lower-level middleware such as deployment containers with pre-fabricated common services (e.g., transactions, persistence, security), QoS-enabled application servers [13] and dynamic distributed resource management systems (such as e.g., [5, 43]).

**Monitoring** Although organizations are expected to implement the agreements themselves, one may not trust that organizations will never cheat or deviate from the established agreement. Hence most architectures support *monitoring* the execution of the internal processes and data of the organizations. When violations occur, notifications are sent to the appropriate parties.

**Enforcement** The system may also trigger actions that aim at rectifying the deviations that cause an agreement violation. Molina-Jimenez et al. [28] state that every enforcement middleware must have two important properties: (1) safety, meaning that local policies of an organization should not be compromised by failures or misbehavior by other parties. (2) liveness, requiring that if all the parties are correct (not misbehaving), then agreed interactions should take place despite finite number of temporary omission failures in the network or computers.

**Dismantling** The execution layer must also offer support for terminating running agreements. This is a succint process that takes a lot of additional coordination between the business parties involved. Typically, it means shutting down the service, or temporarily putting the overall system in a dual mode where ongoing transactions are completely monitored against the agreement, but new transactions are not subject anymore to the agreement.

### 2.4. Distributed architecture

A final interesting dimension is the distributed architecture decisions behind the design of the coordination middleware. As monitoring and enforcement are the most complex functionalities, these are often the most weighing factors in the distributed architecture of the coordination middleware

Generically speaking, a monitoring and enforcement system consists of a *controller* which maintains state about the system in order to measure the progress of the execution of the system in relation to the contract. This state may be updated at run-time when the controller receives events. These events are filtered and dispatched by an *event interception* mechanism.

There are various alternative distributed architectural styles for coordination middleware:

**Centralized vs. Decentralized** Both controller and interception mechanism can be implemented in either a centralized or decentralized way. The latter implies that each organization has deployed its own local instance of the coordination middleware, or the middleware is deployed on a nearby node. The latter entails that a trusted third party system governs all interactions between the organizations.

**Reactively vs. Proactively** Monitoring and enforcement can be implemented either reactively or proactively [27]; the system may interact with the cross-organizational processes in two different ways: a *reactive* system intercepts messages exchanged between business processes and reacts by approving or disapproving them; a *proactive* system drives the cross-organizational interaction between the business processes by inviting the partners to send valid messages.

## 3. Mapping of existing coordination architectures

To illustrate the proposed reference model, we have mapped seven well-known coordination architectures to the reference model. These are Ponder [11], Law-Governed Interaction (LGI) [25], the Business Contract Language (BCL) and Architecture (BCA) [24], Crossflow [15, 14], The FSM-based approach from The Tapas project [28], GlueQoS [41] and T-BPEL [39]. As stated above, these mappings enable us to compare the different coordination architecture and also allow to make constructive proposals for improvements. Figure 3 gives a concise overview of the mapping of these coordination architectures to the elements of the reference model. The detailed descriptions of these mappings can be found elsewhere [40]. For the sake of completeness, a summary of the 7 mappings is given in appendix A.

## 4. Conclusions and Future work

This section aims drawing some general conclusions from the mapping of these seven coordination architectures to the reference model. The remainder of this summarizing section is structured as follows. We first discuss the main differences between policy-based and contract-based approaches. Secondly, future directions in the construction of cross-organizational coordination architectures are proposed.

### 4.1. Differences between policies and electronic contracts

In our opinion three main differences between policies and electronic contracts have emerged from the mapping.

| | Agreement Type *Dim. 1* | Modeling Concepts *Dim 2.a* | Behavioral Concepts *Dim. 2.b* | Management Concepts *Dim 2.c* | Language Technology *Dim 2.d* | Coordination Functionalities *Dim 3.a* | Middleware Architecture *Dim 3.b* |
|---|---|---|---|---|---|---|---|
| Ponder | 2 | Policy | Event calculus | Domains<br>Policy groups<br>Conflicts: modality, application−specific | Object−oriented | Enforcement | Reactive<br>Decentralized |
| Law−governed interaction | 1, 2 | Policy (law) | Events<br>Internal states | Policy groups | Prolog | Enforcement | Reactive<br>Decentralized |
| BCL | 1,2 | Contract (community)<br>Policy | Event types<br>Event patterns<br>Internal states | Conflicts: application−specific | Proprietary | Monitoring<br>Enforcement<br>(Establishment) | Reactive<br>(De)centralized |
| CrossFlow | 1 | Contract | Internal enactment specification | n/a | XML (Workflow) | Negotiation<br>Enactment<br>Dismantling | Reactive & proactive<br>Decentralized |
| FSM−based contracts | 1, 2 | Contract | Finite state machine (FSM) | n/a | FSM formalism | Validation<br>Monitoring | Proactive<br>Decentralized |
| GlueQoS | 3 | Contract (QoS Policy) | Aspect joinpoint model | Conflicts: application−specific | WS−Policy | Negotiation<br>Enactment | Decentralized |
| T−BPEL | 4 | Contract (transaction coupling mode) | Contract is transformed to Java implem. | Conflicts: application−specific | WS−policy | Negotiation<br>Validation<br>Enactment | Decentralized |

**Figure 3. Classification of coordination architectures**

Firstly, policies are focussed on governing business-level interactions and modality constraints (agreement types 1 and 2), while electronic contracts not only support these but also agreements about QoS features and technical protocols (agreement types 3 and 4). Secondly, contract frameworks generally support much more coordination functionalites than policy-based architectures that all exclusively focus on enforcement. Thirdly, the problem of managing policies has been thoroughly studied, whereas management of contracts has been largely ignored in the work on electronic contracts. Especially, the problem of dealing with conflicts between policies has been thoroughly studied, whereas conflicts between electronic contracts have been largely ignored until now. Instead, the focus lies rather on contract validation that detects ambiguities between different clauses in a single contract.

## 4.2. Directions for future work

Based on the reference model and the mapping exercise, we are able to make some constructive protocols for improving the existing coordination architectures.

**Policy and contracts united** First of all, these above observations indicate that contracts and policies relate to each other in major/minor balance (at least when considering cross-organizational coordination). Contract frameworks offer the most complete set of concepts and mechanisms for cross-organizational coordination. Yet policy-driven systems are an important contribution as they offer complementary solutions. This complementarity shows in at least three different ways. First of all, the concepts used for specifying obligations and rights in contracts (Agreement type 2) are very similar to those of policies. As such policy-based enforcement can be leveraged for this type of agreement. Secondly, existing contract-based coordination architecture could be extended with policy-based enforcement. Consider for instance the situation where a policy is dynamically deployed to rectify a contract violation. Thirdly, the mechanisms for policy management can be adapted for management of contracts.

**Enactment of agreements** Clearly for agreement types 1 and 2, enactment of agreements is the least understood area of research. As pointed out in the Crossflow project, the mapping of a contract to an existing infrastructure is beyond the current technological state-of-the art.

**Complete coordination architectures** Currently none of the 7 coordination architectures simultaneously support all four agreement types. Either support for agreement type 1 and 2, or 3 and 4 is offered, but never the combination of all four types. Furthermore, none of the these archi-

tectures support all six kinds of coordination middleware functionalities (negotiation, validation, enactment, monitoring, enforcement, dismantling). Especially the dismanting of agreements is not well supported in particular.

**Standardization** Orthogonal to the issue of improving existing coordination middleware, there is also the issue of standardization. As pointed out in section 2, the reference model could potentially be used as an instrument to derive commonly accepted architectural blueprints for coordination middleware. The reference model after all aims to offer a common vocabulary of important problems to be solved. This helps people in standardization bodies to discuss with less ambiguity about the required features for their particular coordination architectures. A highly related project is this vain is the European Virtual Laboratory for Enterprise based on the INTEROP project [1]. In this project an extensive "Interoperability" glossary and taxonomy has been defined. Obviously, the reference model presented in this paper should integrate and be compatible with this knowledge base.

## References


[1] The INTEROP vlab portal. http://interop-vlab.eu/.

[2] X. Ao, N. H. Minsky, and T. D. Nguyen. A hierarchical policy specification language and enforcement mechanism for governing digital enterprises. In *POLICY*, pages 38–49, 2002.

[3] A. K. Bandara, E. Lupu, J. D. Moffett, and A. Russo. A goal-based approach to policy refinement. In *POLICY*, pages 229–239. IEEE Computer Society, 2004.

[4] A. K. Bandara, E. Lupu, and A. Russo. Using event calculus to formalise policy specification and analysis. In *POLICY*, pages 26–. IEEE Computer Society, 2003.

[5] K. N. Baochun Li. A control-based middleware framework for quality of service adaptations. *IEEE Journal of Selected Areas in Communication, Special Issue on Service Enabling Platforms*, 17(9):1632–1650, September 1999.

[6] L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practice*. Addison-Wesley, 1998. Chapter 2. What is software architecture.

[7] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, pages 35–43, May 2001.

[8] A. Berry and S. M. Kaplan. Open, distributed coordination with finesse. In *SAC*, pages 178–184, 1998.

[9] J. M. Bradshaw, A. Uszok, R. Jeffers, N. Suri, P. J. Hayes, M. H. Burstein, A. Acquisti, B. Benyo, M. R. Breedy, M. M. Carvalho, D. J. Diller, M. Johnson, S. Kulkarni, J. Lott, M. Sierhuis, and R. van Hoof. Representation and reasoning for daml-based policy and domain services in kaos and nomads. In *AAMAS*, pages 835–842. ACM, 2003.

[10] M. Charalambides, P. Flegkas, G. Pavlou, A. K. Bandara, E. C. Lupu, A. Russo, N. Dulay, M. Sloman, and J. Rubio-Loyola. Policy conflict analysis for quality of service management. In *POLICY*, pages 99–108. IEEE Computer Society, 2005.

[11] N. Damianou, N. Dulay, E. Lupu, and M. Sloman. The ponder policy specification language. In M. Sloman, J. Lobo, and E. Lupu, editors, *POLICY*, volume 1995 of *Lecture Notes in Computer Science*, pages 18–38. Springer, 2001.

[12] S. B. et al. Web Services Policy Framework (WS-Policy) and Web Services Policy Attachment (WS-PolicyAttachment). http://schemas.xmlsoap.org/ws/2004/09/policy/.

[13] G. Ferarri, G. Lodi, F. Panzieri, and S. K. Schrivastava. Tapas architecture: Qos enabled application servers. Technical Report Project Deliverable D7, 2003.

[14] P. W. P. J. Grefen, K. Aberer, H. Ludwig, and Y. Hoffner. Crossflow: Cross-organizational workflow management for service outsourcing in dynamic virtual enterprises. *IEEE Data Eng. Bull.*, 24(1):52–57, 2001.

[15] Y. Hoffner, S. Field, P. W. P. J. Grefen, and H. Ludwig. Contract-driven creation and operation of virtual enterprises. *Computer Networks*, 37(2):111–136, 2001.

[16] *26th International Conference on Software Engineering (ICSE 2004), 23-28 May 2004, Edinburgh, United Kingdom*. IEEE Computer Society, 2004.

[17] R. A. Kowalski and M. J. Sergot. A logic-based calculus of events. *New Generation Comput.*, 4(1):67–95, 1986.

[18] D. C. Luckham and B. Frasca. Complex event processing in distributed systems. Technical Report CSL-TR-98-754, 1998.

[19] D. C. Luckham, J. J. Kenney, L. M. Augustin, J. Vera, D. Bryan, and W. Mann. Specification and analysis of system architecture using rapide. *IEEE Trans. Software Eng.*, 21(4):336–355, 1995.

[20] E. Lupu and M. Sloman. Conflicts in policy-based distributed systems management. *IEEE Transactions on Software Engineering*, 25(6):852–869, 1999.

[21] T. W. Malone and K. Crowston. The interdisciplinary study of coordination. *ACM Comput. Surv.*, 26(1):87–119, 1994.

[22] D. L. McGuinness and F. van Harmelen. Owl web ontology language overview. World Wide Web Consortium, Recommendation REC-owl-features-20040210, Feb. 2004.

[23] Z. Milosevic, A. Berry, A. Bond, and K. Raymond. Supporting business contracts in open distributed systems, 1995.

[24] Z. Milosevic, P. F. Linington, S. Gibson, S. Kulkarni, and J. B. Cole. Inter-organisational collaborations supported by E-Contracts. In W. Lamersdorf, V. Tschammer, and S. Amarger, editors, *I3E*, IFIP Conference Proceedings, pages 413–429. Kluwer, 2004.

[25] N. H. Minsky and V. Ungureanu. Law-governed interaction: a coordination and control mechanism for heterogeneous distributed systems. *ACM Trans. Softw. Eng. Methodol.*, 9(3):273–305, 2000.

[26] J. D. Moffett and M. S. Sloman. Policy conflict analysis in distributed system management. *Journal of Organizational Computing*, 1993.

[27] C. Molina-Jiménez, S. Shrivastava, and J. P. Warne. A method for specifying contract mediated interactions. In *EDOC*, pages 106–118. IEEE Computer Society, 2005.

[28] C. Molina-Jiménez, S. K. Shrivastava, E. Solaiman, and J. P. Warne. Run-time monitoring and enforcement of electronic contracts. *Electronic Commerce Research and Applications*, 3(2):108–125, 2004.

[29] O. Nierstrasz. A survey of object-oriented concepts. In *Object-Oriented Concepts, Databases, and Applications*, pages 3–21. 1989.

[30] T. O. M. G. (OMG). The meta-object facility v1.4. Technical Report formal/2002-04-03 edition, The Object Management Group (OMG), April 2002.

[31] M. Paolucci, T. Kawamura, T. R. Payne, and K. P. Sycara. Semantic matching of web services capabilities. In I. Horrocks and J. A. Hendler, editors, *International Semantic Web Conference*, volume 2342 of *Lecture Notes in Computer Science*, pages 333–347. Springer, 2002.

[32] A. Rakotonirainy, A. Berry, S. Crawley, and Z. Milosevic. Describing open distributed systems: A foundation. *Comput. J.*, 40(8):479–488, 1997.

[33] B. Schmid and M. Lindemann. Elements of a reference model for electronic markets. *System Sciences, 1998., Proceedings of the Thirty-First Hawaii International Conference on*, 4:193–201 vol.4, Jan 1998.

[34] C. Schroth. A reference model for seamless cross-organizational collaboration in the public sector. In *In Tagungsband der Multikonferenz Wirtschaftsinformatik (MKWI 2008)*.

[35] C. Schroth, T. Janner, and V. Hoyer. Strategies for cross-organizational service composition. *e-Technologies, 2008 International MCETECH Conference on*, pages 93–103, Jan. 2008.

[36] S. Shrivastava. Tapas final report. Technical Report Project deliverable D20, 2005.

[37] J. Skene, D. D. Lamanna, and W. Emmerich. Precise service level agreements. In ICSE [16], pages 179–188.

[38] M. Sloman. Policy driven management for distributed systems. *J. Network Syst. Manage.*, 2(4), 1994.

[39] S. Tai, T. A. Mikalsen, E. Wohlstadter, N. Desai, and I. Rouvellou. Transaction policies for service-oriented computing. *Data Knowl. Eng.*, 51(1):59–79, 2004.

[40] E. Truyen and W. Joosen. Coordination architectures for cross-organizational service provisioning. Technical Report Project IWT 040116 "'AspectLab", deliverable d3.2.1.a, Dept. of Computer Science, K.U.Leuven, Dec. 2006. http://www.cs.kuleuven.ac.be/∼distrinet/projects/AspectLab/deliverables/d3.2.1.pdf.

[41] E. Wohlstadter, S. Tai, T. A. Mikalsen, I. Rouvellou, and P. T. Devanbu. Glueqos: Middleware to sweeten quality-of-service policy interactions. In ICSE [16], pages 189–199.

[42] U. Yildiz, O. Marjanovic, and C. Godart. Contract-driven cross-organizational business processes. In *The 2nd International Conference on Information Management and Business - IMB 2006*, 2006.

[43] W. Yuan, K. N. abd S. Adve abd D. Jones, and R. Kravets. Design and evaluation of a cross-layer adaptation framework for mobile multimedia systems. In *SPIE/ACM Multimedia Computing and Networking Conference (MMCN'03)*, Santa Clara, CA, January 2003.

## A. Summary of mappings

**Ponder**

1. Agreement type: Ponder [11] allows to capture agreements about *modal constraints* (agreement type 2). Both obligations and authorization policies are supported.

2. Agreement language

   (a) Conceptual model: The Ponder language is *policy*-based.

   (b) Computational model: Ponder comes with an approach for deriving implementable policies from high-level goals [3, 4] based on the *event calculus* [17].

   (c) Management concepts: Concepts for managing policies include *policy domains* for grouping objects to which the same policy applies and *hierarchical grouping* of related policies. Concepts for policy conflict management include support for dealing with *modality* and *application-specific* conflicts. Automatic detection of policy conflicts is also supported, again by using the event calculus as underlying formalism [4, 10].

   (d) Language technology: Ponder is designed as an *object-oriented* language and is statically typed.

3. Coordination middleware

   (a) Coordination middleware functionalities: With respect to coordination middleware functionalities, Ponder only supports *enforcement*, primarily enforcement of access control policies. It does not support establishment.

   (b) Distributed architecture: Policies are enforced by so-called management agents that impose the policy by *re-actively* controlling and adapting the behavior of the underlying application, mostly by intercepting messages. Management agents[38] are injected into the application in a *decentralized way*: management agents for obligation policies are typically injected at the subject side, while authorizations are enforced by management agents at the target side.

**Law-Governed Interaction**

1. Agreement type: Law-governed Interaction (LGI) [25] can be used to regulate *business interactions* between autonomous organizations or agents (agreement type 1), but it also allows expressing *modal constraints* (agreement type 2).

2. Agreement language

   (a) Conceptual model: The basic modeling concept of LGI is the *law*. Abstractly speaking, the concept of a law is in essence the same as a policy.

(b) Computational model: There is no *real distinction between the conceptual and the computational model* of LGI: A law is directly specified as a function that returns an implementable action for every possible *event* that might happen at a given software object. These events must directly refer to methods or changes of *internal states* of software objects. This makes that the law specifications are directly coupled to a specific software system and therefore arebrittle to evolution of the software code.

(c) Management concepts: Similar to Ponder, LGI also supports *agent grouping* and *hierarchical grouping of related laws* [2].

(d) Language technology: Laws are implemented using *Prolog* due to its expressive power and its relatively widespread usage. *Reflection* is also used heavily as an implementation technique.

3. Coordination middleware

(a) Coordination middleware functionalities: LGI does not focus on establishment. With respect to execution, LGI focuses only at policy *enforcement*.

(b) Distributed architecture: A law is enforced by a set of trusted controllers. These controllers are *reactive* and for each agent there is a separate local controller (*decentralized*).

**The Business Contract Language and Architecture**

1. Agreement type: The Business Contract Language (BCL) [24] is targeted towards regulating *business interactions* between organizations and allows defining *modal constraints* (agreement types 1 and 2).

2. Agreement language

(a) Conceptual model: The basic modeling concept of BCL is the notion of *community*. Communities are in fact contract templates that already contain general contract behavior that is essential, and so cannot be varied, and those parts that can be expected to vary.

(b) Computational model: The behavioral concepts of BCL include *event patterns* (e.g., Sequence of Events, Disjunction of events and Event causality), *internal states* and their changes in response to the events, and *event types* to be created when certain conditions have been matched, e.g. creation of contract violation or contract fulfillment events.

(c) Management concepts: Except support for dealing with *application-specific conflicts* such as conflict of duty, BCL offers no further management concepts.

(d) Language technology: BCL is implemented using a *proprietary* language technology.

3. Coordination middleware

(a) Coordination middleware functionalities: As underlying contract framework, a generic Business Contract Architecture (BCA) is proposed. The core of this contract architecture supports contract execution only; contract *establishment is delegated to additional components*. Thus the core of BCA silently assumes that agreed upon contracts are stored in a contract repository. For contract execution, BCA supports contract *monitoring* and *enforcement* only. It does not provide an enactment component by means of which organizations can dynamically create the necessary infrastructure and allocate the necessary resources to exploit a contract.

(b) Distributed architecture: BCA is a flexible architecture; event interception and controller functions can be implemented *both centralized or decentralized*. The Contract Monitor component performs *re-actively* evaluations of contracts to determine wether parties' obligations have been satisfied or whether there are violations to the contract.

**Contract-driven creation of virtual enterprises in Crossflow**

1. Agreement type: The Crossflow project [15, 14] presents a contract architecture that is mostly focussed on *agreement type 1*, i.e. regulating business-level interactions between organizations, especially in the context of cross-organizational workflow management.

2. Agreement language:

(a) Conceptual model: Crossflow views contracts as the central theme that runs throughout the enterprises' life cycle. A contract specifies an *abstract workflow* for regulating the interactions between the business partners, and procedures for monitoring workflow execution and adjusting the workflow in case of deviations to the contract.

(b) Computational model: In order to turn these contracts into implementable actions, a blueprint for contract enactment (called the *Internal Enactment Specification*) is prepared, one blueprint for each participating organization.

(c) Management concepts: No additional management concepts are offered.

(d) Language technology: The language technology used in the Crossflow project is *XML*.

3. Coordination middleware

   (a) Coordination middleware functionalities: The focus of Crossflow lies mostly on *contract negotiation, contract enactment and contract dismantling*. An unusual approach to *contract monitoring and enforcement* is taken however: contracts themselves may contain clauses for how the behavior of the service provider can be monitored by the service consumer (and vice versa), and what rectifying actions are possible.

   (b) Distributed architecture: At the architectural level, the contract framework consists of two building blocks: (1) a Virtual Market (used to find business partners and negotiate contracts dynamically) with as *central* component a matchmaking engine and (2) an Enactment Infrastructure (needed to enact (i.e., set-up, execute, terminate) the contracted service. *Both reactive and proactive contract execution* is supported.

**Electronic contracts based on FSM's**

1. Agreement type: Work as part of The Tapas project [36] focusses on agreement types 1 and 2: *business interactions* and *rights and obligations* of the signed organizations.

2. Agreement language

   (a) Conceptual model: Contracts are modelled in Tapas as a set of *finite state machines*(FSMs), one for each contracted party [28].

   (b) Computational model: Similar to the above Law-Governed Interaction approach there is *no real distinction between the conceptual model and the computational model* of contracts in Tapas: FSM's are directly linked to implementable actions. The states of the FSM correspond with the progress of the execution or fullfilment of the contract; the input symbols are events that trigger the contract; and the output symbols correspond with the set of obligations and rights that the party is respectively subject to, as well as what actions the party has the right or obligation to perform.

   (c) Management concepts: No additional management concepts are supported.

   (d) Language technology: Contracts are directly specified using the *FSM formalism*.

3. Coordination middleware

   (a) Coordination middleware functionalities: The work covers the following coordination functionalities: (i) *contract validation* to detect and remove ambiguities between the clauses of a contract, and (ii) *contract monitoring* with the focus on capturing non-repudiable evidence of the actions performed by the business parties.

   (b) Distributed architecture: The contract framework is deployed in a *de-centralized* manner. Furthermore, the architecture is *pro-active* as the FSMs controls which operations are invoked on the contract objects.

**GlueQoS**

1. Agreement type: GlueQoS [41] is a middleware-based approach to match, interpret, and mediate QoS requirements of clients and servers. The quality of service requirements are mostly related to *QoS features* such as security, reliability and performance (agreement type 3).

2. Agreement language

   (a) Conceptual model: The basic modeling concept of GlueQoS is the *QoS policy*. The QoS policy is essentially a *contract* that describes what is the combination and configuration of QoS features (e.g. security) that is acceptable for a client and a server system.

   (b) Computational model: For implementing an agreed QoS policy, QoS features are directly implemented using aspects that intercept the necessary events and state changes by means of a generic join poin model.

   (c) Management concepts: With respect to management concepts, GlueQoS allows mediating *application-specific conflicts* between a client and a server organization with respect to the QoS policies.

   (d) Language technology: Client and Server policies are specified using the *WS-Policy standard* [12].

3. Coordination middleware

   (a) Coordination middleware functionalities: GlueQoS focusses on *contract negotiation* and *contract enactment*. It does not focus on contract execution.

   (b) Distributed architecture: When the client requests a session to the server, a QoS policy for that session will be computed at the client-side, by means of a matchmaking engine (*decentralized architecture*). An agreed QoS policy is enacted by means of deploying aspects. As contract monitoring and enforcement is not the focus of this work, no controller or event interception is necessary.

**T-BPEL: Transaction policies for service oriented computing**

1. Agreement type: T-BPEL [39], which is related to GlueQoS, supports advertising and negotiating between

different *transaction protocols* (such as direct transaction processing, queued transaction processing and compensation-based transaction processing). The focus lies thus on agreement type 4.

2. Agreement language

   (a) Conceptual model: T-BPEL introduces the concept of *transaction coupling mode* which is a *contract* about the used transaction protocol(s) between a web service provider and an orchestrating BPEL business process that invokes the web service. The preferred transaction protocols of the business process and the partner web services are expressed as by so-called policies that are directly attached to them.

   (b) Computational model: Client and provider policies as well as transaction coupling modes can be gradually refined from abstract transactional modes to concrete implementable protocols (e.g., WS-ReliableMessaging specification). The BPEL process itself is then transformed into a Java implementation (exposed as a SOAP-based service) that uses the selected transaction protocols.

   (c) Management concepts: No additional management concepts are provided, except that *conflicts between different policy assertions* can be mediated.

   (d) Language technology: Client and provider policies are described using the *WS-Policy language*. [12].

3. Coordination middleware

   (a) Coordination middleware functionalities: The underlying contract framework supports (i) *validation of client policies*, (ii) *contract negotiation* by means of policy matching, and (iii) *contract enactment*.

   (b) Distributed architecture: The distributed architecture of T-BPEL is *decentralized*, and very similar to GlueQoS, the distinction between pro-active or reactive is irrelevant.

# Design of Services as Interoperable Systems - An E-Commerce Case Study

Stephan Kassel
*Westsächsische Hochschule Zwickau, Institute for Management and Information,*
*PSF 201037, 08012 Zwickau, Germany*
*Stephan.Kassel@fh-zwickau.de*

## Abstract

*In this paper, some foundations of a decision support model for a full-service e-commerce provider, providing a SAAS (software as a service) business model, are presented. The decision model is targeted to explicitly address interoperability issues, to provide a simple but powerful communication aid for the negotiations between service provider and customer, and to aim in automatically composing reliable software systems from service components.*

## 1. Introduction

In the last years, e-commerce vendors changed their business model, from the original selling of technology like e-commerce systems and corresponding integration technology to the company's legacy systems, to more complex technological services, like integration with public market places [1], or even business services, like warehousing, e-marketing, or invoicing.

This has led to more complex e-commerce systems with a richer set of features, but hand in hand with the need of enhanced customizing services for e-commerce platforms. In this increasingly difficult business, leading e-commerce vendors have adopted to customer's need by two very different strategies.

One strategy lies in providing even more business functionality, leading to so called "full service e-commerce". The e-commerce company not only provides software for selling goods on the world-wide web, but enhances this software with additional services. This is possible, because there are a restricted number of services, which are widely recognizes as important by most players on the e-commerce market. These services have a high degree of modularity, leading to an encapsulation metaphor, this defining individually usable business functionality. The processes, which have to be defined in an e-commerce application, can be synthesized bottom-up from the services, which should be integrated into the overall solution. Common examples for these kinds of services are

- hosting services, consisting of providing hardware systems for e-commerce. together with server rooms, internet connections etc.,
- inventory management, providing warehousing, storage management, packaging, and delivery of goods,
- online marketing, providing e-mail campaigns, individual marketing programs, affiliate programs, etc.,
- search engine optimization, consisting of keyword management, special marketing campaigns, link placement, etc.
- multimedia presentation of products,
- payment services and loyalty programs, etc..

The second strategy to deliver added value to the customers, lies in the provisioning of software as a service (SaaS). This means, that the e-commerce systems are capable of utilizing standards interfaces to be able to call and provide services, which adopt to the service-oriented architecture (SOA). Via webservices mechanisms, centralized "yellow pages" for service registration, and flexible provision / calls of services, the e-commerce systems are evolving to an integration platform for complex business processes. Thus, business process management (BPM) and e-Commerce are coming together to provide more complex services for the end customers of the e-commerce system, and to integrate the business of the end customers with the business of the company owning (or leasing) the e-commerce system.

Both strategies of e-commerce vendors lead to a higher demand for explicit interoperation between platform owner and customer, to seamlessly integrate the service processes, and to commonly define the needed level of interoperability. The business processes have to be crisply defined, at least those processes, which are directly part of the interoperation.

## 2. The case study

### 2.1. Business area of the industrial partner

The industrial partner of the project, which is now in the definition phasis, is a medium-sized Canadian enterprise with a german subsidiary. The main business of the german branch of the company lies in integration and interoperation services for customers, who want to gain expertise in e-commerce. The founders of the german business have a strong background in e-commerce. The integration business of the company comprises of selling goods via different internet channels (dedicated shop systems as well as public marketplaces), by integration of the customer's warehouse or ERP systems and defining the interoperation processes for the e-commerce component. Thus, the industrial partner provides some scalable e-commerce services, starting with just a technical integration, but reaching up to a complete set of business definition and interoperation services.

This kind of business demands for a dynamic pricing model, which has to take into account the complexity of the service, the number of services provided, the amount of business delivered on the e-commerce channel, the dedication of hardware and software to the special customer, and other business related information.

Although the pricing model is very complex, there is no support in finding the right price for a customer 'till now. The pricing is done in an iterative way with several sessions between e-commerce provider and the customer, who wants to use the e-commerce services.

### 2.2. A model for the pricing of services

The ad-hoc pricing of the services delivered by the e-commerce provider holds some serious business risks:

- the components are not priced well,
- the needed and paid performance of the e-commerce system for the customer has to be estimated very roughly,
- the pricing model is not very transparent, so the customer cannot gain trust into the service provider, and
- if there are any problems in the delivery of the service, the re-negotiation is always a negotiation of the complete set of services, because they cannot be separated at all.

There is a need for some delta pricing models, because most of the service constituents are already pre-built, they only have to be re-arranged and customized to the special customer.

So the research, which has to be done, consists of building a specialized decision model, which can be used in the early phases of a customization project. This decision model has to take into account that the software which has to be built, mainly consists of pre-built business components which are already suited to provide a rich set of services to the customer. These services definitely have to be tailored, but the variation of the services normally are not too numerous.

## 3. State of the art

There are a lot of research domains, which are touched by the task of the case study. So, the overview of the state of the art of these areas has to be very short, and only a few fundamental work is considered, which could be helpful in providing ideas and theories for the solution of the problem.

### 3.1. Decision support systems

Decision support systems are delivering a support in making semistructured or unstructured decisions. To achieve this, they consist of a flexible tool set with analyzing capabilities. There is a number of well-known systems, falling mainly into two classes [2].

One class of systems is model-driven. They are standalone systems, which are specifically designed for their tasks. They have a deep model of the overall problem, and they are most of the times standalone systems. Most of these systems are defined with artificial intelligence methods. Expert systems are a well-known example of this class if systems. The analysis capability of these systems is built on strong theories and models, combined with user interfaces designed for end users. The disadvantage of this system class lies in the complexity of the models. Building an explicit model with a strong theory is tedious, error-prone and often the users are not capable of building and maintaining these systems.

The second class of systems is data driven. Their purpose lies in analyzing large pools of data. These data are compiled into large data warehouses. The analyzing capabilities of these systems are based on flexible ways to view and combine mass data to analyse long-term trends. Additionally, the analysis of the data can be partly automated via data mining. This system class is very useful for strategic decisions and overall steering of a company, but it is not detailed enough to provide support for decisions like the

pricing of a customer-specific solution. (This is the disadvantage of lacking an explicit user model.)

Nowadays, there is a trend from decision theories and systems to decision aiding methodologies resulting in decision aiding systems [3]. In systems following this trend, not the decision itself is done by the system, but the system is assisting in making rational decisions. The decisions of the case study, which is presented here, has a simple, but powerful model. This model is based on some common methodologies widely used in economics and business. This kind of aiding system is needed to provide a solution to the issue of the pricing services. These issues are:

- to explicitly address the different levels from business processes to the programming level,
- to provide a negotiation aid for the business level,
- to directly connect business functionality and feature sets with pricing of the solution,
- to explicitly include business interoperability in the model of the business functionality,
- to provide an integrated view of the problem, leading to a consistent behaviour suited to the needs of the customer,
- to model user requirements as a mediation process,
- to automatically compose service components as well as reliable software systems, and
- to explicitly address interoperability.

### 3.2. Service interoperability

The area of service interoperability is not very well defined. Although there is a wide set of literature in the area of interoperability (e.g. [4], [5], [6], or [7]), the majority of the research still concentrates on the technical levels of interoperability. Even the roadmap of interoperability research of the European Union [8] still has an emphasis on technical interoperability levels, although the importance for defining the business level of interoperability is explicitly stated. The term of service interoperability is hardly defined, but we clearly have to differentiate between the (very technical) definition of service-oriented architectures (SOA), which only comprises of low-level, automated more technical-oriented services, and the business-oriented terms of service industries, comprising of immaterial “products”, which is traditionally termed as service outside of the IT-industry.

There is some research done in the enterprise interoperability group, which is a good basis for the definition of complex service interoperability. Especially the work in the area of model-driven interoperability, like [9], [10] and [11], provides some good basis for an enhancement to service interoperability. There are some models of service interoperability existing, (e.g. [12] and [13]), which are very useful as a fundamental basis to solve the pricing problem. However, these models are still lacking simplicity in the practical usage, and there are severe problems, which are typical for service interoperability, discussed in [14]. An adaptive approach to solve these issues is presented in [15], but it is still very generic, and has to be adopted very carefully to be used in a specific context like the case study at hand.

## 4. Solution proposal

To solve the pricing problem which is sketched in section 2.2, an architecture is used, which is similar to the idea of model-driven interoperability [9] with the enhancements for an overall service architecture, that have been proposed by Xu et al. in [13]. This architecture shows very promising characteristics, and therefore our idea is based on it.

But there still is a lack in the mapping from service demands to the pricing, which has to be done here. The classical way of the mapping goes from the demands to some fundamental service component specification, and later to the implementation.

We propose a different architecture, which is very useful, if the services can be pre-defined. This is the case in our example, so we introduce an additional modelling level, which is based on some very common economic model, namely decision trees.

The top level of our architecture consists of some kind of service grid, where the different business services, which are recurrent parts of an e-commerce platform, are presented to the customer, who can choose a subset, which he wants to be included in the service offering.

For each of the choosen services, a decision tree exists, which is a consistent way of defining business numbers or choosing special variations of the services. Thus the solution can be defined very well by walking through the decision trees and choosing a leaf for each tree, which is part of the offering.

At this point, there are two steps, which are following now. One step is the pricing, which can be directly derived from the choosen leaves. The other step is the construction of business processes, which can be partly defined from the leaves. By choosing from the service grid, and making decisions about the specific peculiarities of the services, a combined price for the overall solution is found and, at the same time,

the overall business processes can be constructed by the combination of the service process parts from the different leaves.

This leads to a model-driven way of defining the services, which can be enhanced in the common way of BPM. So the business processes are enhanced later to include the technical details of the service, and some workflow engine can automate these processes. In the concrete example, there is one more step, by compiling the services to be runnable with a high performance, if needed.

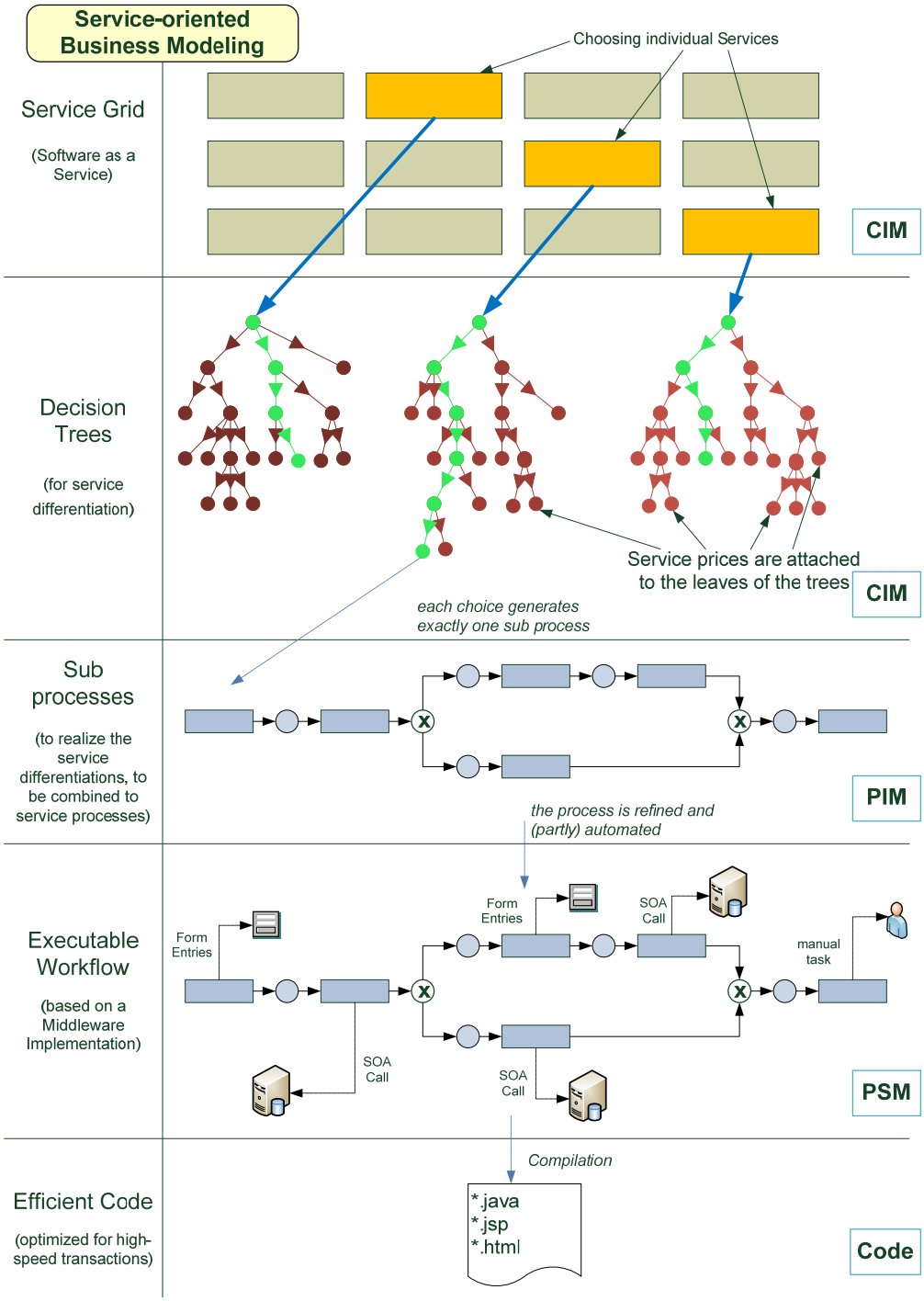



**Figure 1. Service architecture**

## 5. Conclusions

In this paper, we have discussed a serious business problem, coming out of an interoperability context. The business levels are explicitly addressed, and the interoperability of the business is part of the model. But also dynamic interoperability on a systems level can be included into the model.

The views of the model are integrated. The architechture is directly based on common standards like MDA and model-driven interoperability (MDI), as well as the SMDA, and it seems to fit well into these architectures. In the proposed architecture, the main focus is set on the different business levels, but there is an explicit proposal of some automatic mapping / model transformation between the different architectural levels (in a restricted domain, so we don't have to explicitly deal with semantic mapping of differing ontologies, which considerably simplifies the proposed architecture, compared to semantic-transforming models).

Our further research deals with a reference implementation of the upper system levels, to get feedback from practical use, and to be able to refine the proposed architecture. We have a special focus on the handiness of the system for practical purposes to achieve a real business value of the solution. Therefor, we want to utilize transactional cost theory to evaluate the theoretical appropriateness of the solution for some restricted usage areas. There is some chance that the restrictions of the domain can help to provide some useful system for business interoperability.

## 6. References


[1] S. Kassel, and K. Grebenstein, "Adapting manufacturing to customer behavior. Lessons learned from trading goods on public market places", *Proceedings of PROLAMAT 2006*, Shanghai (China), 2006.

[2] K. C. Laudon, and J. P. Laudon, Management Information Systems, Prentice Hall, 10th ed., 2007.

[3] A. Tsoukiàs, "From decision theory to decision aiding methodology", *European Journal of Operational Research*, Vol. 187 (2008), pp. 138-161.

[4] G. Doumeingts, J. P. Müller, and G. Morel, *Enterprise Interoperability: New Challenges and Approaches*, Springer, Berlin, 2007.

[5] R. J. Gonçalvez, J. P. Müller, K. Mertins, and M. Zelm, *Enterprise Interoperability //: New Challenges and Approaches*, Springer, London, 2007.

[6] K. Mertins, R. Ruggaber, K. Popplewell, and X. Xu, *Enterprise Interoperability III: New Challenges and Industrial Approaches*, Springer, London, 2008.

[7] P. Bernus, and M. Fox, *Knowledge Sharing in the Integrated Enterprise: Interoperability Strategies for the Enterprise Architect (IFIP)*, Springer, Berlin, 2007.

[8] M-S. Li, R. Cabral, G. Doumeingts, and K. Popplewell, *Enterprise Interoperability research roadmap*, July 2006, http://cordis.europa.eu/ist/ict-ent-net/ei-roadmap_en.htm

[9] P. Girard, and G. Doumeingts, "GRAI-Engineering: a method to model, design and run engineering design departments", *International Journal of Computer Integrated Manufacturing*, Vol. 17 Issue 8, 2004, pp.716-732.

[10] S. Blanc, Y. Ducq, and B. Vallespir, "Evolution management towards interoperable supply chains using performance measurement", *Computers in Industry*, Vol. 58 Issue 7, 2007, pp.720-732.

[11] K. Mertins, T. Knothe, and F.-W. Jäkel, "Interoperability – Network Systems for SMEs", in [6], pp. 511-520.

[12] N. Protogeros, D. Tektonidis, A. Mavridis, C. Wills, and A. Koumpis: "FUSE: A Framework to Support Services Unified Process", in [6], pp. 209-220.

[13] X.F. Xu, T. Mo, and Z. J. Wang, "SMDA: A Service Model Driven Architecture", in [5], pp. 291-302.

[14] Z. Wang, and X. Xu, "Ontology-based Service Component Model for Interoperability of Service Systems", in [6], pp. 367-380.

[15] K. Popplewell, N. Stojanovic, A. Abecker, D. Apostolou, G. Mentzas, and J. Harding, "Supporting Adaptive Enterprise Collaboration through Semantic Knowledge Services", in [6], pp. 381-393.

# Improvement Model for Collaborative Networked Organizations

Dr. Josef Withalm
*Siemens IT Solutions and Services,*
*A-1101 Vienna, Austria*
*josef.withalm@siemens.com*

Walter Wölfel
*Siemens IT Solutions and Services,*
*A-1101 Vienna, Austria*
*walter.woelfel@siemens.com*

## Abstract

*The abstract is to be in fully-justified italicized text, at the top of the left-hand column as it is here, below the author information. Use the word "Abstract" as the title, in 12-point Times, boldface type, centered relative to the column, initially capitalized. The abstract is to be in 10-point, single-spaced type, and up to 150 words in length. Leave two blank lines after the abstract, then begin the main text.*

## 1. Introduction

Small and Medium Enterprises (SMEs) have huge improvement potential both in domain and in collaboration / interoperability capabilities. Before implementing respective improvement measures it's necessary to assess the performance in specific process areas which we divide in domain (e.g. tourism) and collaboration oriented ones.

In ECOLEAD [4] reference models were developed for companies, which are interested in joining a collaborative network depending on its characteristic (i.e. VBE (Virtual Breeding Environment), PVC (Profession Virtual Community), VO (Virtual Organization)), in which this organization may reside actually. These reference models encompass checklists, templates, interfaces (to exist in software), tools, specifications, architectures, SW components or services. These artifacts may be applied to facilitate the agreement concerning business strategies, business models and above all business processes for organizations which are due to join a collaborative network. If business strategies are mentioned, subtasks (i.e. legal issues, trust building (overcoming the competition issue)) must be covered too. Each company is then empowered to configure / implement essential business processes to join a collaborative network in the respective phase. To assess this claimed collaboration behavior the process areas of CMMI are evaluated insofar, if they are appropriate or should be extended / modified concerning collaboration issues.

One of the most important issues, which are tackled by the EU 7th Framework project COIN, is the synopsis of EC (Enterprise Collaboration) and EI (Enterprise Interoperability.

Enterprise Collaboration comes from a business perspective and identifies the process of enterprises - mainly SMEs - to set-up and manage cross-enterprise win-win business relations in response to business opportunities.

Enterprise Interoperability originates by the ICT world and identifies a capability of enterprise software and applications to exchange information and to mutually understand the information exchanged at the level of data, applications, processes and enterprise models involved.

Both in EC and in EI the behavior of organizations regarding interoperability must be improved. First step before tackling measures for improvement is an actual assessment of the mentioned behavior of these organizations.

There are in theory many assessment methodologies developed and some of them are widely accepted within industry. The most important of them are on the one hand CMMI and on the other hand EFQM (European Foundation of Quality Management) which will be discussed in more detail in chapter 2. Concerning quality criteria there is also a prevailing methodology - namely BSC (Balanced Score Cards) - applied in industry.

But what are the prerequisites for organizations to interoperate together?

Within industry it's widely recognized that before organizations tackle the issue of interoperation important business oriented issues must be solved. Among them are issues as a common strategy and common business models for the questionable organizations. Having solved these tasks and having set respective quality criteria in form of BSC (there they really are focusing what is their strategy about customers, business processes, employees, and finances) the next focus is shaping out respective business processes among the partners which should of

course follow the defined business strategy and business models.

Exactly there seems the borderline between Enterprise Collaboration (which was very well defined and worked out in the EU project ECOLEAD) and Enterprise Interoperability (which on the other hand was partly implemented by ATHENA [1]).

Of course the modeling and specification of business processes is most important.

A later implementation must take into account a lot of specifics of the concerned organizations i.e. legacy software, data repositories, ERP systems and so on. A further important issue is the separation in horizontal and vertical business processes. So the interoperability issue may also be tackled more generic and must be implemented for the concrete application.

Other issues are the quality criteria for interoperability. Many of them are already tackled by the so called non functional requirements as performance, usability, availability…

## 2. Theory

In the following section, the theoretical background of the paper will be presented on the one hand to understand the basic principles of the CMMI® and on the other hand to point out why small and medium enterprises dominated industry might be an ideal playground for CMMI® applications.

### 2.1. Capability Maturity Models

Primarily CMM® [6] was developed at the SEI (Software Engineering Institute) on behalf of the Department of Defense (DoD) in the U.S.A. in order to establish a model that identifies mature and capable enterprises in the market that are able to manage SW projects for the DoD. In the meantime the original intention of CMM® changed: it can now be interpreted as an instrument to find strengths and weaknesses of organizations in specific process areas where appropriate improvement measures should be implemented [3].

In the current marketplace, there are maturity models, standards, methodologies, and guidelines that can help an organization to improve business operations. However, most available improvement approaches focus on a specific part of the business and do not take a systemic approach to the problems that most organizations are facing. By focusing on improving one single area of a business (e.g. such as marketing or distribution), area focused models unfortunately have perpetuated the stovepipes and barriers that exist in organizations [3].

Capability Maturity Model® Integration (CMMI®) provides an opportunity to avoid or eliminate these stovepipes and barriers through integrated models that transcend disciplines. CMMI® for Development consists of best practices that address development and maintenance activities applied to products and services as well as product's lifecycle conception, delivery and maintenance. Its main emphasis is on 'building' and maintaining the overall product and service bundle.

CMMs focus on improving processes in an organization. They contain essential elements of effective processes for one or more disciplines (such as quality management or yield management) and describe an evolutionary improvement path from ad hoc, immature processes to systematic, well-structured mature processes with improved quality and effectiveness [3].

**2.1.1. Constellations.** This improvement framework can also be applied to other areas of interest, where the framework groups best practices into what is called "constellations." A constellation is a collection of CMMI components that are used to build models, training materials, and appraisal documents.

Recently, the CMMI® model architecture was improved to support multiple constellations and the sharing of best practices among constellations and their member models. Work has begun on two new constellations (see Figure 1): one for services (CMMI® for Services) and the other for acquisition (CMMI® for Acquisition).

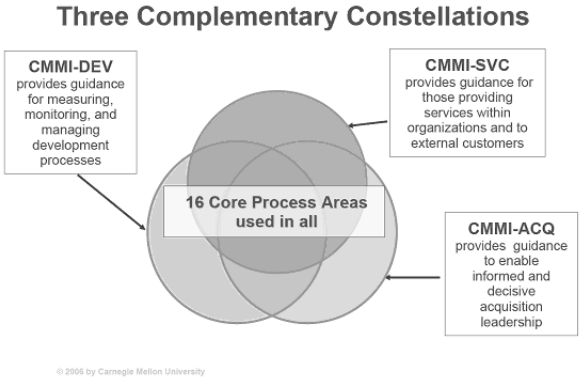



**Figure 1: CMMI® constellations [3]**

**2.1.2 Representations.** *Continuous representation* enables organizations to select a process area (or group of process areas) and improve related processes. This representation uses capability levels to characterize improvement relative to an individual process area.

The *staged representation* uses predefined sets of process areas to define an improvement path for an organization. This improvement path is characterized by maturity levels. Each maturity level provides a set of process areas that characterize different organizational behaviors [3].

**2.1.3. Process Areas.** A process area is a cluster of related practices in an area that, when implemented collectively, satisfy a set of goals considered important for making improvement in that area.

**Table 1: CMMI® process areas**

| | |
|---|---|
| Causal Analysis and Resolution | Project Monitoring and Control |
| Configuration Management | Project Planning |
| Decision Analysis and Resolution | Process and Product Quality Assurance |
| Integrated Project Management | Quantitative Project Management |
| Measurement and Analysis | Requirements Development |
| Organisational Innovation and Deployment | Requirements Management |
| Organisational Process Definition | Risk Management |
| Organisational Process Focus | Supplier Agreement Management |
| Organisational Process Performance | Technical Solution |
| Organisational Training | Validation |
| Product Integration | Verification |

There are 22 process areas, presented in Table 1 in alphabetical order by acronym (see [3]).

## 2.2. SMEs in tourism: the need for CMMI® solutions

Tourism seems to be an appropriate example, where improvements both in collaboration but also service provisioning are extremely important. That's why the envisaged assessment method could benefit SMEs in tourism (SMTEs)

As mentioned above, CMMI® can support the improvement of management processes. This also holds true for service enterprises: However, the majority of leisure and tourism businesses are small or medium-sized and a large number of them are family businesses [2].

SMTEs have not merely strategic disadvantages as described in [9].

Nevertheless the majority of disadvantages of SMTEs exist because small enterprises suffer from lacking economies of scale and scope. This results to high fixed costs and relatively high costs per unit.

These advantages and disadvantages contour strategy alternatives for small- and medium-sized tourism enterprises. Thus possibilities to reduce cost of production hardly exist, while on the other hand differentiation strategies often fail due to SME owner/managers' short term (myopic) thinking and lack of market research capabilities. It should therefore be possible to communicate the benefits (efficiency and effectiveness) of strategic co-operations within given destinations among small- and medium-sized enterprises and entrepreneurs with respect to these new forms of management [7]. Advantages of local, interregional and / or national co-operations could provide for additional resources, reduced costs and risks in product development, new markets, improved qualifications and/or increased competitiveness.

When having a closer look at the knowledge and qualification hurdle recent industry developments revealed gaps in special SMTEs' knowledge areas. Much of the differences in the innovation behaviour between industrial and service sectors are associated with the different nature and characteristics of services-production and –marketing. Six aspects of services production/marketing in the tourism sector stand out in particular, i.e.

- Intangibility of services and the associated quality uncertainty of customers,
- Simultaneity of production and consumption of services,
- Non-storability of services,
- High risks/cost associated with fluctuations in the rate of capacity utilization,
- Difficulty in correctly forecasting consumer needs and preferences for hospitality and tourism services,
- Sensitivity of services production to increases in labour cost on account of labour intensity of services production [5]

These characteristics of tourism services lead to an increasing demand for entrepreneurial and managerial qualifications in the following management areas:

- Quality management
- Capacity Management
- Product development
- Human resource management
- eCommerce

## 3. Methodology

For adapting the CMMI® theory we will follow a two step approach with special focus on determination of *collaboration/interoperability oriented* as well as *domain oriented (i.e. tourism) process areas*.

In the course of the presentation we are constraining on EC oriented process areas.

### 3.1. Determination of collaboration oriented process areas

In the first step the compatibility of the 22 process areas of CMMI® (see 2.1.3) with the CNO will be analyzed. Especially it will be analyzed, if they are appropriate or they should be extended or modified.

**Table 2: Relevant process areas for CNOs**

| | |
|---|---|
| Causal Analysis and Resolution | Organisational Process Performance |
| Configuration Management | Organisational Training |
| Decision Analysis and Resolution | Process and Product Quality Assurance |
| Measurement and Analysis | Requirements Development |
| Organisational Innovation and Deployment | Requirements Management |
| Organisational Process Definition | Risk Management |
| Organisational Process Focus | Supplier Agreement Management |

ECOLEAD [10] addresses the two most fundamental and inter-related focus areas that are the basis for dynamic and sustainable networked organisations: Virtual Breeding Environments, and Dynamic Virtual Organisations.

Concerning the CMMI® process areas Dynamic Virtual Organisations behave more or less like distributed projects and hence the most process areas of CMMI® may be applied. Completely different is the situation in the case of Virtual Breeding Environments. In this case every process area must be evaluated individually. The most concerned areas are listed in Table 2.

Exemplary the extension respectively modification of one process area will be discussed more detailed. Thereto the process area Configuration Management (CM) was selected.

The Configuration Management process area supports all process areas by establishing and maintaining the integrity of work products using configuration identification, configuration control, configuration status accounting, and configuration audits. The work products placed under configuration management include the products that are delivered to the customer, designated internal work products, acquired products, tools, and other items that are used in creating and describing these work products. Examples of work products that may be placed under configuration management include plans, process descriptions, requirements, design data, drawings, product specifications, code, compilers, product data files, and product technical publications.

In Figure 2 the following bubbles will be skipped: *Create a Release Baseline* and *Track Change Requests*. This means that *Change Requests* and *Change Request Database* will be omitted.

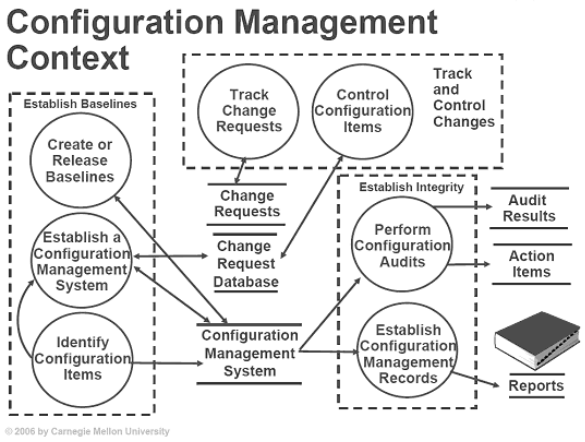



**Figure 2: Process area configuration management [3]**

Whereas in Configuration Items the following work products: code, compilers, product data files, and product technical publications should be removed by these ones: Profile, the History, the Evidence, the Bag of Assets, the VBE Governance, the VBE Values System, and the Trust. Moreover also Strategic Alliance and the Opportunity-based CNO should be taken into account.

Furthermore the process areas Configuration Management and Requirement Management are more strongly interlinked. Especially Requirement Management's focus will shift from "classical" functional requirements (competences of the network partners) to non-functional requirements concerning social competences and trust.

### 3.2. Determination of tourism oriented process areas

The pilot project SMART-UP cut in on the weaknesses of SMTEs as described above and

developed an internet learning and know-how transfer platform for (owner) managers in the tourism industry [8]. Four modules have been developed according to market-research on qualification needs of SMTE entrepreneurs: Quality Management, Yield Management, Human Resource Management, Product Development and eCommerce. In the following, the area of quality management serves as a case study example to adapt a CMM process in SMTES.

In a first step SMTEs have to identify needs to select a process area by a mode of representation. For SMTEs both forms of representation are challenging, as in the case of continuous representation the SME owner or entrepreneur should be aware which qualification areas can be chosen as process areas. In addition, owner manager have to understand the dependencies between various process areas. For many SMTEs staged representation can be recommended because many owner managers do not really know where they can start with process improvements.

For small businesses many of these requests force them to radically change common patterns of management: constant learning has to be reported and discussed and an open as well as strategic and long-term oriented learning process is a prerequisite for the implementation of CMM in SMTEs.

## 4. Expected results

At the ENTER 2007 [11] it was shown that ECOLEAD concepts especially those of VBE, can be applied to Destination Management Organizations (DMOs) and hence improve their performance. DMOs tasks are to bundle and provide all tourism services in a region/destination. Therefore many organizations should cooperate, and the DMO should build a Virtual Breeding Environments (VBE). The basic idea of supporting DMOs in building VBE will be continued, as was announced at ENTER 2005 [10]. The following approach will be applied to DMOs of the specific region, which is involved in oncoming projects (as for instance Olympic Games, Ski Championships, EURO2008). Within the proposed DMO the building of virtual organizations is intended, providing ticketing, entertainment, travelling, and transportation, together with all organizations, which could be involved in such a mega event.

First experiences with demonstrators in different domains - when applying these ECOLEAD results in trials and take-ups (ECOLEAD 2004, http://www.ecolead.org) - have shown that the maturity level of SMEs is extremely different. So the best approach seems to develop the presented CMMI® approach and to support SMEs to implement the required concepts and solutions.

Afterwards the project team together with the DMO will be involved in preparation endeavors, and in the first phase all results how to build a breeding environment will be realized. In this phase, many of the formerly developed guidelines, checklists, tools can be applied in that sense that especially all questions concerning the business strategies – above all trust buildings and legal issues – and business models will accelerate the decision process, which organizations should/could participate.

Especially applying rules for trust building and business models will be facilitated after having assessed those SMEs in the tourism destination that should be considered. Not until the maturity level is assessed and the improvement measures are implemented negotiations by collaboration agents with all of them may be initialized as was presented in the ENTER 2007 [11] conference. Then it will become clearer, which organisations are appropriate to join the network?

Further research in the field of SMEs should answer the following questions:

- Are SMEs ready for the recommended assessments?
- Are SMEs able to implement the improvements?
- Is it possible to standardize the assessment procedure?
- Are there experienced experts both for assessments and improvements available?
- Is it possible to establish certification bodies for assessors?
- Is it possible to build services which belong to respective process areas of CMMI?

A specific research in COIN is dedicated to the last question above, which will be introduced briefly.

CMMI has a well defined structure (see Figure 3 below):

- Maturity levels
- Assigned Process areas
- Specific goal : which should be measurable
- Specific practices: which should be modelled and in concrete applications also implemented

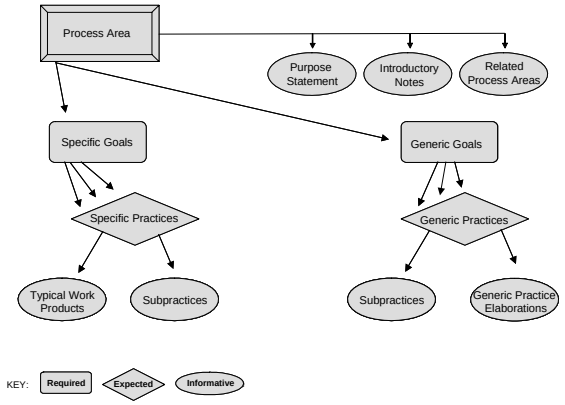



**Figure 3: Process area configuration management [3]**

Now in COIN the idea is the following one: both EC and EI should enable organizations the set up (creation) and operation of VO's which are built on business opportunities.

The borderline between EC and EI are from the business point of view the following: in EC the potential participating/interested organizations undertake endeavors to agree on common business strategy and business models. Only after this agreement has settled it makes sense to model common business processes.

Exactly that's the border crossing from EC to EI. As in EI the individual organizations will later on tailor/configure their existing business processes according the starting EI baseline services which will be developed by COIN.

Furthermore process areas - similar of those of the classical CMMI - will be defined with respective goals and practices.

Of course these process areas will on the one hand belong to EC (many "practices" are already developed in ECOLEAD), but must now fit to the respective structure of process areas in CMMI. These specific practices will be modelled as services.

Summarizing the tasks:

- EC and EI process areas will be defined in analogy to the classical CMMI.
- Per process area specific services which correspond to specific practices are modelled.
- CMMI assessment may be performed either classical or by a so called CMMI check.
- According the outcome it will be clear which process area met or partly met and more concretely which corresponding specific practices are not reached (don't forget: CMMI has specific goals for each practice).
- Now the concerned organization - respectively in case of EC the corresponding VBE - may add the respective services where the respective goal for the respective specific practice was not met.
- The adding of these services may be implemented in a new orchestration / choreography of the whole service platform either for the VBE / individual organization or only for the individual organization.

Of course all these endeavours about defining process areas must be synchronized with the COIN Generic Service Platform.

A further optimization of this process (of establishing process areas) would be to distinguish to which quadrant of BSC (see Figure 4) a new process area best fits.

That's one of an optimization effort another one would be to really find target values of the specific goals of an process area, so that it would be automatically measurable by BSC tools.

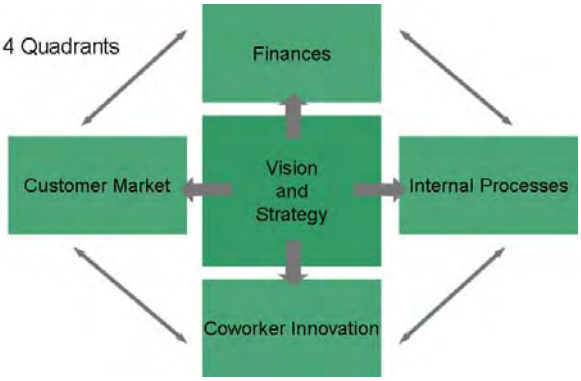



**Figure 4: Overview of BSC**

## 5. Conclusions and future work

The process to start the definition of a new constellation will be launched within COIN, the recently started FP7 successor of ECOLEAD, ATHENA and other EU projects concerning EC and EI. Both, COIN partners, EU project officer, and reviewers, all agree that CMMI represents the ideal framework to disseminate innovative and challenging results to those SMEs who demand and need this knowledge to better face future business management challenges.

Hence in COIN a prior goal is the adaptation and the concurrent examination of these process areas in collaboration with SMEs. However, in ECOLEAD it was an essential insight that domain specific process areas play a crucial role for the success of CMM applications. Therefore this modified CMMI should be applied on SMEs.

However, it seems to be an illusion to assume that a new constellation of CMMI can be defined in short term. Ongoing trials of these assessments on existing SMEs will enable a very pragmatic implementation process and prevent the establishment of scientific but non-practical process models which will not be applicable for SMEs.

Parallel to those activities initiatives regarding standardization are taken up. Experiences in the area of IT industry confirm that organisations are ready for these assessments not until a certification is available (e.g. ISO 9000).

Together with CMMI following activities have the potential to improve the performance of SMEs:

- Tourism process areas are partly developed in SMART-UP [8]. They can be an example for further evaluation and definition of domain oriented process areas (e.g. in service industry).
- Core Process areas of CMMI® will be extended concerning collaboration behaviour in the course of ECOLEAD and its presumable successor COIN.
- Capability maturity assessments enable SMEs a position fixing and consequently to an improvement of their business performance.
- Accomplishing better profits of the SMEs themselves and indirectly to the whole sector / region.
- Experts from universities and consultants have the possibility to become a certified assessor.
- Enabling new business opportunities for SMEs.
- Both collaboration-oriented and domain oriented process areas will assess the IT competences of the SMEs.
- Building of services which belong to respective process areas of CMMI.

## 6. References


[1] ATHENA, "ATHENA Home Page", ATHENA IP. http://www.athena-ip.org/ (last visited 2006).

[2] Buhalis, D. and Peters, M. (2006). SMEs in tourism In Buhalis, D., Costa, C. (eds.), Tourism Management Dynamics: Trends, Management and Tools, Elsevier: Amsterdam et al., pp. 116-129.

[3] Carnegie Mellon (2006). CMMI® for Development, Version 1.2. Pittsburgh: Carnegie Mellon Software Engineering Institute.

[4] ECOLEAD (2004). Contract for: Integrated Project – Annex 1: Description of Work. European Collaborative networked Organisations LEADership (IST-1-506958).

[5] Fitzsimmons, J.A. and M. Fitzsimmons (2000). Service Management. 3rd edition, McGraw-Hill: Boston.

[6] Humphrey, W. S. (1989). Managing the Software Process. Reading, MA: Addison-Wesley.

[7] Pechlaner, H., & Raich, F. (2004). Vom Entrepreneur zum "Interpreneur" - die Rolle des Unternehmers im Netzwerk Tourismus. In Weiermair K., Peters M., Pechlaner H. & Kaiser M.-O. (Eds.), Unternehmertum im Tourismus: Führen mit Erneuerungen,. Berlin: Erich Schmidt, pp. 123-138.

[8] Peters, M., Withalm, J., Weiermair, K. (2002). Small and Medium Sized Enterprises Alliance through Research in Tourism (SMART-UP). In Wöber, K.W., Frew, A.J. and M. Hitz (eds.), Information and Communication Technologies in Tourism 2002, Wien - New York: Springer Verlag, pp. 145-156.

[9] Peters, M., Withalm, J., Wölfel, W. (2008). Capability Maturity Models for SMEs and Collaborative Networked Organisations in Tourism. In O'Connor, Höpken W., Gretzel U. (Eds.). Information and Communication Technologies in Tourism 2008, Wien - New York: Springer Verlag, pp. 568-579.

[10] Withalm, J., Wölfel W., Supper C. (2005). Introducing Virtual Organizations to Tourism. Paper presented at ENTER 2005.

[11] Withalm, J., Wölfel W., Smolak I., (2007). Collaboration Agents. In Sigala M. et al. (Eds.). Information and Communication Technologies in Tourism 2007, Wien - New York: Springer Verlag, pp. 289-300.

# Towards a Business-IT Alignment Maturity Model for Collaborative Networked Organizations

Roberto Santana Tapia*, Maya Daneva, Pascal van Eck, Roel Wieringa
Department of Computer Science
University of Twente
P.O. Box 217, 7500 AE Enschede, The Netherlands
r.santanatapia, m.daneva, p.vaneck, r.j.wieringa@utwente.nl

## Abstract

*Aligning business and IT in networked organizations is a complex endeavor because in such settings, business-IT alignment is driven by economic processes instead of by centralized decision-making processes. In order to facilitate managing business-IT alignment in networked organizations, we need a maturity model that allows collaborating organizations to assess the current state of alignment and take appropriate action to improve it where needed. In this paper we propose the first version of such a model, which we derive from various alignment models and theories.*

## 1 Introduction

In modern organizations, business-IT alignment (B-ITa) is a hard problem that requires continuous attention. There is a considerable literature on measuring and improving B-ITa in single organizations (e.g., [19, 22, 24, 30]) but the problem of B-ITa in collaborative networked organizations (CNOs) has hardly been studied. Yet, the problem is important because improved B-ITa entails a more efficient use of IT in the CNO supporting the integration of enterprise applications and processes across organizational boundaries.

CNOs form the core of a new discipline [9, 10] that focuses on the structure, behavior, and dynamics of networks of independent organizations that collaborate to better achieve common goals. CNOs are characterized by being formed by organizations which have a pre-disposition to collaborate in order to attend a common interest using IT, and by being associated with effective coordination and shared decision making. These characteristics provide opportunities to generate commitment within markets which demand very quick, high-quality and cost-saving services from organizations.

In such a context, maturity models (MMs) are a suitable vehicle for CNOs to gain a deeper understanding of their current B-ITa, and to plan what steps to take toward better alignment. In this paper, we will define a conceptual framework, in the form of a MM for assessing and improving B-ITa in CNOs. Clearly, some other models and theories (e.g., [12, 27, 34, 44]) can also be used to understand aspects related to B-ITa in CNOs. Nevertheless, none of these models and theories covers all the necessary domains that need to be considered by CNOs when achieving B-ITa. This motivated us to adopt the position that these models and theories might be used as starting points in cross-organizational B-ITa initiatives, but they need to be integrated.

The contribution of this paper is twofold. First, we present a systematic approach for the development of a MM in the form of a MM process model. Second, we suggest a state-of-art literature-based MM that can be used in a CNO setting to assess processes related to those B-ITa attempts which integrate multiple perspectives. The paper is organized as follows: Sect. 2 provides background on the concepts we use. Sect. 3 presents related work emphasizing the needs of MMs in CNOs. In Sect. 4 we describe the MM we are developing. Furthermore, we discuss its adoption in CNOs and its preliminary evaluation in Sect. 5. Finally, Sect. 6 summarizes our conclusions and presents our immediate future work.

## 2 Conceptual Background and Definitions

### 2.1 Business-IT Alignment

For the purpose of our research, we define B-ITa as the *process to make IT services support the requirements of the business*, whether such services are individually or collaboratively offered. We do not consider alignment as a steady

*Supported by the Netherlands Organization for Scientific Research (NWO) under contract number 638.003.407 (Value-Based Business-IT Alignment).

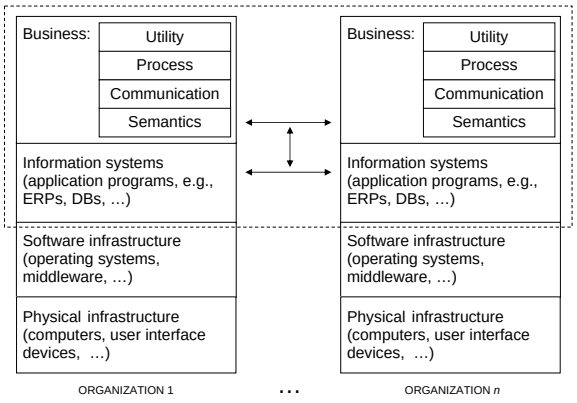



**Figure 1. Business-IT alignment framework in CNOs (adapted from [51]).**

state but as a process that needs to be performed continuously. By 'IT services' we mean services offered by computerized information systems. By the 'requirements of the business' we mean the systems requirements derived from analyzing the goal(s) of the CNO. We will focus on operational B-ITa, which consists of aligning the operational activities of IT systems and people to each other so that optimal IT support for business requirements is achieved. This contrasts with strategic B-ITa, where business and IT goals and policies are decided without fixing operational details [13, 30, 42].

We analyze the B-ITa concept in CNOs based on the scheme shown in Fig. 1. The horizontal layers classify entities in a service provisioning hierarchy in a business: physical entities provide services to a software infrastructure, which provides services to information systems, which provide services to businesses. In the business layer, we take four views on businesses: businesses provide services that have a utility, they perform processes to provide these services, they communicate with one another as part of performing these processes, and while doing that, they exchange data that has semantics. Participating organizations in a CNO need both to fit the different entities (horizontal arrows) as well as to address B-ITa (vertical arrow). Our interest is in the upper two layers of the framework (area delimited by the dotted line), because there is where business and IT alignment in CNOs takes place.

## 2.2 Collaborative Networked Organizations

We define a CNO to be any *"mix-and-match" network of profit-and-loss-responsible organizational units, or of independent organizations, connected by IT, that work together to jointly accomplish tasks, reach common goals and serve customers over a period of time* [35]. Virtual enterprises [2], value constellations [46], extended enterprises and collaborative highly integrated supply chains [18] are some forms of CNOs. Our interest is in IT-enabled CNOs, i.e., collaborations that are made possible by IT where the participants interoperate with each other by means of information systems. We believe that IT makes global competition and collaboration possible, forcing organizations to focus on what they can do well and facilitating collaboration between organizations with complementary competencies.

CNOs continue spreading since hypercompetitive environments [6] exist. This kind of environments forces organizations to re-think the way they are doing business by connecting and aligning the business and IT operations among them to meet goals. Participants in a CNO can be seen as distinct loosely coupled stakeholders with commonly conflicting interests and goals [16]. However, if they want to collaborate, they need to formulate a clear-enough common goal(s) toward which they strive together. This goal is not necessarily the goal of all participants. The common goal is an agreement between the customer-facing organization and its direct partners. This goal might include also other participating organizations in the CNO, but not necessarily.

CNOs are dynamic, because their environments are characterized by rapid changes in IT, easy competitors' market entry and uncertain market demands. Having well-defined collaborative work structures as basis [36], participants need to react promptly to customer needs. They will collaborate while an interesting 'business' opportunity exists. When this opportunity is over, the CNO dissolves while, perhaps, the organizations are active in other CNOs or look for new complementarities that allow them to participate in new 'business' opportunities.

## 2.3 Maturity models

MMs describe the evolution of a specific entity over time. Commonly, this entity is an organizational area or function. MMs have been developed to assess specific areas against a norm. Based on maturity assessments, organizations know the extent to which activities in such areas are predictable. That is, organizations can be aware of whether a specific area is sufficiently refined and documented so that the activities in this area now have the potential to achieve its desired outcomes. MMs apply a life-cycle approach where an area develops over time until it reaches its highest maturity level.

The first well-known maturity model was the software capability maturity model[1] (SW CMM) proposed by Carnegie Mellon University's Software Engineering Institute (SEI). This model identifies, specifically for software production, several levels of software process management sophistication.

[1] More information on http://www.sei.cmu.edu/cmm/

Essentially, MMs make it easier for organizations to establish goals for process improvement and identify opportunities for optimization, since these models describe basic attributes that are expected to characterize a particular area for each maturity level. By comparing an organization's characteristics and attributes with a MM, an organization will identify how mature it is in order to increase its process capability: first, establishing goals for the improvement of processes and then, taking action to achieve them.

# 3 Needs of MMs in CNOs

In the context of a CNO, proper understanding of the domains involved in B-ITa requires the integration of different models. There have been some proposals for assessing B-ITa. However, as these proposals are oriented to single organizations, they fail to take special characteristics of CNOs into account, such as the need for coordination, the lack of centralized decision making or the heterogeneity of IT architectures. Besides such proposals, there are also models that can be used to assess the maturity of one different aspect within CNOs. However, to the best of our knowledge, B-ITa is still not addressed by a single model in a CNO context that addresses all relevant aspects. In this section, we present a summary of different B-ITa MMs and MMs for CNOs that can be found in the literature. We did a semi-systematic literature review using a systematic literature review process [7] to select the related work presented in this section. The performed literature review consisted of a broad search of academic and practitioners' information sources. We approached the literature search using several electronic indexing services (e.g., ACM Digital Library, Google Scholar, Citeseer library, and IEEExplore). A set of key words was used: *alignment, business IT alignment, strategic alignment, IT alignment, architecture alignment, maturity model, assessment tool, measurement guide, networked organization, business network, collaborative enterprises*. We also used some alternative terms for alignment: *balance, harmony, fit*, and *linkage*. We traced the references in the identified papers to get access to other relevant sources. We reviewed the abstracts and the conclusions of the identified documents in order to determine their relevance to our research.

## 3.1 B-ITa Maturity Models

### 3.1.1 Luftman's MM

Luftman's strategic alignment model is an approach to determine a single organization's B-ITa based on six domains, namely skills, technology scope, partnership, governance, competency measurements, as well as communications [30]. Each of these domains is assigned five levels of alignment. The level of alignment for each individual domain is determined by means of answers to some questions. Luftman's model has been developed based on his practical experience and research, so this model is a pragmatic model. However, it disregards interrelations among the domains that explain B-ITa and it is focused to single organizations.

### 3.1.2 CIO Council's assessment guide

The Chief Information Officer (CIO) Council, a consortium of US Federal executive agency CIOs, developed an architecture-specific alignment and assessment guide [24]. This guide provides an overview of the integration of enterprise architecture concerns into the information technology investment planning process. It is useful to determine to what degree a proposed investment aligns with business strategies, and to know how well the technology of investments aligns with the infrastructure architecture. This assessment model is primarily focused on investment studies in federal agencies. It does not identify specific B-ITa domains, and thus it does not provide support to identify opportunities of improvement in organizations on some particular areas.

### 3.1.3 Duffy's MM

The MM developed by Duffy [22] consist of six domains required to understand B-ITa. The model is based on the premise that a reliable partnership between IT and non-IT executives is fundamental for achieving a successful B-ITa. Duffy recognizes that IT and business objectives are interdependent, and therefore, a division of practices into IT and non-IT areas would generally be unfavorable. Although the six domains reflect this position of the author, the model does not have explicit maturity levels for each of the domains. Instead, Duffy combines the six domains figuring out four B-ITa scenarios where organizations can be categorized. Such scenarios are the maturity levels in the model. Duffy's MM also is only focused on single organizations.

### 3.1.4 de Koning et al.'s model

Based on the analysis of business-IT excellence in several successful organizations in The Netherlands, and with the help of five hundred managers, de Koning and van der Marck [19] came up with ten questions that can be used to identify the level of alignment in single organizations. Those questions can be answered in a scale from 1 to 10 where the highest score means 'it entirely applies to my organization' and the lowest score means 'it entirely does not apply to my organization'. Although they do not identify B-ITa domains, this simple tool covers several B-ITa-related topics. However, the levels they present are limited to three:

immature, puberty and mature; this restricts the results and it neglects the assessment of the processes that do actually help to achieve B-ITa.

#### 3.1.5 van der Raadt et al.'s MAAM

The Multi-dimensional Assessment model for architecture Alignment and Maturity (MAAM) [49] can be used to assess architecture within organizations. The MAAM helps to identify the current situation of an organization's architecture, and to define improvement points and plans. The authors claim that a correlation exists between architecture alignment and architecture maturity. They claim that when architecture maturity increases, architecture alignment generally increases too (and v.v.). The MAAM consists of six interrelated domains that explain the alignment and maturity of an architecture. However, the model only assess such an architecture considering B-ITa as a stage that can be reached by increasing architecture maturity.

#### 3.1.6 COBIT

COBIT, issued by the IT Governance Institute[2], is a guide to employ management best practices and to measure IT processes. Version 4.0 of this guide provides a clear support to assess the align of IT with the business processes. For example, under the 'Defining a strategic IT plan' process, COBIT outlines how to engage IT managers on alignment with business goals and to develop a proactive process to quantify business requirements. However, (i) the focus of COBIT lies on IT Governance, (ii) it does not address a networked organization perspective, and (iii) COBIT deals with B-ITa from a strategic perspective.

#### 3.1.7 Laagland et al.'s assessment tool

According to Laagland et al. [50], the degree of alignment is mostly stipulated by the degree of maturity of changes on architecture. Managers must then look at the maturity of their organizations' architecture as start point for identifying improvement measures for B-ITa. This assessment tool enables to get aware where an organization stands, what its competencies are, and which measures must be implemented to reach a higher level of maturity. The tool describes the roles of business/IT managers, architects, project managers, and the like, for each of the architecture levels. With the model, it is possible to identify how organizations handle managing architecture and B-ITa.

### 3.2 Maturity Models for CNOs

Since we are developing a MM for CNOs, our literature review also covers some MMs that can be applied in collaborative settings (e.g., EIMM [29], IT outsourcing MM [1], extended CMM [41], E2AMM [44], SCM MM [32], and CollabMM [31]). Each of these models covers a particular domain related to B-ITa in CNOs (e.g. architecture or processes), disregarding other necessary domains that need to be taken in consideration by CNOs when achieving B-ITa. So, none of those models explicitly helps to assess B-ITa. It can be argued that those models could complement each other to assess B-ITa in CNOs. However, CNOs should spend considerable time and effort to understand and apply each of the models, and to analyze how the results could combine to plan B-ITa improvement actions. Therefore, to have a selection of domains in a single integrated model is useful for CNOs.

As no current B-ITa MM addresses alignment in CNOs and no MM for CNOs addresses more than a single aspect of B-ITa, we are filling this gap in CNOs by developing a new MM: the so called ICoNOs MM (**I**T-enabled **Co**llaborative **N**etworked **O**rganization**s** **M**aturity **M**odel) to assess B-ITa in CNO settings. We present this MM in the next section.

## 4 The ICoNOs MM

Developing MMs systematically is not a topic that is widely covered in the literature. Instead, most of the MM literature presents the resulting models only and does not discuss the model developing process itself. The development of the ICoNOs MM consists of several steps (see Fig. 2). Detailed explanation of most of these steps can be found in our earlier work [38]. Below we present a summary only. We make the note that because Fig. 2 is a high level view of our MM development process, we excluded any discussion on feedback loops needed to keep the MM updated in a dynamic environment. We, however, acknowledge the importance of monitoring the MM and managing the evolution of CNOs when the MM is modified.

The first step in developing a MM is to determine the **SCOPE** of the model, which means to set the boundaries for the model's application and use, and to define the purpose of the model. This is to differentiate the model from existing MMs. The second step is to **DESIGN** the model and covers:

- the specification of the model's type. MMs can be classified as *assessment* MMs and *development* MMs. The first type consists of normative models which serve as assessment tools that target certification, and help improve the organization's image as a reliable partner (e.g., the SEI series of CMMI-compliant models). The second type includes development tools that organizations use as guides for implementing best practices that, ultimately, lead to improvements and better results.

[2]More information on http://www.isaca.org/

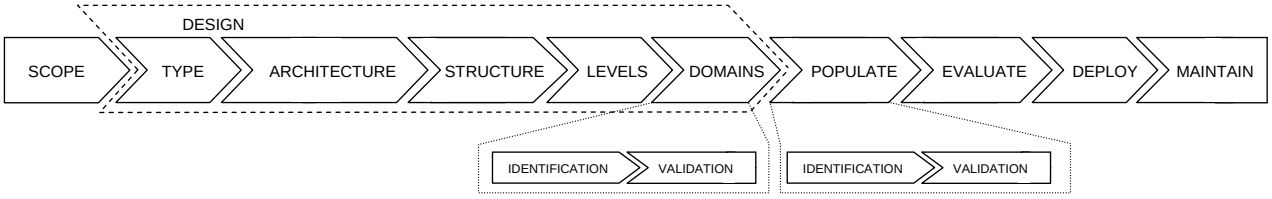



**Figure 2. MM development process.**

- the determination of the model's architecture. The architecture of a MM prescribes the manner in which maturity levels can be reached. For example, in a *staged* MM, before reaching level 3, an organization needs to achieve successfully what is mentioned in level 2 for all the domains included in the MM. In a *continuous* MM, each domain can be approached separately. This architecture allows selection of the order of improvements that best meets the objectives of organizations.

- the organization of its structure. The structure of a MM presents the organization of the model's components. It defines if the MM does include domains and key areas, and how they are decomposed and used to reach maturity levels.

- the definition of the maturity levels. This step implies to define the number of discrete levels of maturity for the model (typically five or six), and their qualifiers and definitions.

- the identification of the domains. The last step related to the design of the model is the identification of the domains to which the levels apply. A domain is a relevant aspect within the scope of the MM. For example, CMMI for software development [14] recognizes 4 domains: process management, project management, engineering and support. This task is not simple because after identifying the domains, they need to be validated to assure that they correspond to the purpose of the MM.

Once the design of the model is completed, process areas need to be identified for each domain so that we **POPULATE** the model with observable domain assessment criteria. A process area is a group of practices in a domain which, when implemented collectively, satisfy goals considered important for making an improvement in that domain (e.g., a process area in the IS architecture domain is 'IS portfolio management'). After populating the model, it must be validated in order to **EVALUATE** its applicability and generalizability. The objective is to validate the entire MM to test it for relevance and rigor. Following population and evaluation, the MM must be made available for use to verify its generalizability (step **DEPLOY**). To provide its acceptance and to improve its standardization, the MM must be applied in organizations that differ from the organizations that were involved in its design and population. The identification of organizations that may use the model and the application of the model to multiple organizations are the final steps towards its spreading and acceptance. Finally, it is important to track the evolution of (i) the organizational area or function that is assessed using the MM, and (ii) the requirements of the organizations that apply the model, in order to **MAINTAIN** the MM over time to keep it up-to-date. For example, the first MM developed by the SEI was the SW CMM. However, they observed that organizations would like to focus their improvement efforts not only in software engineering but also across different organizational functions. Therefore, the SEI came up with an integrated MM (CMMI) combining models from different disciplines to support the enterprise-wide process improvement that organizations were pursuing.

This paper focuses on the step POPULATE of the MM development process. Therefore, in the remainder of this section we report on the B-ITa process areas included in our model. First, we present the STRUCTURE , LEVELS and DOMAINS for a better understanding of the ICoNOs MM.

## 4.1 Structure of the model

The structure of the ICoNOs MM is based on CMMI for development [14]. It means that our MM builds on prior work. For example, some CMMI design choices are also present in ICoNOs. This situation avoids starting with the development of the model from scratch, and, most important, it also prevents our future users from starting over when adopting our MM for their B-ITa assessments.

The ICoNOs MM has four layers of aggregation. The upper layer consists of the domains that must be addressed in a CNO when achieving B-ITa, i.e., partnering structure, IS architecture, process architecture and coordination (see Sect. 4.3). The next three layers reflect the overall CMMI structure (see [14, p.30]). For example, in each of the domains we can also find process areas. Process areas are sets of activities that are performed to make improvements in a particular domain (see Sect. 4.4). Similarly to CMMI, the ICoNOs MM process areas have specific and generic

goals, which the activities in the process area are supposed to achieve. The specific goals describe characteristics that must be present to satisfy a particular process area; they are specific for this area. There are also goals, called generic goals, that apply to all process areas, although their instantiation for each process area can differ. For example, a CMMI generic goal is 'the process is institutionalized as a defined process'. Our MM will incorporate the generic goals of CMMI[3]. The goals will be decomposed in specific and generic practices describing what a CNO may implement to achieve the specific and generic goals. These practices will be expected and are not mandatory. This means that it will be permitted to implement alternative practices in substitution for the specific and generic practices that the ICoNOs MM will include. The only condition is that the goals must be satisfied, to perform a process, to reach a specific maturity level.

### 4.2 The B-ITa levels

The ICoNOs MM has five levels of maturity (see Fig. 3). Levels are used to describe an improvement path recommended for a CNO that wants to improve processes to achieve B-ITa. To reach a particular level, a CNO must satisfy all the set of process areas that are targeted for improvement in a particular B-ITa domain. The levels are:

**Level 1: Incomplete.** At maturity level 1, processes related to a particular B-ITa domain are usually not performed or partially performed. It means such a particular domain is not explicitly considered when a CNO strives for B-ITa.

**Level 2: Isolated.** At maturity level 2, processes are the basic infrastructure in place to support a particular B-ITa domain. They (i) are planned and executed in accordance with a policy; (ii) employ skilled people who have adequate resources to produce controlled outputs; (iii) are monitored, controlled, and reviewed. However, such processes are isolated initiatives that are not managed from the entire CNO perspective.

**Level 3: Standardized.** At maturity level 3, processes are directed to make improvements in the standardization and management of a particular B-ITa domain. Processes are performed from a CNO perspective (i.e., they are cooperation initiatives). They are well characterized and understood, and are described in standards, procedures, tools, and methods.

**Level 4: Quantitatively Managed.** At maturity level 4, processes use statistical and other quantitative techniques. Quantitative objectives for quality and process performance are established and used as criteria in managing the process.

[3] A detailed list of these generic goals can be found in [14]

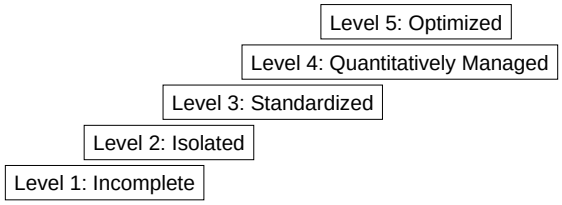



**Figure 3. The B-ITa levels.**

Quality and process performance is understood in statistical terms and is managed throughout the life of the process.

**Level 5: Optimized.** At maturity level 5, processes are improved based on an understanding of the common causes of variations inherent in the process. The focus of an optimized process is on continuously optimizing the range of process performance through both incremental and innovative improvements.

### 4.3 The B-ITa domains

Once the levels are defined, domains where these levels must apply need to be identified. A domain is a group of processes that help to have improvements in a particular CNO area. In previous work [37, 39, 40], we have reported on how we have used a focus group and case studies to identify the domains that need to be addressed by CNOs in their efforts for aligning business and IT. Fig. 4 presents the fit among the B-ITa domains. In the following, we give a short summary of these domains.

- Partnering structure, defined as the CNO work division, organizational structure, and roles and responsibilities definition that indicate where the work gets done and who is involved.
- IS architecture, defined as the fundamental organization of the information management function of the participating organizations embodied in the information systems, i.e., software applications, that realize this function, their relationships to each other and to the environment, and the principles guiding their design and evolution.

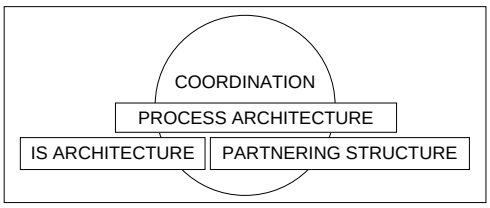



**Figure 4. The B-ITa domains.**

- Process architecture, defined as the choreography of all (individual and collaborative) processes needed to reach the shared goals of the participating organizations.
- Coordination, defined as the mechanisms to manage the interaction and work among the participating organizations taking into account the dependencies and the shared resources among the processes.

## 4.4 The B-ITa process areas

Several theories and models, developed elsewhere, are potentially useful to give insights for understanding the processes related to B-ITa in CNOs (e.g., [4, 5, 12, 14, 17, 20, 21, 22, 25, 26, 27, 30, 33, 34, 43, 47, 48, 49]). Our position is that it would be practical for CNOs to have a selection of those processes in a single model. Fig. 5 establishes a map relating several theories and models to the four B-ITa domains introduced in the previous subsection. It must be noted that each theory and model covers much more than the constructs (i.e., processes and process outcomes) we present in the figure. That is, in our research, we take from each theory/model those constructs only, which could have a relation to the four B-ITa domains. Clearly, it can be argued that we do not include all theory/model constructs with a possible relation with the B-ITa domains. However, after an exhaustive analysis of the theories and models, we decided to include only general constructs. For example, the 'Requirements development' process of CMMI covers specific characteristics that are considered, in a general way, by the 'Requirement management' process which we do take into account in our mappings (see Fig. 5). In this figure the acronyms **PS**, **IS**, **PA** and **CO** stand for partnering structure, IS architecture, process architecture, and coordination, respectively.

The leftmost and the rightmost columns in this figure present the theories or models taken from the literature. By 'model' we mean a conceptual model, i.e., a set of constructs used to describe B-ITa or a domain of B-ITa; by 'theory' we mean a model plus claims about empirical relations between some concepts, i.e., correlational or causal relationships. From each theory/model we have selected constructs, assigned these to a B-ITa process area and assigned the B-ITa process area to a B-ITa domain. For example, Gunderson's theory of system safety analysis (depicted in the upper left-hand corner of Fig. 5) is mapped onto the RAM B-ITa process area of the ICoNOs MM, where RAM stands for Risk Analysis and Mitigation. The arrow connecting 'system safety analysis' to the oval labeled **IS** indicates that this is associated to the IS architecture domain. We make the note that the definitions of some constructs made us decide to map them to more than one B-ITa process area. For example, Hoque's theory of portfolio management can be applied to process architecture (PPM B-ITa process area) and to IS architecture (IsPM B-ITa process area). The assignment of process areas to domains is summarized in Fig. 6. It can be argued that the positioning of the processes into a specific B-ITa level seems arbitrary. However, the decisions for such a positioning were driven by the definition of each process and by what we have seen in the three case studies we conducted to validate the design of ICoNOs. Recently, we began to conduct a new case study to identify whether the process areas of the ICoNOs MM are present in a real-life CNO, and to validate their positioning into the model. It is too early to make a conclusion but from the evidence obtained heretofore in the case study site, we can anticipate that the SPD B-ITa process area could fit better in level 3 of the IS architecture domain than in level 2.

### 4.4.1 Partnering structure process areas

We present the B-ITa process areas (in alphabetical order), grouped into the four B-ITa domains. For each process area, we provide (in parentheses) the level in which the process area is positioned, and the reference of the theory(ies) and/or model(s) from we derived it.

Our model includes 7 process areas in the partnering structure domain. These process areas are:

**BMD** *Business model definition* (L2). To define a blueprint of how the CNO works, describing how different variables of the collaboration fit together as a system to help creating value for each participant [27].

**GSC** *Governance structure and compliance* (L3). To structure the priorities and allocation of resources and decision rights to create accountability; and to ensure that activities are performed in conformity with policies and procedures. A successful compliance process will be performed through definition of effective policies and procedures [22, 33, 47, 48, 49].

**IoPD** *Inter-organizational policies definition* (L3). To define the plans of action, intended to influence and determine decisions, including shared risk and rewards policies to increase mutual benefits perception and shared commitment [30].

**MRE** *Metric-based exploration of roles* (L4). To employ approaches as relational exchange techniques, organizational communication's mechanistic and system-interaction methods to study organizational communication, structure and roles in the collaboration [26].

**OSD** *Organizational structure definition* (L2). To define the inter-organizational ties constructing a framework for inter-organizational decision making and placing power and authority in order to regulate the CNO work [30].

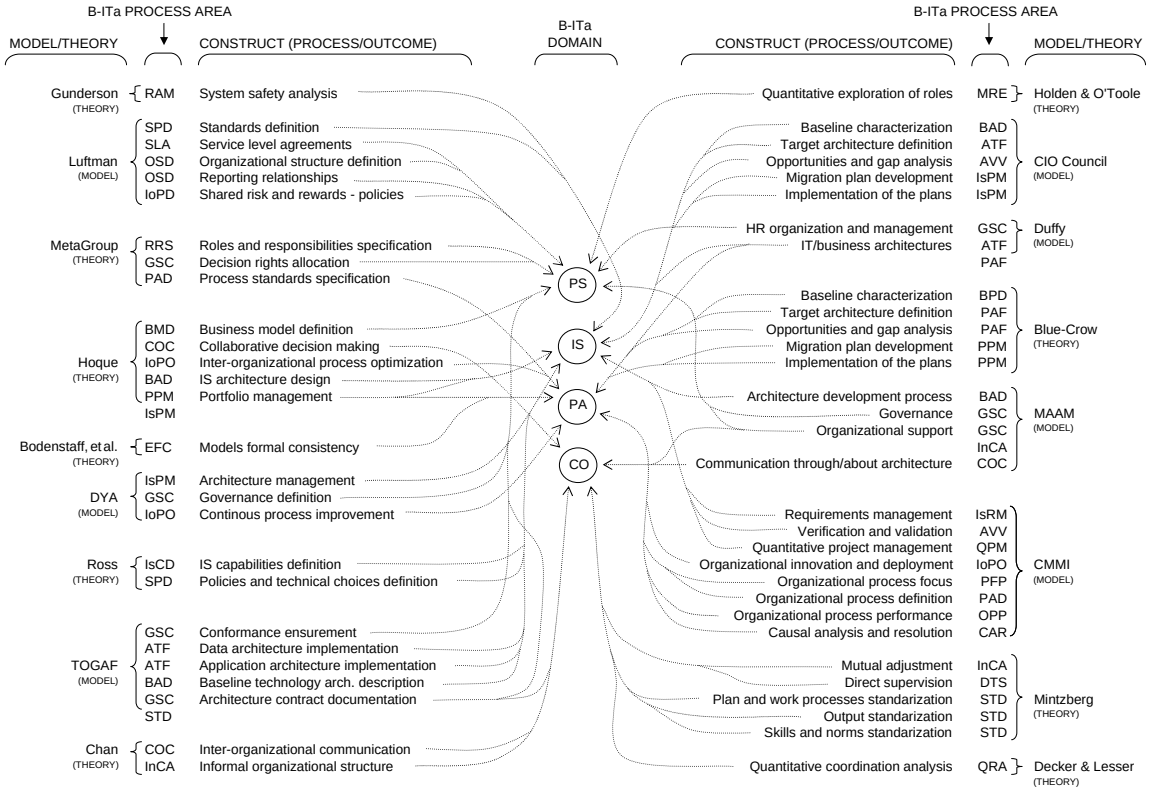



**Figure 5. Map modeling theories applicable to the B-ITa domains.**

**RRS** *Roles and responsibilities specification* (L3). To specify the roles and responsibilities, and their related guide principles, of the participants in the CNO after define its organizational structure [33].

**SLA** *Service level agreements definition* (L2). To describe the agreements on the deliverables, quality, and fitness-for-purpose of services that have an impact on the work of each participating organization. A successful implementation of these agreements will be delivered through effective governance structure [30].

### 4.4.2 IS architecture process areas

The ICoNOs MM covers 9 process areas into this domain. These process areas are:

**ATF** *IS architecture target formulation* (L3). To evaluate, select and design ISs needed to support the desired to-be state of the IS architecture taking into account business and IT drivers, and the processes to support [21, 22, 47].

**AVV** *IS architecture verification & validation* (L3). To perform periodically gap analysis to make sure changing IS requirements are managed in consistent fashion with IS architecture targets. A successful verification & validation will be performed through an effective IS target formulation [14, 21].

**BAD** *Baseline IS architecture description* (L2). To create a snapshot of the existing ISs and data, assessing what the current status of the CNO is concerning ISs [21, 27, 47, 49].

**IsCD** *IS capabilities definition* (L3). To define the ability of the collaboration to achieve new forms of competitive advantage by ISs to achieve congruence with the business environment where it works [43].

**IsPM** *IS portfolio management* (L3). To create the right mix of information systems investments to properly use limited resources while providing the maximum business benefit. A successful IS portfolio management will be delivered through the execution of the other IS processes effectively [21, 27, 48].

**IsRM** *IS requirements management* (L2). To manage the changing IS requirements during their engineering process and the development of the required ISs [14].

**QPM** *Quantitative IS portfolio management* (L4). To use quantitative techniques to analyze, assess, and control

| | PARTNERING STRUCTURE | | IS ARCHITECTURE | |
|---|---|---|---|---|
| 5 | | | Risk analysis and mitigation | RAM |
| 4 | Metric-based roles exploration | MRE | Quantitative IS portfolio management | QPM |
| 3 | Governance structure and compliance<br>Roles and responsibilities specification<br>Inter-organizational policies definition | GSC<br>RRS<br>IoPD | IS architecture target formulation<br>IS capabilities definition<br>IS architecture verification and validation<br>IS portfolio management | ATF<br>IsCD<br>AVV<br>IsPM |
| 2 | Business model definition<br>Service level agreements definition<br>Organizational structure definition | BMD<br>SLA<br>OSD | Baseline IS architecture description<br>Standards and principles definition<br>IS requirements management. | BAD<br>SPD<br>IsRM |
| 1 | | | | |

| | PROCESS ARCHITECTURE | | COORDINATION | |
|---|---|---|---|---|
| 5 | Inter-organizational process optimization<br>Causal analysis and resolution | IoPO<br>CAR | | |
| 4 | Organizational process performance<br>Event logs formal consistency | OPP<br>EFC | Quantitative coordination relation analysis | QRA |
| 3 | Organizational process focus planning<br>Process architecture target formulation<br>Process architecture definition<br>Process portfolio management | PFP<br>PAF<br>PAD<br>PPM | Standardization<br>Communication-oriented coordination | STD<br>COC |
| 2 | Baseline process architecture description | BPD | Informal communication adjustment<br>Direct supervision | InCA<br>DTS |
| 1 | | | | |

**Figure 6. The ICoNOs MM.**

IS portfolio assets, managing such a portfolio from a quantitative perspective [14].

**RAM** *Risk analysis and mitigation* (L5). To identify sources of flaws and other problems (e.g., requirements inconsistencies, poor portfolio management, lack of IS principles) in the IS architecture domain, and to take action to prevent such situations in the future [25].

**SPD** *Standards and principles definition* (L2). To define technology standards, policies and development principles stating direction or practice on how the collaboration should deal with the ISs [30, 43].

#### 4.4.3 Process architecture process areas

Our model includes 9 process areas which refer to this domain. These process areas are:

**BPD** *Baseline process architecture description* (L2). To create a snapshot of the existing processes, identifying and analyzing what the current status of the CNO is concerning processes [4].

**CAR** *Causal analysis and resolution* (L5). To identify sources of flaws and other problems in the process architecture domain, and to take action to prevent such situations in the future [14].

**EFC** *Event logs formal consistency* (L4). To use event logs for checking traceability of execution processes during collaboration, and for controlling whether profitability estimates are realized [5].

**IoPO** *Inter-organizational process optimization* (L5). To evaluate the process architecture in order to deploy incremental and innovative improvements to gain inter-organizational efficiency and competitive advantage. A successful process optimization relies on effective process focus planning and process architecture definition [14, 27, 48].

**OPP** *Organizational process performance* (L4). To establish and maintain a quantitative understanding of the performance of the standard processes set in support of quality and process-performance objectives [4, 14, 21].

**PAD** *Process architecture definition* (L3). To establish and maintain a repository of CNO processes, assets and work environment standards. A successful process architecture definition depends on an effective baseline process architecture description [14, 33].

**PAF** *Process architecture target formulation* (L3). To evaluate, select and design processes needed to support the desired to-be state of the process architecture taking into account business and strategy drivers [4, 22].

**PFP** *Organizational process focus planning* (L3). To plan, implement, and deploy process improvements based on a thorough understanding of strengths and weaknesses of the collaboration's processes and process assets. A successful process planning will be performed through effective process architecture definition [14].

**PPM** *Process portfolio management* (L3). To direct limited resources in terms of funds, people, etc., into the processes to create a holistic process orientation [4, 27].

#### 4.4.4 Coordination process areas

The process areas covered by the ICoNOs MM into this domain are:

**COC** *Communication-oriented coordination* (L3). To agree on communication channels, sharing knowledge and learning in order to respond effectively to immediate client's needs and to determine what future markets will require [12, 17, 27, 49].

**DTS** *Direct supervision* (L2). To supervise the work by specific persons who take the responsibility for the processes, providing instructions to others and monitoring their actions [34].

**InCA** *Informal communication adjustment* (L2). To adjust and control the work among the participating organizations by informal communication among the actors outside the imposed hierarchical constrains for day-to-day operations [12, 34, 49].

**QRA** *Quantitative coordination analysis* (L4). To use techniques (e.g., causal model analysis) to link the interrelationships, called coordination relations, to the local scheduling constraints of the participating organizations [20].

**STD** *Standardization* (L3). To coordinate work and interactions by standardizing the processes, outputs and/or skills among the participating organizations [34, 47].

# 5 Discussion

## 5.1 Practical adoption of the ICoNOs MM

A key design decision that impacts adoption in practice of the ICoNOs MM is the decision to assign separate maturity level to each participant in a CNO. In other words, each single participating organization within a CNO can have a different level of B-ITa maturity. Although ICoNOs is being developed to assess the alignment of the entire CNO, the decisions concerning achieving, or assessing, B-ITa in a CNO can be made by one participating organization. Thus, not all participants in a CNO have to adopt the ICoNOs MM, or do so at the same time. While this design decision facilitates adoption, it is not the case that B-ITa maturity levels of each participant in a CNO are completely independent. Instead, the maturity of one participating organization influences the maturity of the alignment between business and IT of the entire CNO. For example, a participant with a specific level of B-ITa maturity as single organization can impose other participants to collaboratively achieve the same maturity level as a networked organization.

We consider chief officers of the partnering organizations in a CNO as the key users of the ICoNOs maturity assessments. This assumption is motivated by published results of researchers (e.g. [8, 11, 23, 28]), which show that the most powerful initial step to achieve B-ITa is to build strong organizational support through strong commitment of CIOs and/or CEOs. If chief officers want to improve B-ITa, they need first to assess the processes related to B-ITa, and commit as B-ITa catalysts and sponsors. Applying these findings to our work, chief officers must be actively involved in the CNO B-ITa project in at least three ways: (i) influencing the CNO to use the ICoNOs MM, (ii) choosing the best team to manage the B-ITa improvement effort, and (iii) monitoring the assessment and improvement process in each B-ITa domain. As the ICoNOs MM is a continuous MM [38], it lets chief officers assess each B-ITa domain separately (see Fig. 7). This feature of the model will let CNOs focus, for instance, on the domains with a low level of maturity. Those domains that are associated with higher maturity can, then, be candidates for inclusion in later improvements efforts.

## 5.2 Preliminary Evaluation

At this stage of our work, we do not have empirical evidence for the correlation between the ICoNOs MM's domains and process areas, and the business-IT alignment success in CNOs. Such a validation of the model is only possible after evaluating its design and population. However, we did an early evaluation of the question whether our approach is capable of dealing with any type of CNOs and for which type of CNOs its results will bring most benefits (i.e., the results would be most insightful). When conducting case studies to validate the design of the MM, we made sure to study different case study sites. We chose CNOs from different countries, one international and two of national nature, one entrepreneur-led and two government agencies, and one with a large amount of participants and two with only 2 or 3 participating organizations. We must also note that the B-ITa key drivers they have are different. The key drivers of one of the studied CNOs are to control costs and to manage risk, while the B-ITa key drivers of the other sites are to improve quality and to increase effectiveness.

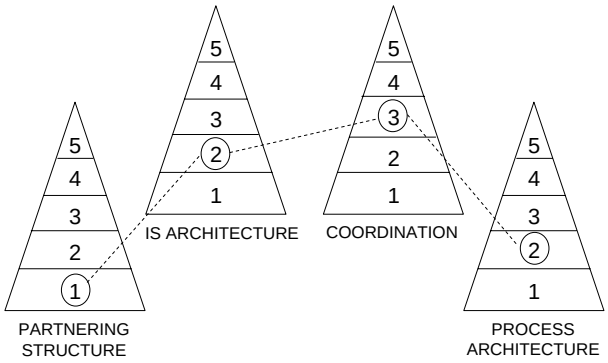



**Figure 7. The pyramid view of the model.**

So far, we claim that the ICoNOs assessment results are useful for CNOs that meet the CNO characteristics reflected in our definition of CNO (see Sect. 2). That is, collaborations where (i) participants pool costs, skills, and core competences to provide world-class solutions that could not be provided by any of them individually; (ii) information systems are able in each of the participants to respond dynamically to meet the ever-changing customer needs and to communicate and share information among them; (iii) participants have a clear understanding of the common goal(s) and the functions of each of the participating organizations in order to know what is expected from each of them.

## 5.3 Open Issues

Some interesting open matters remain to be addressed to produce a complete MM. First, we acknowledge that despite the fact that we associated each process area to one B-ITa domain only, these B-ITa process areas have an effect on each other regardless of the domain in which we included them. For example, the process area of 'Process architecture target formulation' is an input to the process

area of 'IS architecture target formulation' when addressing a design of the information systems required to support the CNO processes. We note that 'IS architecture target formulation' is a process area in the IS architecture domain and 'Process architecture target formulation' is a process area in the process architecture domain. To identify the possible relationships among the different process areas is part of the work required to provide a complete MM. We also want to provide a clear picture of the relations among the B-ITa domains, as the MAAM [49] does.

Second, to have a complete MM, the ICoNOs MM must incorporate specific goals and practices (see Sect. 4.1) describing characteristics that must be present to satisfy the B-ITa process areas. These specific goals and practices could be seen as the results to be achieved and the activities to be performed in each of the process areas included in the model. Third, validating a MM by means of a comparison with another model is considered a difficult task, as there is no reference model in practice. Therefore, for the EVALUATE step of the MM development process presented in Fig. 2, we plan to use expert panels [3], focus groups [15, 45] and testing pilots (where sponsorship from CNOs would be necessary in order to use a prototype of the model to appraise the maturity of their B-ITa).

## 6 Conclusion and Future Work

The goal of this paper is to present (i) a process model that can be used as a guide for developing maturity models, and (ii) the first version of a maturity model to assess processes related to business-IT alignment in collaborative networked organizations (CNOs): the ICoNOs MM. Based on an analysis of the potential applicability of several theories and models in the area of business-IT alignment, we present process areas grouped in four domains: partnering structure, IS architecture, process architecture and coordination (see Sect. 4.3). These domains should be addressed by networked organizations in their efforts to achieve business-IT alignment. Unlike maturity models for assessing alignment in single organizations, the ICoNOs MM is applicable at the CNO level. This maturity model is a promising attempt to properly understand the domains involved in collaborative business-IT alignment in terms of process maturity.

We stress that the ICoNOs MM is a work in progress to be further developed, revised, and eventually modified. Details remain to be worked out in the future as more knowledge becomes available from a case study we are conducting in a CNO to empirically identify whether the process areas included in ICoNOs are present in the investigated case study site. Our work for the immediate future includes identifying the specific goals and practices for each of the process areas (see Sect. 4.1). Future work also includes validating the maturity model as a whole. We plan to use testing pilots, expert panels and focus groups to address this validation.

## References


[1] O. Adelakun. IT outsourcing maturity model. In *Proceedings of the 13th European Conference on Information Systems, The European IS Profession in the Global Networking Environment, ECIS'04*, 2004.

[2] F. M. Barbini and A. D'Atri. How innovative are virtual enterprises? In *ECIS*, 2005.

[3] S. Beecham, T. Hall, C. Britton, M. Cottee, and A. Rainer. Using an expert panel to validate a requirements process improvement model. *Journal of Systems and Software*, 76(3):251–275, 2005.

[4] Blue-Crow. Business process modelling and analysis, 2007.

[5] L. Bodenstaff, A. Wombacher, and M. U. Reichert. On formal consistency between value and coordination models. Technical Report TR-CTIT-07-91, Enschede, October 2007.

[6] W. C. Bogner and P. S. Barr. Making sense in hypercompetitive environments: A cognitive explanation for the persistence of high velocity competition. *Organization Science*, 11(2):212–226, 2000.

[7] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil. Lessons from applying the systematic literature review process within the software engineering domain. *J. Syst. Softw.*, 80(4):571–583, 2007.

[8] M. Broadbent and E. Kitzis. Interweaving business-driven IT strategy and execution: Four foundation factors. *Ivey Business Journal*, 69(3):1–6, 2005.

[9] L. M. Camarinha-Matos and H. Afsarmanesh. *Collaborative Networked Organizations: A Research Agenda for Emerging Business Models*. Kluwer Academic Publishers, 2004.

[10] L. M. Camarinha-Matos and H. Afsarmanesh. The emerging discipline of collaborative networks. In *Virtual Enterprises and Collaborative Networks*, pages 3–16. Kluwer Academic Publishers, 2004.

[11] B. Campbell. Alignment: Resolving ambiguity within bounded choices. In *Proceedings of the PACIS 2005*, pages 1–14, Bangkok, Thailand, 2005.

[12] Y. E. Chan. Why haven't we mastered alignment? the importance of the informal organization structure. *MIS Quarterly Executive*, 1(21):76–112, 2002.

[13] Y. E. Chan, S. L. Huff, D. W. Barclay, and D. G. Copeland. Business strategic orientation, information systems strategic orientation, & strategic alignment. *Information Systems Research*, 8:125–150, 1997.

[14] CMMI Product Team. CMMI for Development, Version 1.2: Improving processes for better products., 2006.

[15] D. R. Cooper and P. S. Schindler. *Business Research Methods*. Boston, [Mass., etc.]: McGraw-Hill, 8th edition, 2003.

[16] D. Damian. Stakeholders in global requirements engineering: Lessons learned from practice. *IEEE Software*, 24(2):21–27, 2007.

[17] M. Daneva and R. Wieringa. A coordination complexity model to support requirements engineering for cross-organizational ERP. In *RE'06: Proc. of the 14th IEEE Int. Requirements Engineering Conference*, Minneapolis, MN, USA, 2006.

[18] E. W. Davis and R. E. Spekman. *The Extended Enterprise: Gaining Competitive Advantage through Collaborative Supply Chains*. Financial Times Prentice Hall, 2003.

[19] D. de Koning and P. van der Marck. *IT Zonder Hoofdpijn: Een Leidraad voor het Verbeteren van de Bedrijfsprestaties*. Prentice Hall, 2002. In Dutch.

[20] K. Decker and V. Lesser. Analyzing a quantitative coordination relationship. *Group Decision and Negotiation*, 2(3):195–217, 1993.

[21] DOC Enterprise IT Architecture Advisory Group. Information technology architecture: What is it, why should you care, and how do you do one?, 2004.

[22] J. Duffy. Maturity models: Blueprints for e-volution. *Strategic and Leadership*, 29(6):19–26, 2001.

[23] B. A. Edwards. Chief executive officer behaviour: The catalyst for strategic alignment. *Journal of Value-Based Management*, 13(1):47–54, 2000.

[24] Federal Architecture Working Group. *Architecture Alignment and Assessment Guide*. The Federal Chief Information Officer Council, 2000.

[25] S. Gunderson. A review of organizational factors and maturity measures for system safety analysis. *Syst. Eng.*, 8(3):234–244, 2005.

[26] M. T. Holden and T. O'Toole. A quantitative exploration of communication's role in determining the governance of manufacturerretailer relationships. *Industrial Marketing Management*, 33(6):539–548, 2004.

[27] F. Hoque. *The Alignment Effect*. FT Press, 2002.

[28] G. S. Kearns and A. L. Lederer. A resource-based view of strategic IT alignment: How knowledge sharing creates competitive advantage. *Decision Sciences*, 34(1):1–29, 2003.

[29] T. Knothe, K. Schneider, D. Böl, T. Kahl, S. Schuster, F. Lillehagen, J. Krogstie, and H. G. Solheim. Framework for the establishment and management methodology, 2005. Deliverable A.1.4.1, ATHENA, Integrated Project Contract Num. IST-507849.

[30] J. Luftman. Assessing IT-business alignment. *Information Systems Management*, 20:9–15, 2003.

[31] A. Magalhaes, R. Araújo, and M. R. S. Borges. Designing collaborative processes. In *Proceedings of the 8th Workshop on Business Process Modeling, Development and Support (BPMDS'07) in the 19th International Conference on Advance Information Systems Engineering (CAISE'07)*, 2007.

[32] K. McCormack and A. Lockamy. The developmentof a supply chain management process maturity model using the concepts of business process orientation. *Supply Chain Management*, 9(4):272–278, 2004.

[33] META Group. Architecture maturity audit: Part 1. *META Practice*, 4(4), 2000.

[34] H. Mintzberg. *Structure in Fives: Designing Effective Organizations*. Prentice Hall, Englewood Cliffs, NJ., second edition, 1993.

[35] R. Santana Tapia. What is a networked business? Technical Report TR-CTIT-06-23a, University of Twente, Enschede, The Netherlands, 2006.

[36] R. Santana Tapia and N. Zarvić. Value-based partnering structure design for networked businesses: A multi-method approach. In *Proceedings of 21st Bled Conference "eCollaboration"*, pages 263–276, Bled, Slovenia, 2008.

[37] R. Santana Tapia, M. Daneva and P. van Eck. Business-IT alignment domains and principles for networked organizations: A qualitative multiple case study. 3rd International Workshop on Enterprise Integration, Interoperability and Networking. EI2N'08. Submitted and under review.

[38] R. Santana Tapia, M. Daneva and P. van Eck. Developing an inter-enterprise alignment maturity model: Research challenges and solutions. In C. Rolland, O. Pastor, and J.-L. Cavarero, editors, *Proc. of the 1st Int. Conf. on Research Challenges on Information Science (RCIS'07)*, pages 51–59, Ouarzazate, Morocco, 2007.

[39] R. Santana Tapia, M. Daneva and P. van Eck. Validating adequacy and suitability of business-IT alignment criteria in an inter-enterprise maturity model. In *Proceedings of the Eleventh IEEE International EDOC Enterprise Computing Conference, Annapolis, MD, USA*, pages 202–213, Los Alamitos, October 2007. IEEE Computer Society Press.

[40] R. Santana Tapia, P. van Eck and M. Daneva. Inter-organizational business-IT alignment maturity: Validating the domains of an alignment assessment instrument. International Conference on Information Systems, ICIS'08. Submitted and under review.

[41] N. Ramasubbu, M. Krishnan, and P. Kompalli. A process maturity framework for managing distributed development. *IEEE Software*, 22(3):80–86, 2005.

[42] B. H. Reich and I. Benbasat. Development of measures to investigate the linkage between business and information technology objectives. *MIS Quarterly*, 20:55–81, 1996.

[43] J. Ross. Creating a strategic IT architecture competency: Learning in stages. *MISQ Executive*, 2(1):31–43, 2003.

[44] J. Schekkerman. *Extended Enterprise Architecture Maturity Model Support Guide v2.0*. Institute for Enterprise Architecture Developments, 2006.

[45] Y. Simmons. Using focus groups as an applied research tool. *Howe's Now*, 6(2), Apr 2000.

[46] D. Tapscott, D. Ticoll, and A. Lowy. *Digital Capital - Harnessing the Power of Business Webs*. Nicholas Brealy Publishing, 2000.

[47] The Open Group. *The Open Group Architecture Framework TOGAF –2007 edition (Incorporing 8.1.1)*. Van Haren Publishing, 2007.

[48] M. van den Berg and M. van Steenbergen. *DYA: Stap voor stap naar professionele enterprise-architectuur*. Ten Hagen & Stam uitgevers, 2004. In Dutch.

[49] B. van der Raadt, J. F. Hoorn, and H. van Vliet. Alignment and maturity are siblings in architecture assessment. In *CAISE '05: Proceedings of the 17th International Conference on Advanced Information Systems Engineering*, volume 3520/2005 of *LNCS*, pages 357–371, Porto, Portugal, 2005. Springer.

[50] H. van der Zee, P. Laagland, and B. Hafkenscheid. *Architectuur als Managementinstrument: Beheersing en Besturing van Complexiteit in het Netwerktijdperk*. Ten Hagen Stam Uitgevers, 2001. In Dutch.

[51] P. van Eck, H. M. Blanken, and R. Wieringa. Project GRAAL: Towards operational architecture alignment. *International Journal of Cooperative Information Systems*, 13:235–255, Sep 2004.