An Interoperability Classification Framework for Method Chunk Repositories

Per Backlund[1], Jolita Ralyté[2], Manfred A. Jeusfeld[3], Harald Kühn[4]
Nicolas Arni-Bloch[2], Jan B.M.Goossenaerts[5], and Frank Lillehagen[6]

[1]University of Skövde, P.O. Box 408, SE 541 28 Skövde, Sweden
Per.Backlund@his.se
[2] CUI, University of Geneva, Rue de Général Dufour, 24, CH-1211 Genève 4, Switzerland
{Jolita.Ralyte, Nicolas.Arni-Bloch}@cui.unige.ch
[3]Tilburg University, CRISM/Infolab, 5000 LE Tilburg, The Netherlands
Manfred.Jeusfeld@uvt.nl
[4]BOC Information Systems GmbH, Rabensteig 2, A-1010 Vienna, Austria
Harald.Kuehn@boc-eu.com
[5]Eindhoven University of Technology, Den Dolech 2, Paviljoen D 12, Postbus 513, 5600 MB Eindhoven, The Netherlands
J.B.M.Goossenaerts@tm.tue.nl
[6]TROUX Technologies AS, PO Box 482, N-1327 Lysaker, Norway
Frank.Lillehagen@Troux.com

Abstract. The competitiveness and efficiency of an enterprise is dependent on its ability to interact with other enterprises and organisations. In this context interoperability is defined as the ability of business processes as well as enterprise software and applications to interact. Interoperability remains a problem and there are numerous issues to be resolved in different situations. We propose method engineering as an approach to organise interoperability knowledge in a method chunk repository. In order to organise the knowledge repository we need an interoperability classification framework associated to it. In this paper we propose a generic architecture for a method chunk repository, elaborate on a classification framework and associate it to some existing bodies of knowledge. We also show how the proposed framework can be applied in a working example.

1 Introduction

Interoperability is defined as "the ability of Enterprise Software and Applications to interact" (Interop, 2005). We claim that it is impossible to provide one universal method for interoperability problems solution and we propose to define a knowledge base of reusable method chunks each of them addressing one or more specific interoperability problems. In order to support situation-specific method construction and application, a collaborative tool must be developed supporting method chunks construction and storage as well as their selection and reuse in different projects. The specialisation of such a knowledge management tool for the interoperability domain

requires the creation of a mapping from the method chunks to the interoperability problems, i.e. an indexation mechanism associating each method to one or several well-defined interoperability problems. The definition and classification of interoperability problems is necessary for interoperability situation assessment and selection of the method chunks satisfying this situation.

In this paper we view information systems development as knowledge work (Iivari, 2000; Backlund, 2004) with the aim of exploring an interoperability classification framework for a Method Chunk Repository which can be used to solve industry relevant interoperability problems. We propose method engineering as a means for dealing with some aspects of interoperability. However, in order to make an interoperability method chunk repository useful we must supply a classification of interoperability problems which can be used to guide the repository user in composing methods. The proposed repository should deal with interoperability problems within information systems development; hence we will anchor the classification scheme in the information systems body of knowledge (Iivari et al., 2004).

The focus cannot be placed on the applications alone. In order to achieve meaningful interoperability organisations must be interoperable on, at least three levels: a business layer, a knowledge layer and an ICT systems layer [4]. This includes the business environment and business processes on the business layer, the organisational roles, skills and competencies of employees and knowledge assets on the knowledge layer, and applications, data and communication components on the ICT layer. Similarly, but from a more software-architecture oriented view, Schulz et al. [12] conclude that interoperability is achieved on the following levels: inter-enterprise coordination, business process integration, semantic application integration, syntactical application integration, and physical integration. According to these authors interoperability should be analysed from an enterprise view (i.e. interoperability between two or more organisations), an architecture & platform view (i.e. between two or more applications/systems) and an ontological view (i.e. the semantics of interoperability).

As can be seen from the above descriptions, interoperability is a multifaceted concept. In order to be able to match a specific problem situation of a particular case to method chunks enabling the problem solution, we need a mechanism supporting method chunks indexation on the one hand and situation assessment on the other hand. This mechanism is referred to as a matching/classification framework. It must bring diverse bodies of knowledge together and extend them with interoperability concepts.

The remainder of this paper is organised as follows. In section 2 we introduce a collaborative Method Engineering platform based on a method chunk repository for interoperability. The building blocks of the Method Engineering platform and the roles involved in platform use are presented. In section 3 we briefly analyse interoperability problems in an enterprise context and present a framework to classify interoperability problems and method chunks. Section 4 illustrates how the classification framework is applied to an industrial case. The paper ends with a review of this work, and outlines future research.

2 Collaborative Method Engineering Platform for Interoperability

A collaborative platform for situational method engineering must support two main activities: situation-specific method construction and method application in the corresponding system development project. The method construction activity requires capabilities for reusable method chunks definition, storage and classification with respect to the problems they help to solve. It also aims to support the characterisation of each project situation and selection and assembly of method chunks fitting the situation at hand. The method application requires services for the obtained method enactment and evaluation of its applicability in the corresponding situation. The knowledge about positive or negative experience of method application is captured in terms of best practices and/or experience reports.

Fig. 1~~Fig. 1~~ illustrates the architecture of our platform divided into three layers: *usage*, *service* and *data*.

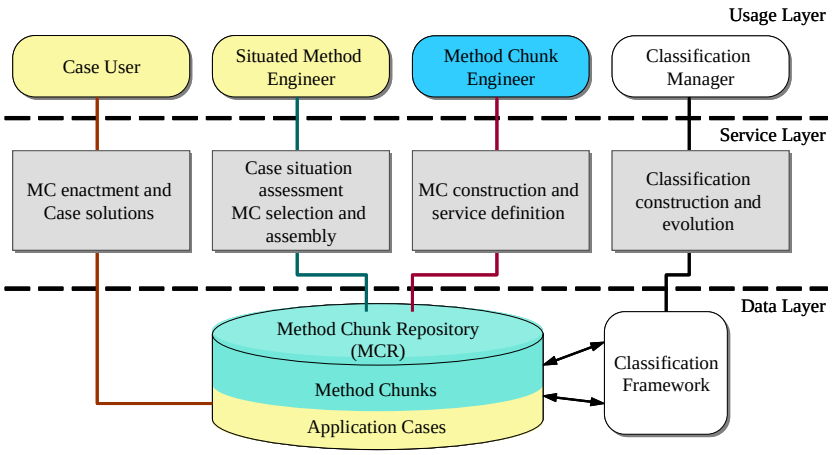

Fig. 1. Architecture for a collaborative method engineering platform

The *data layer* concerns the knowledge repository, called the *Method Chunk Repository* (MCR), used by the platform. This repository contains two types of interconnected knowledge: the method knowledge expressed in the form of reusable method chunks and the knowledge related to the experience of method chunks application in specific industrial cases. A *method chunk* is an autonomous, cohesive and coherent part of a method providing guidelines and defining related concepts to support the realisation of some specific information system development activity in a particular context. Within the scope of the Interop NoE (Interop, 2005), our objective is to define and store method chunks providing solutions for various interoperability issues. The metamodel of a method chunk can be found in (Ralyté and Rolland, 2001; Mirbel and

Ralyté, 2005). Furthermore, the MCR collects the practice and experience of using the method chunks in terms of *application cases*.

Each case included exhibits a number of interoperability issues that instantiate the interoperability issue types identified in the *classification framework*. In order to match the problem situation of a particular case to method chunks thus enabling a solution, we need a mechanism supporting method chunks indexation on the one hand and situation assessment on the other hand. This mechanism is referred to as a matching/classification framework. In this work we focus our attention to the classification part of this framework. Each method chunk stored in the MCR is explicitly related to one or several interoperability issues defined in the classification framework.

The *service layer* of our platform provides several services supporting method engineering and method usage activities including: construction of method chunks and related services, classification framework management (construction, extension, and adaptation), method chunks selection, adaptation and assembly for specific cases, and case-specific method enactment and validation.

Finally, the *usage layer* defines different categories of platform users including: method chunk engineer, classification manager, situated method engineer and case user. The *method chunk engineer* is an expert in the method engineering domain. His/her role is to populate the MCR with method chunks, which can be extracted from existing traditional methods or defined from scratch on the basis of domain knowledge and experience. The method chunk engineer will also develop services for method chunks application and provide a descriptor (Ralyté and Rolland, 2001; Mirbel and Ralyté, 2005) for each method chunk characterising, with the help of the classification framework, the context of its application and the interoperability issues it helps to solve.

The *classification manager* is responsible for defining and managing the method chunk classification framework. Such a framework should be extensible and evolutionary. Good knowledge about the information systems development domain and some selected application or problem domain, such as interoperability in our case, is required to enact this role.

The *situated method engineer* is in charge of constructing a case-specific method for each case. His/her work consists of three main tasks: characterising the case situation by using the classification framework, selecting method chunks satisfying this situation and assembling retrieved method chunks in order to provide a coherent and complete method for the specific case.

Finally, the *case user* will apply the case-specific method in the development of a corresponding project and will provide an experience report including the evaluation of the applied method chunks and their fitness to this case.

The interoperability classification framework forms an important part of the MCR structure since it is used for method chunk classification as well as for case assessment. Therefore, we focus our attention to a set of interoperability problems in order to illustrate its applicability in an industrial case. However, it may be noted that the framework itself comprises a broader scope.

3 Classification of Interoperability Issues

Based on a survey of literature (Rahm and Bernstein, 2001; Xu and Newman, 2006; Botta-Genoulaz et al., 2005; Domínguez and Zapata, 2000) and collective industrial experience in NoE Interop (Interop, 2005) and IP Athena (Athena, 2005) we identify and characterise a set of interoperability issues in enterprise information systems (section 3.1) prior to proposing a classification framework in section 3.2.

3.1 Interoperability Classes

Interoperability issues can be summarised to comprise five different classes:

- *Business Management*,
- *Process Management*,
- *Knowledge Management*,
- *Software Management*, and
- *Data Management*.

In general, they reside in the various levels of the framework presented in (Schultz et al., 2003) including: communication (interconnection and protocols), data (access to and change of information), service (access to and exchange of services/functions), processes (sequences of activities and rules), knowledge (knowledge assets and organisational roles) and business (method of work, legislation and contracts) level. In the following we briefly illustrate these classes in order to characterise the basic conditions for the classification framework..

The utilisation of repetitive business processes across multiple organisations constitutes a potential area for improvement throughout the entire supply chain. Many aspects are generic and involve repeated periodic processing of similar or identical orders. Business decision-making activities are of paramount importance to enterprises, affecting day-to-day operations as well as medium and long-term planning and execution of activities. Therefore, an integral mechanism is required to support the decision-making process at various levels, by considering results coming out of daily operations. The provision of (near) real-time aggregated views of key business information in relation to the above business decision-making activities can be done by accessing and integrating data in existing legacy systems. Such aggregated views will enable actors to take more accurate and timely decisions, exploiting to the full extend the capabilities of existing ICT systems.

The time from order to delivery could be shortened by better process interoperability. This can be achieved by the ability of a process to make its requested and offered services/interfaces “visible”. Shortening the time between different processes, e.g. from raw materials suppliers, has a direct effect on the delivery date. In this context we identify the fact that applications focus on transactions as opposed to processes as an issue to take into account.

The knowledge associated to a product over its entire lifecycle needs to be shared between stakeholders. This entails an adequate and common understanding of product

and process information rather than merely transferring information between stakeholders. Knowledge can be organized according to domain standards.

To make knowledge sharing efficient there is a need for support for stakeholders' collaboration. This implies communication/collaboration infrastructure integration by using standard middleware and communication protocols, which allow the seamless communication and interoperability of model-generated workplace applications. We identify two critical issues in this matter: shared data integration and data access synchronization. Shared data integration entails reconciliation of business level information exchanged between the stakeholders that support collaboration and common understanding.

As enterprises are more and more using commercial of the shelves software (COTS), the used solutions are highly generic and require an important parameterisation/customisation and administration to adapt the solution to the business context. This customisation should be as easy as possible by operators, without implying modification of technical interfaces by software engineers. This fact makes easy customisation of software products and automatic reorganisation of the technical interfaces even more important. The need for documented publication of applications and software product services increases.

Data format interoperability is the ability of a process/application to exchange data with one/more partners by means of a common data format or via a mapping between the proprietary format and an intermediate common format. Hence the enterprise architecture has to take the support of the main technical middleware frameworks in a coherent way into account.

3.2 Interoperability Classification Framework

Recently there has been an increasing interest in creating a body of knowledge for software engineering (Swebok, 2004) and information systems development (Iivari et al., 2004) respectively. These efforts aim to structure relevant knowledge within the areas that they are to cover. Notably, neither of them explicitly deals with the concept of interoperability.

Iivari et al. (2004) propose five ontological domains (in the terminology of Bunge (1983), each of these domains merges both ontological and epistemological aspects) for information systems experts.

The *organisational domain* refers to the knowledge about social contexts and processes in which the information system is used. The *application domain* refers to the knowledge about the application domain for which the information system is intended. The *development process* knowledge refers to the methods and tools used in systems development. The *IT application* domain refers to the knowledge about typical IT applications and their use in a certain application domain. The *technical domain* refers to the hardware and software of an information system. Fig. 2 indicates relationships between the IS ontological domains (first column) and the classes (third column) of interoperability issues (examples in fourth column) identified in section 3.1. In the technology and IT application domains we find issues of data management and soft-

ware management, hence relating the IS field closely to the field of software engineering. Application domain knowledge includes issues concerning business management and process management, i.e. how typical applications work in a particular domain. Finally, organisational domain knowledge has to do with knowledge management in a general sense even though certain issues may be refined to specific application domains.

The software engineering body of knowledge (Swebok, 2004) refines the technology and systems development process knowledge domains of the IS body of knowledge (Iivari et al., 2004) by identifying the following knowledge areas: (software) requirements, design, construction, testing, maintenance, configuration management, engineering management, engineering process, tools and methods and quality. Even though we do not use those exact terms in Fig. 2 we note that, in Swebok (2004), interoperability is seen as an emergent property, which is dependent on the system architecture, with the focus set on software interoperability. Apart from that, not much attention is paid to interoperability. Hence we may add an interoperability aspect to all knowledge areas of Swebok (2004) as well as the IS body of knowledge.

The Enterprise Ontology (2003) is a collection of concepts and their definitions from the business domain. It is subdivided into the aspects activities and processes, organisation, strategy, and marketing. The Enterprise Ontology can be used to further characterise the business domain in a similar way as Swebok (2004) can be used to characterise the ICT development domain.

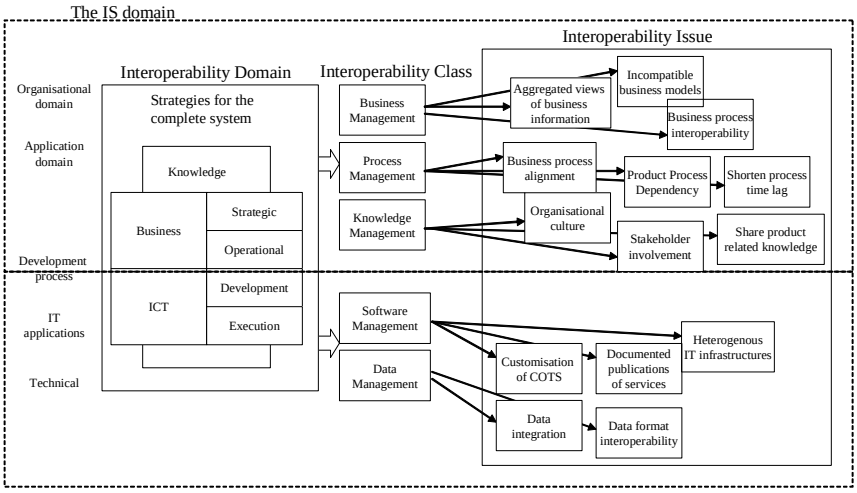

Fig. 2. Interoperability Classification Framework

As indicated in Fig. 2~~Fig. 2~~ we bring together the IS domain and the interoperability domain to provide a structure for classifying interoperability issues. Interoperabil-

ity is not only a problem concerning software and technologies. It is also a problem that concerns knowledge and business references that must be shared in order to achieve interoperability (Chen and Doumeingts, 2003). Thus the interoperability domain comprises knowledge about business/organisational as well as technical aspects (Wainwright and Waring, 2004). Development process knowledge, as described by Iivari et al., (2004), resides in both spheres since the total of all aspects has to be taken into account in IS development.

We envisage an approach to understand the technical, strategic and organisational behaviours from a holistic perspective. That is, organisations are complex and any effort has to handle multiple aspects in order to achieve interoperability between systems. Furthermore, interoperability is a strategic issue; hence interoperability has to incorporate strategic planning for the entire system. To conclude, interoperability between two organisations is a multifaceted problem since it concerns both technical and organisational issues, which are intertwined and complex to deal with.

We also make it possible to incorporate the knowledge areas of the software engineering body of knowledge through the software management and data management interoperability classes. Hence, we propose an extensible classification framework, which is anchored in the information systems body of knowledge and the software engineering body of knowledge. Moreover, we cater for the possibility to allocate problems to the epistemological domain that has the proven methods and knowledge available to solve them. This is particularly useful since real world problems typically concern a combination of ontological areas. The method chunks stored in the MCR are indexed by the classification scheme allowing for better support to all roles identified in the usage layer of Fig. 1.

4 Applying the Classification

A typical case from the real world contains multiple interoperability issues. We use as an example the experience from the *public utility sector*, here the water sector consisting of organisations that *supply fresh water*, organisations that *process sewage water*, and local municipalities that *raise taxes* on both, in particular wrt. the sewage water. In the Netherlands, fresh water supply and sewage water processing are done by organisations that have no need to exchange data since the cost for fresh water supply is based on consumption whereas the cost for sewage water is based on the number of persons in a household. A European guideline stimulates countries to base the sewage water invoice on consumption as well. Since there are no metering devices installed for sewage water per household, the only way to do so is to rely on the metering for the fresh water consumption of the household. To complicate the situation further, the local municipalities use to include taxes on the sewage water invoice that are currently based on sewage water price. As the computation of the sewage water price changes, the tax calculation has to change as well. In the following, we analyse the interoperability problems occurring in the case and classify them into our framework.

IP1: The *business models* of the three organisations are incompatible. The fresh water organisation raises income based on the consumption. The sewage water organisation and the local municipalities use number of persons in a household as basis for their invoice. Moreover, the participating organisations have different concepts for the addressee of the invoice. The fresh water organisation has a concept of a customer linked to a fresh water supply end point. The other two organisations use the concept of a household with a number of citizens associated to it. To integrate the business models of the three organisations, one needs to come up with calculations on a common data basis that fulfils the expectations of the three organisations.

IP2: The *business processes* of the three organisations are not aligned. In particular, the invoicing processes are taking place at different points of time. Specifically, the time when a fresh water invoice is printed is completely independent from the time when the sewage water invoice is printed. The processes for maintaining the customer and citizen data sets in the participating organisations need to be aligned since it may well be that a person is still in the customer data set of the fresh water organisation while already being removed from the citizen data set.

IP3: The three organisations use completely heterogeneous *IT infrastructures*. The data exchange between the local municipality and the sewage water organisation is done by physically sending spreadsheet files on computer-readable media. The fresh water organisation relies on an ERP system to manage all its data and processes. It is unclear whether to use a common platform to which all three organisations supply data, or to send data directly to each other, or for one of the three organisations to play the role of a data integrator.

IP4: The *data structures* are heterogeneous. That holds for all fields relevant for creating the invoices, e.g. the address field, the date field etc. The heterogeneity is resolved by ad hoc procedures to reformat the exchange files. For those parties that do not yet exchange data, the problem of heterogeneity is not yet analysed.

IP5: The *cultural background* and habits in the three organisations is different und difficult to harmonise. The non-profit character of the local municipalities may clash with the more commercial attitude found in the fresh water company. The challenge is to make the right people communicate and exchange information about their respective goals and capabilities. A further complication is that the cooperation is forced upon the participating organisations by the European directive.

Table 1 classifies the five identified interoperability problems into the framework represented in Fig. 2. The classification of the case problem is a manual process and is the first step of the MC enactment and cases solutions service of the MCR. The classification limits the scope of applicable solutions as well as the type of change to be expected from the solution. We applied the following approach for the classification of the case problems:

1. Determine the IS domain of the case problem: The IS domain is characterising the type of knowledge that is necessary to understand the case problem. For example, IP4 belongs to the IS domain 'Development process'. Here, the Swebok (2004) knowledge base can be used to characterise the field.

2. Determine the interoperability domain: This classification characterises the type of interaction that causes the case problem. For example, IP2 is about the alignment of business processes of operational users at different enterprises.
3. Determine the interoperability class: This class is specifying which type of management activity is related to the interoperability problem. It also specifies which expert is to be consulted to solve the problem.
4. Determine the interoperability issue: The set of issues is build upon experience, i.e. whenever a case problem occurs one looks up whether there is a similar issue in the method chunk repository. The issues are the most specific abstractions of past case problems. The interoperability issue is the item that is linked to the potential solutions in the method chunk repository.

This stepwise approach focuses the case user (see Fig. 1) towards the most relevant interoperability issue for the case problem to be classified. By linking the case problem to the respective categories, the case user also pre-selects the group of people to be involved in solving the problem at hand. The closer the case user describes the case problem along the 4 categories, the easier is the classification process. We plan to support the classification by a user interface that provides questions for classifying the case into the first 3 categories and then proposes the most applicable interoperability issues. If no issue is found, an update request for the classification manager of the method chunk repository is formulated.

Table 1. Case classification

Case problem	IS Domain	Interoperability Domain	Interoperability Class	Interoperability Issue
IP1	*organisational*	*business/strategic*	*business management*	*incompatible business models*
IP2	*organisational*	*business/operational*	*process management*	*business process alignment, bp interoperability*
IP3	*IT application*	*ICT/execution*	*data management*	*heterogeneous IT infrastructures*
IP4	*development process*	*ICT/development*	*data management*	*data integration, data format interoperability*
IP5	*organisational*	*business/ operational*	*knowledge management*	*organisational culture*

Out of the five identified case problems, three originate from the organisational domain, i.e. require a solution that is not just a technical one. Only problem IP4 apparently requires to change the IT systems, namely to provide the required data in the right format at the right time. By this example we show how the outcome of the case classification is used to search for applicable method chunks in the repository.

We note that a case like the one discussed above touches multiple interoperability issues, which need to be tackled in an orchestrated effort. An open problem is still whether the solution to a complete case should be regarded as a whole, because the solutions to the interoperability problems highly depend on each other, or whether the individual solutions to the individual interoperability problems should be regarded as stand-alone.

5 Conclusion

In this paper we have outlined a generic architecture for an interoperability method chunk repository. One important aspect of such a repository is a classification framework, which can assist in selecting appropriate method chunks for resolving multifaceted interoperability problems. We note that neither Swebok (2004) nor Iivari et al. (2004) address interoperability explicitly. Hence, we introduce this as a new aspect to take into account. Our results are summarised as follows:

- We propose a generic architecture for a method chunk repository.
- We propose an application of method engineering concepts to organise interoperability knowledge for management and interaction.
- We define a framework consisting of the IS domain, the interoperability domain, interoperability classes, and interoperability issues.
- We show how this framework can be applied to a real world case.
- We propose a stepwise procedure designed to focus the MCR user towards a suitable interoperability issue matching the problem at hand.

The strength of the proposed classification framework is that it incorporates the business/organisational domains as well as the technical domain. Furthermore, the current framework is extendible to comprise more detail.

In order to show the applicability of the framework we have used it to classify a set of interoperability issues in a real world case. In a future application of a similar scheme, we claim that it will be possible to guide a method chunk user in the selection of relevant method chunks. This selection process will be particularly useful in the orchestration of method chunks resolving intra and inter organisational interoperability issues.

Related work concerning tool and platform independence aims to develop a generic platform, which caters for the combination of tools. This is considered an interoperability problem between modelling tools, whereas our work is more focused on domain specific interoperability problems. Hence our work is considered to be a practical utilisation of such a platform. Future work within our project will include the construction of a method chunk repository prototype using the Metis platform (Troux Technologies, 2005). This will lead to a practical application of method chunks within the NoE Interop (2005).

References

ATHENA (2005) European Integrated Project No. 507849. Available at http://www.athena-ip.org. Accessed 2005-12-09.

Backlund, P (2004) An Analysis of ISD as Knowledge Work – an Analysis of How a Development Method is Used in Practice. In Information Systems Development (ISD 2004): Advances in Theory, Practice and Education, pp. 125-136

Botta-Genoulaz V., Millet P.-A. and Grabot B. (2005) A survey on the recent research literature on ERP systems. Computers in Industry (56) pp. 510-522

Bunge, M. (1983). Epistemology & Methodology I: Exploring the World, Treatise on Basic Philosophy Vol. 5, Reidel, Boston.

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

Domínguez E. and Zapata M.A. (2000) Mappings and Interoperability: A Meta-modelling Approach. ADVIS 2000, Ed. T. Yakhno. LNCS 1909, Springer-Verlag, pp. 352-362.

Enterprise Ontology (2003) Enterprise Ontology Project, http://www.aiai.ed.ac.uk/project/enterprise/enterprise/ontology.html. Accessed 2005-12-14

INTEROP (2005) Interop Network of Excellence IST – 508011 Presentation of the Project. http://interop-noe.org/INTEROP/presentation . Accessed 2005-12-07

Iivari, J. (2000) Information Systems Development as Knowledge Work: The body of systems development process knowledge. Information Modellinga and Knowledge Bases XI, IOS Press, pp. 41-56.

Iivari, J., Hirschheim, R. and Klein, H.K (2004) Towards a distinctive body of knowledge for Information Systems experts: coding ISD process knowledge in two IS journals. Information Systems Journal (14) pp. 313-342.

Mirbel, I. and Ralyté, J. (2005) Situational Method Engineering: Combining Assembly-based and Roadmap-driven Approaches. To appear in the Requirements Engineering Journal, electronic publication is available at: http://dx.doi.org/10.1007/11568322_14.

Rahm E. and Bernstein P. A. (2001) A survey of approaches to automatic schema matching. The VLDB Journal, 10, pp. 334-350.

Ralyté J. and Rolland C. (2001). An Approach for Method Reengineering. Proceedings of the 20th International Conference on Conceptual Modeling (ER2001), LNCS 2224, Springer-Verlag, pp.471-484.

Schulz K., et al. (2003) A Gap Analysis; Required Activities in Research, Technology and Standardisation to close the RTS Gap; Roadmaps and Recommendations on RTS activities. Deliverables D 3.4, D 3.5, D 3.6. IDEAS Thematic Network - Contract no.: IST-2001-37368.

SWEBOK (2004) Guide to the Software Engineering Body of Knowledge 2004 Version. Available at http://www.swebok.org/. Accessed 2005-12-08

Troux Technologies (2005) Metis by Troux: Providing the Capabilities for EA Success. http://www.troux.com/ Accessed 2005-12-14

Wainwright D. and Waring T. (2004) Three domains for implementing integrated information systems: redressing the balance between technology, strategic and organisational analysis. International Journal of Information Management 24 (2004) pp. 329–346

Xu X.W. and Newman S.T. (2006) Making CNC machine tools more open, interoperable and intelligent—a review of the technologies. Computers in Industry 57, pp. 141.152.