INTEROP deliverable DTG 6.1 : state of the art: Exploration of methods and method engineering approaches

Network of Excellence - Contract no.: IST-508 011
www.interop-noe.org

Deliverable DTG 6.1

State of the Art: Exploration of Methods and Method Engineering Approaches

Deliverable No. and title:	DTG6.1 State of the Art: Exploration of Methods and Method Engineering Approaches
Classification:	Public
Project Responsible:	Jolita Ralyté, UNIGE #52
Main author:	Per Backlund, Univ.of Skövde #48
Contributing authors:	Jolita Ralyté, Nicolas Arni-Bloch, UNIGE #52 Frank Lillehagen, Troux, #55 Harald Kühn, BOC, #28 Jan Goossenaerts, TU Eindhoven, #14 Brian Elvesæter, SINTEF, #16 Jaana Wäyrynen, KTH, #47
Completion Date:	2005-11-11
Version	Version 0.4

Table of contents

Executive Summary

The main objective of this deliverable is to provide material for the requirements analysis for a method chunk repository. The work has been carried out within INTEROP Task Group 6 starting at the Valencia work shop in May 2005 until the Bologna workshop in October 2005. The deliverable focuses on the state of the art analysis, the concepts and the collaboration platform on which we plan to build the rest of our work to develop, manage, apply and use method chunks and to analyse the impact on key interoperability issues. We strive to reuse available knowledge and results and associate our efforts to the ATHENA project.

IT solutions provided by new technologies, give rise to development and sharing of reusable method chunks. By a method chunk we mean an autonomous and coherent part of a method supporting the realisation of some specific system development or management activity Method chunks and reference models can become visual models in digital form to support user interactive adaptation, extension, use, management and maintenance of other models, knowledge and data. By utilising new technology it is our aim to provide a flexible and useful method chunk repository.

Method chunks can be used to package a variety of solutions as reusable, adaptable services and software components. In this context we find that a repository solution can be used to address a set of interoperability issues which have been identified to have relevance to both industry and research. We describe how a method chunk repository can be organised and populated to provide concrete support in dealing with interoperability in practice.

We have identified a set of interoperability issues from the ontology, enterprise modelling and architecture & platform domains of INTEROP. In order to provide a better scoping of problems, these have been complemented by a set of interoperability issues concerning business management, process management, knowledge management, information management, software management and data management.

The deliverable is a living document which reflects the work plan of Task Group 6. The parts which are completed so far concern the definition of method chunk, concepts and terms, a method chunk meta-model the initial analysis of method chunks, and interoperability issues to be considered in developing the method chunk repository. This content provides the foundation for the requirements on the method chunk repository.

PART I

I.1 Introduction

This deliverable presents the first phase of the work of the Task Group 6: Methods, Requirements and Method Engineering for Interoperability (TG6). This fist phase consists of a state of the art analysis of method engineering using IT, definition of the notion of reusable method chunk, an analysis of method use and available methods deployment of a collaboration platform to develop and apply method chunks and build a Method Chunks Repository (MCR), and an analysis of the interoperability issues involved in developing and using method chunks.

This deliverable presents the work which has been done from M17 to M24. The final results of the deliverable will be presented in M36 according to the TG6 work plan. During the first phase of our work we have also produced a detailed action plan for setting up and populating the repository of method chunks. Since our aim is to let the deliverable be a part of our work process it contains sections which are not yet populated with any detailed results.

This deliverable contains two parts. Part I consists of the general introduction and conclusions of the deliverable. Part II.2 presents a state of the art description and analysis of method engineering and method use. We also present some examples of available methods. Part II.3 characterises the interoperability issues to be solved in order to be able to define the requirements for a method chunk repository.

I.2 Methods to produce the deliverable

The plan for this deliverable is to some extent dependent on the results from deliverables produced in parallel with INTEROP work-packages and other task-groups. Some of these activities are delivering their final results towards the end of 2006. Hence, we do not yet have INTEROP materials to populate all sub-sections, however, there are other relevant sources for content available to us. Where these are quoted or commented on due reference is given.

At the Valencia workshop (May 23rd) we have produced a general structure for the deliverable as a whole, which is aligned to the work plan of TG6. Before and during the Bologna workshop the core team worked to complete the structure and most of the contents of the deliverable.

The sections of the deliverable which are completed so far concern the definition of method chunk, concepts and terms, a method chunk meta-model the initial analysis of method chunks, and interoperability issues to be considered in developing the Method Chunk Repository (MCR).

To summarise, this focuses the state of the art analysis, the concepts and the collaboration platform on which we plan to build the rest of our work to develop, manage, apply and use method chunks and to analyse the impact on key interoperability issues. We strive to reuse available knowledge and results and associate our efforts to the ATHENA project.

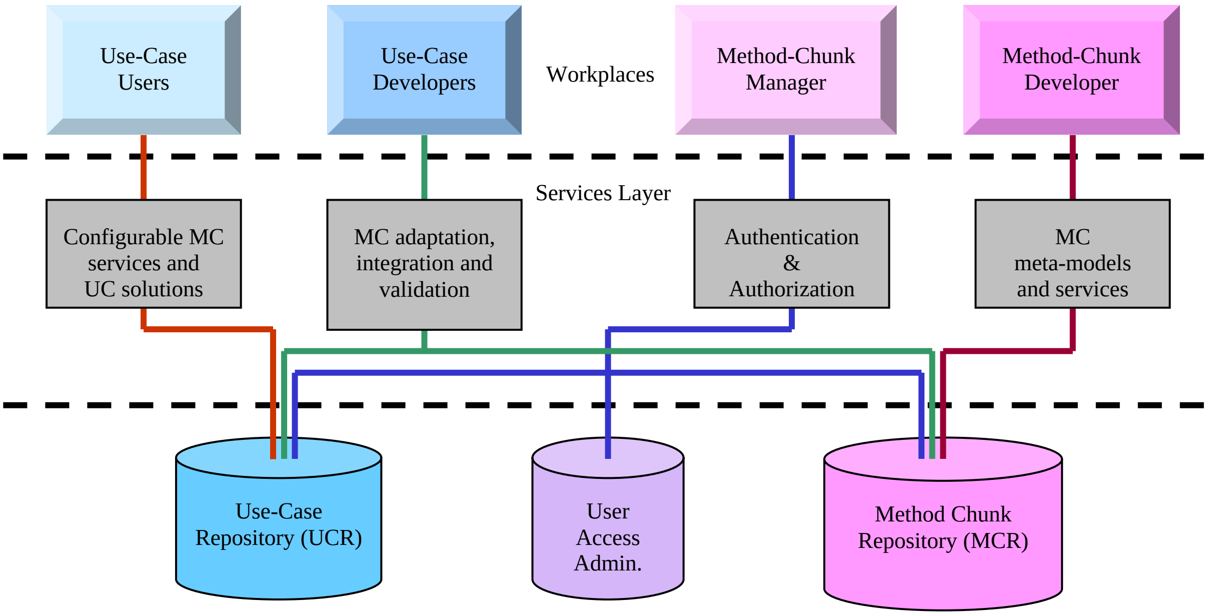

Use-Case Users
Use-Case Developers
Workplaces
Method-Chunk Manager
Method-Chunk Developer
Services Layer
Configurable MC services and UC solutions
MC adaptation, integration and validation
Authentication & Authorization
MC meta-models and services
Use-Case Repository (UCR)
User Access Admin.
Method Chunk Repository (MCR)

Figure 1: The ATHENA MPCE (ATHENA, 2005c) modelling and execution platform will be prototyped by TG6 to support Method Chunk development, provide repository services and support user applications.

The figure above defines the user categories and main capabilities of the work envisioned in TG6. The method chunk developer will develop meta-models integrating methods and parameters, and develop services and a characterisation scheme for each type of method chunk. The method chunk manager will authenticate and authorize the other users to access the respective parts of the repository according to the user categories they belong. The case developers will use the services to extract, adapt and apply the method chunks and develop solutions that will be made available to the respective users.

I.3 Main Results

The results presented in this document were produced during M17-24 of the new INTEROP work programme. These results consist of an analysis of relevant interoperability issues from INTEROP and ATHENA projects. We conclude that these two sources together provide a good overview of the interoperability problems we plan to consider for the INTEROP method chunk repository.

We also provide an analysis of some methods available within the INTEROP and ATHENA communities. We conclude that these projects constitute some specific project and case oriented sets of methods, such as industry specific approaches, methodologies, platforms and solutions. These are characterised by their application in a specific industry domain; and some general methodologies, which are characterised by being applicable across industry domains.

The main results produced by the TG6 and presented in this deliverable will therefore be:

- A state of the art exploration concerning method engineering and method use;
- A definition of a reusable method chunk;
- An architecture for an operational Method Chunk Repository with a user interface to upload, characterise, view, adapt and apply method chunks;
- Brief description of opportunities for new IT solutions provided by new technologies, giving rise to the development and sharing of reusable method chunks;
- Brief description of how method chunks can be delivered to a variety of solutions as reusable, adaptable services and software components;
- An analysis of how Interoperability Issues are impacted by method chunk development and use, emphasizing the benefits and opportunities arising from their use.

In this early version of the deliverable only high-level descriptions on these results are available.

I.4 Conclusions

The first phase of work in TG6 is aimed to analyse the requirements on a method chunk repository for the interoperability domain. After this analysis the Task Group will continue to work on the actual method repository by setting up an infrastructure which will be populated by method chunks with industry relevance.

Our analysis of method use shows that organisational issues such as e.g. work practices have to be considered when introducing a method chunks repository. Furthermore, we have identified a set of interoperability issues concerning ontology, enterprise modelling and architectures & platforms. These will be used to further analyse industry relevant interoperability issues.

We find that method chunks can be implemented as software components, intelligent agents, declarative tasks or as services.

We have identified relevant concepts and interoperability issues in order to be able to carry out a detailed requirements analysis of the method chunks repository.

PART II

II.1 Introduction

Part II presents the main results of this deliverable. It consists of two chapters: Chapter II.2 provides a characterisation of some central concepts: methods, method engineering and organisational method use. These sections contribute to the proposition of how method use can be enabled by a method chunks repository.

Chapter II.3 deals analyses the requirements on methods for interoperability. This is done by a characterisation of interoperability issues relevant to both industry and research. Chapter II.3 will be concluded (in the next phase of work) by a gap analysis and a set of specific requirements for the INTEROP method chunks repository.

II.2 Analysis of Method Use in Organisations

The analysis presented here is focused on method use in industrial enterprises. There exist many definitions of the terms method and methodology. The term methodology is sometimes denoted to mean a collection of methods that together form an integrated and structured approach for work. However, it is not within the scope of this deliverable to further analyse the distinction. Hence we conclude that the term method (or methodology) as used in this deliverable refers to a regular and systematic way of accomplishing a result. Methods cover a wide spectrum of industrial capabilities and services routed in either pragmatic or scientific working methods. Anything taking a well-defined input and yielding a well-defined output can be said to represent a method in the industrial sense. This means that methods do not have to be described in detail by the actions or operations performed on the input to yield the output, but for ease of reuse, composition and management it would certainly be a requirement to have access to the behavioural and functional descriptions. In the context of this work we also conclude that a method may be decomposed into method chunks.

In order to guide our work we will proceed according to a structure of identifying problems, opportunities and directives for the INTEROP Method Chunk Repository. A brief and tentative (because introductory and illustrative) search for problems, opportunities and directives delivers the following short list:

- Problems:
  - there exist no authoritative compilations of method chunks that can be assembled to fit particular project contexts, and such that deliverables of applying one method chunk, can be mapped to inputs of another method chunk;
  - method chunks are rarely presented as elements that are separated from the problems solved with them, or from the cases used to illustrate their application;
  - there is no agreed taxonomy of method chunks;
  - methods are in the heads of people, and not yet in the services of systems;

  - there are no services to assess which methods to use in a given situation, given the bounded rationality of the engineer, often, sub-optimal methods are selected, making projects expensive and crippling system development.

- Opportunities:
  - there exists a near-consensus in the Method Engineering community, on the requirements for a method-chunk repository;
  - there exists a wealth of architectural styles (service oriented, agents,...) that can assist the development of an MCR;
  - platforms such as Athena's MPCE are offering infrastructural services, needs-awareness and proto-communities that are conducive for the MCR development and test.
- Directives:
  - the INTEROP community must find ways to reduce the fragmentation of its knowledge on method engineering;
  - the INTEROP community must find ways to empower enterprises, large and small to address interoperability problems of enterprise software applications.

The methods used in industry can be categorised according to three axes:

1) Whether they represent Approaches, Methodologies, Platform/infrastructure or Solutions;
2) Industrial sector and application area;
3) At which layer of architecture and domain the method applies.

Most approaches are today either represented as guidelines in manuals or as procedures that are delivered by IT providers of tools or systems implementing the method. The majority of methodologies are either manuals, but a rapidly growing number are visual models of varying expressiveness and richness. Platforms and infrastructures for model-configured, user-composed services development and management are emerging, e.g. the MPCE of the ATHENA project being applied to develop our repository. Method chunks for interoperability solutions can typically be guidelines for data exchange, data integration, information logistics mapping, model transformations and comparisons, and for workplace model-design and generation.

Nearly all sector solutions and applications are proprietary IT systems where the methods are rigidly implemented. This often forces the organization to adapt to the systems in order to enable effective use.

Method chunks implemented as meta-models of active knowledge models can in the future be regarded as sets of semantically neutral methods, and if implemented on a platform similar to the ATHENA MPCE it may become just another category of Model-configured, User-composed Platform Services (MUPS).

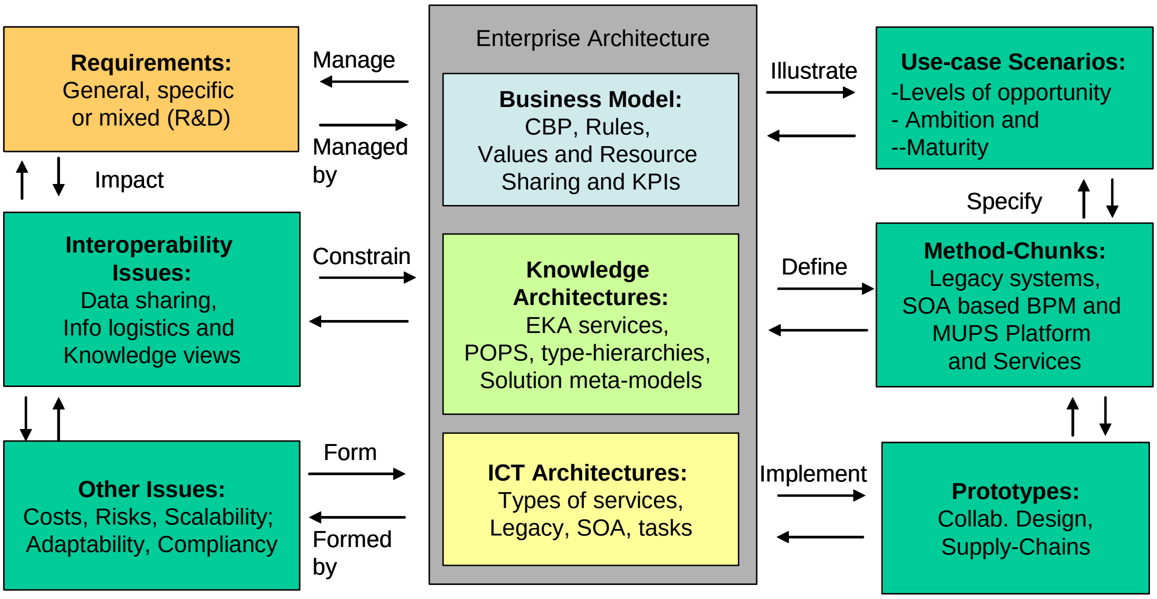

Requirements:
General, specific or mixed (R&D)
Manage
Managed by
Impact
Interoperability Issues:
Data sharing, Info logistics and Knowledge views
Constrain
Other Issues:
Costs, Risks, Scalability; Adaptability, Compliancy
Form
Formed by
Enterprise Architecture
Business Model:
CBP, Rules, Values and Resource Sharing and KPIs
Knowledge Architectures:
EKA services, POPS, type-hierarchies, Solution meta-models
ICT Architectures:
Types of services, Legacy, SOA, tasks
Illustrate
Use-case Scenarios:
-Levels of opportunity
- Ambition and
--Maturity
Specify
Define
Method-Chunks:
Legacy systems, SOA based BPM and MUPS Platform and Services
Implement
Prototypes:
Collab. Design, Supply-Chains

Figure 2: The Enterprise Architecture is the repeatable knowledge representation for all enterprises. With the advent of the Enterprise Knowledge Architecture users can interact with the active knowledge models and familiar views of the enterprise to integrate, adapt, apply and manage the use of method-chunks.

In this context we also identify ADONIS, which is a business modelling tool with components such as information acquisition, modelling, analysis, simulation, evaluation, process costing, documentation, and import/export (BOC 2005). Its main feature is its method independence. This means, that starting from the ADONIS meta tool level arbitrary business modelling tools can be derived. Such tools are represented by so-called (ADONIS) method libraries, which allow in particular the definition of arbitrary modelling languages without any programming effort.

The ADONIS customizing architecture consists of three layers where metamodelling capabilities form the core (Karagiannis, Kühn 2002). In ADONIS, a modelling language is specified by defining its object-oriented metamodel. Basically, a metamodel consists of model types, classes, relation types, inter-model relation types, and their attributes.

We have analysed the reference models of relevance to the ATHENA use-cases and test-cases for interoperability and for the purpose of re-engineering them. Of particular interest is the MPCE platform with its Enterprise Knowledge Architecture (EKA), (ATHENA, 2005c). Using this platform and the services that will be provided in version 2.0 of MPCE, many of the reference models can become visual models in digital form to support user interactive adaptation, extension, use, management and maintenance of other models, knowledge and data.

II.2.1 Method Engineering

Many different definitions of a method can be found in the literature. Generally speaking, a method describes a regular and systematic way of how to accomplishing something. In the domain of Information Systems engineering, Brinkkemper (1996) defines a method as “an approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rule, structured in a systematic way in development activities with corresponding development product". According to Booch (1991), a method is "a rigorous process allowing to generate a set of models describing different perspectives of the system under development by using some well defined notations".

From the engineering perspective (see Figure 3), a method is made up of a set of product models and a set of corresponding process models. A product model represents the concepts that are used in the method, relationships between these concepts as well as constraints that they have to satisfy. A process model represents the way to accomplish the development of the corresponding product.

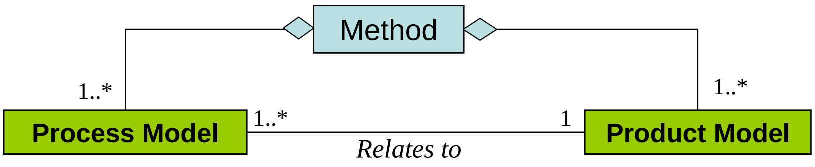


Figure 3: Engineering view of a method.

II.2.1.1 Method Engineering – Process

The discipline dealing with the design, construction and adaptation of methods, techniques and tools for the development of software and information systems is called Method Engineering.

Because of the fact that engineering situations vary considerably from one application system development project to another the traditional systems development methods are often not well suitable. Even though they claim to be universal and propose a large number of models and views for system analysis and specification, they cannot foresee all possible development situations. As a consequence, each software and information systems project requires for a specific method and tool to support the development process. It is clear that traditional method construction techniques are too expensive and time-consuming and are not well appropriate to tackle the project-specific method construction. As a reaction to this problem, the notion of Situational Method Engineering (SME) was first proposed by Kumar and Welke in 1992 as a new method engineering discipline the aim of which is to construct new methods and the associated tools or to adapt existing ones to every system development project. A number of SME approaches have been already proposed in the literature. Most of them use *assembly* techniques based on the reuse of existing method parts in the construction of new methods or in the enhancement of existing ones. Figure 4 summarises the process of Situational Method Engineering based on two core steps:

1. The reengineering of existing methods/models/ideas into reusable method chunks and
2. The engineering of new situation-specific methods by selecting and assembling method chunks.

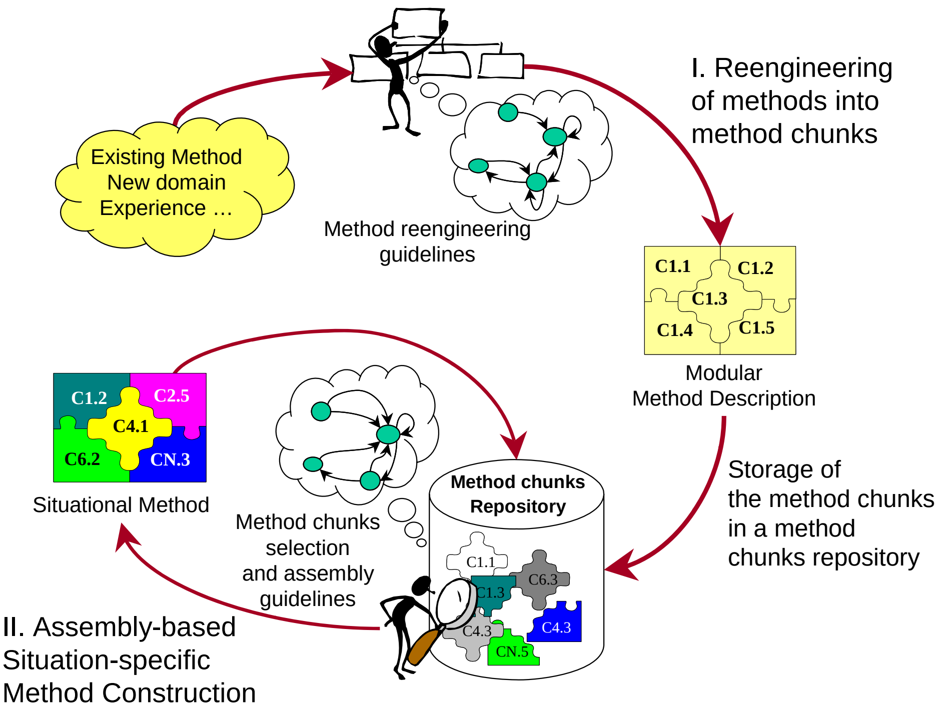

Existing Method
New domain
Experience …
Method reengineering guidelines
I. Reengineering of methods into method chunks
C1.1
C1.2
C1.3
C1.4
C1.5
Modular Method Description
Storage of the method chunks in a method chunks repository
Method chunks Repository
C1.1
C6.3
C1.3
C4.3
C4.3
CN.5
Method chunks selection and assembly guidelines
C1.2
C2.5
C4.1
C6.2
CN.3
Situational Method
II. Assembly-based Situation-specific Method Construction

Figure 4: Process of situational method engineering.

II.2.1.2 Method Engineering - Concepts

Method chunk. Based on the observation that any method has two interrelated aspects, product and process, several authors propose two types of method components: process fragments and product fragments (Harmsen et al., 1994; Brinkkemper et al., 1998). Other authors consider only process aspects and provide repositories of process components/fragments (Firesmith and Henderson-Sellers, 2002).

Based on the propositions given in (Ralyté and Rolland, 2001b; Wistrand and Karlsson, 2004, Prakash, 1999), we integrate the process and product aspects in the same module that we call a *method chunk*. A method chunk is an autonomous and coherent part of a method supporting the realisation of some specific system development or management activity. Such a modular view of methods favours their adaptation, configuration and extension. Moreover, this view permits to reuse chunks of a given method in the construction of new ones.

As a part of a method, a method chunk ensures a tight coupling of some process part of a method process model and its related product part. In the product part the product to be delivered by the method chunk is captured whereas in the process part the guidelines allowing to produce the product are given. For example, the method chunk providing guidelines (process part) for a use case model construction should also provide definitions of the concepts as actor, use case, scenario, extend relationship, etc. (product part) used in this model. As shown in Figure 5 introducing the notion of the method chunk, the corresponding product and process parts constitute the *body* of the method chunk.

The *interface* of the method chunk captures the context in which the method chunk can be applied. It is formalised by a couple <*situation*, *intention*>, which characterises the situation that is the input of the chunk process and the intention (the goal) that the chunk achieves. "To construct a use case model" is an example of the intention associated to the method chunk providing guidelines for the use case model construction; the "Problem statement" specifies the situation in which this chunk can be applied that is the input product necessary to start the execution of this method chunk.

Besides, a *descriptor* is associated to every method chunk. It extends the contextual view captured in the chunk interface to define the context in which the chunk can be reused. The descriptor takes values from the method chunks classification framework and associates them to the corresponding method chunk.

The detailed meta-model of method chunk and the guidelines supporting method chunks definition will be provided in the deliverable DTG6.2.

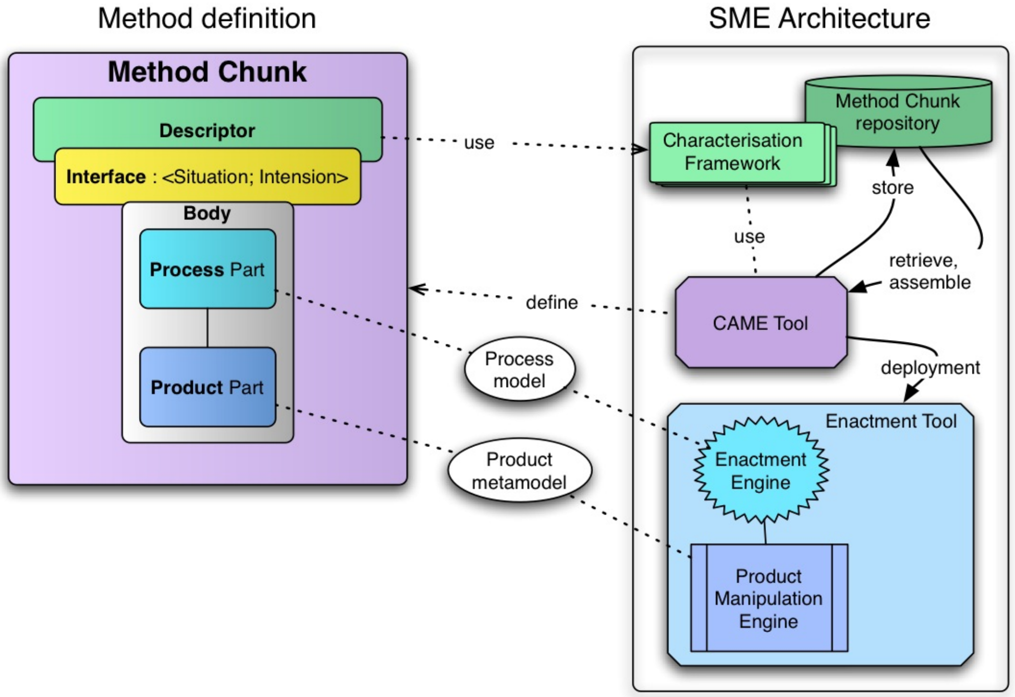


Figure 5: Notion of the method chunk.

Method Repository. The prerequisite for modular method construction is a method repository containing a large collection of method chunks. Different propositions for method repositories are given in (Saeki et al., 1993; Harmsen et al., 1994, Ralyté, 1999) but currently none of them is functional. Our objective is to provide a method repository (Figure 5) containing a collection of method chunks supporting different issues dealing with interoperability of enterprise applications and platforms. Method chunks to be stored in the repository can be obtained by decomposition of existing methods or by creating them from scratch in order to support some specific system engineering activity.

Method Engineering Techniques. Situational Method Engineering deals with assembly-based method construction and adaptation techniques as well as modular method configuration techniques.

Following assembly-based approaches (Brinkkemper et al., 1998; Ralyté and Rolland, 2001a), new methods can be constructed by selecting method fragments/chunks from a method repository in such a way that they must fit project method requirements. Method configuration techniques deal with modular method adaptation to a given project situation (Bajec et al., 2004; Wistrand and Karlsson, 2004) or a particular engineer task (Mirbel and Ralyté, 2005).

In our work we plan to use an assembly technique for situation-specific method construction and configuration (Figure 5).

Supporting Tools. The method construction process asks for software support. Computer Aided Method Engineering Tools (CAME) have been designed and developed for this purpose. As shown in Figure 5, a CAME tool should provide support for the following method engineering activities: evaluation of the current project situation and determination of requirements for a project-specific method, storage of method chunks in a method repository, retrieval and assembly of method chunks, validation, verification and adaptation of the obtained situational method.

So far, a number of meta-CAISE products and prototypes such as MetaEdit+ (Kelly et al., 1996), MViews (Grundy and Vanable, 1996) and Concept-Base (Jeusfeld et. al., 1998) have been developed. They provide help in methods and method chunks definition but do not support the obtained method execution. A complete SME architecture should also include a method enactment engine and guidance mechanism supporting the development and elaboration of a product following the method process specification.

Our objective is to develop a tool supporting all the required facilities for SME.

II.2.2 Available Methods

This section will present the methods identified in the State-of-the-art (SoA) descriptions from year 1 of INTEROP (www.interop-noe.org). Furthermore, we are also concerned with identifying methods from other areas of interest. We identify the ATHENA project as one important source of information in this aspect.

In general we identify two perspectives of interest: *industry specific and general methods*. *Industry specific* methods are characterised by their application in a specific industry domain. We refer to this as vertical classification, i.e. they are specific to finance, telecommunication, logistics, manufacturing etc.

Some examples, which represent a purposeful sample, of industry specific methods are:

- The NGOSS framework (Tele Management Forum) is the New Generation Operations Systems and Software that the telecom industry worldwide is using as a guide to help the evolution of its processes and systems (TMF 2005). Its' aim is to provide a technology and process roadmap which allows a simplified implementation of business process automation coupled with significantly improved business flexibility and agility.

- The Supply-Chain Operations Reference-model (SCOR) is a process reference model that has been developed and endorsed by the Supply-Chain Council for supply-chain management (SCOR 2005). SCOR is a process reference model for supply-chain management, spanning from the supplier's supplier to the customer's customer. The SCOR-model has been developed to describe the business activities associated with all phases of satisfying a customer's demand. By describing supply chains using process building blocks, the Model can be used to describe supply chains that are very simple or very complex using a common set of definitions.
- The Insurance Application Architecture (IAA) is a comprehensive set of insurance specific models that represents best practices in insurance and is a natural extension to the Component Business Model (IAA 2005). IAA describes the business of the insurer and is an efficient communication bridge between business and technology communities. It is designed to be readily accessible to business users and by focusing on industry issues such as Sales and Customer Services, Marketing and Analytics, Customer Relationship Management, Core Systems, Insurance Claims and Risk and Compliance. IAA comprises:
    - Foundation Models: insurance terms and definitions for communication and standardization
    - Information Models: insurance data content for an enterprise-wide view of information and data rationalization
    - Process Models: insurance business processes content for areas such as business process modelling, simulation and execution
    - Integration Models: business services content for component based development and services oriented architectures
    - Product Models: a method for accelerating insurance product design.

General methods are characterised by being applicable across industry domains. We refer to this as horizontal classification. Such methods are typically focused on issues such as information systems engineering, enterprise architecture management, competence management etc.

Some examples of general methods are:

- A variety of Enterprise Modelling Languages (EMLs) exist, aiming to describe processes and other enterprise aspects. Most EMLs are implemented in tools with proprietary terminology, modelling constructs and templates. Each modelling language employs a particular syntax, and limited, tool-embedded semantics and graphical notations. The tools support conflicting modelling paradigms. Hence the foundations of Enterprise Modelling, Engineering and Visual Management are fragmented and lack formal definitions. Unified Enterprise Modelling Language (UEML) intends to solve the problem of multiple Enterprise Modelling Languages. It was developed in a series of projects: IDEAS, UEML and INTEROP. UEML's main objective is to provide industry with a unified and expandable modelling language for enterprise modelling (UEML 2005a, UEML 2005b).
- The RAIS methodology from ATHENA. "RAIS – Requirements, Architecture, Interoperability issue and Solutions bring together all the aspects of requirements

engineering and analysis for the design and development of appropriate solutions. The purpose of RAIS is to identify, understand and describe interoperability issues and support gap analysis towards implementation approaches and technologies. This can be done by modelling the dependencies among these different concepts." (ATHENA, 2005a, p. 6)

- The Enterprise Knowledge Development from the Hyper-knowledge project. "EKD is an approach that provides a systematic and controlled way of analysing, understanding, developing and documenting an enterprise and its components, by using Enterprise Modelling. […] The deliverables of the EKD Process are a number of conceptual models that examine an enterprise and its requirements from a number of interrelated perspectives. These models are abstractions from the physical world. For a given enterprise, these models will collectively constitute the Enterprise Model. (Bubenko et al., 2001).

A study of existing reference models (ATHENA, 2005b) has revealed a need to provide modern development environments, integrated modelling and execution platforms, to the organizations developing and delivering the approaches, methods and partial solutions that these reference models represent. Furthermore, the ATHENA MDI (Annex 1) methodology may be defined in terms of a set of reusable method chunks, which may contribute to populating the method chunk repository.

Based on our study of ATHENA case and relevant reference models and the solution approaches, methods and platforms being developed in ATHENA, one conclusion is that the re-engineering initiatives and the industry sector initiatives must get more attention to exploit the momentum and awareness of the potential values to be harvested. Following ATHENA, other projects will naturally re-engineer the many reference models that Athena will (a) not have resources nor (b) competence or (c) rights to deal with.

There is clearly a need for increased user interaction and services to support continuous stakeholder involvement, and interaction in verifying and adapting data, information and knowledge in extended enterprises, including industry sector and community reference models.

II.2.3 Method Use

This section will explore some of the assumptions underlying organisational use of methods. The aim is to characterise method use with respect to different usage situations and perceptions of what a method is. We find these aspects important to promote a better use the proposed method chunk repository. Hence this section will characterise the method in use as opposed to the method in concept in order to better analyse what kind of support is needed.

To start our analysis, we identify a set of usage aspects, which indicate the diversity of method use:

- Syntactical issues (modelling syntax etc.),
- Lifecycle issues (which phases are relevant in relation to certain method chunks),
- Organisational standards (e.g. organisational decisions to use certain methodologies),
- Intended platform (database, web environment),

- Competence (what competences are needed in different situations and how does it evolve over time),
- Data representation,
- Context (situations in which the method chunks are used),
- People aspects (concerning the individual developer's knowledge),
- Organisational aspects (e.g. division of labour).

Organisations use methodologies to enhance their work. Russo and Stolterman (2000) explore the assumptions underlying information systems methods. Some interesting assumptions in the context of our work are:

- The existence of knowledge about good design practice. There is an assumption that development process knowledge can be made explicit in the form of methods. This may well be the case but it is an entirely different thing to use these normative descriptions.
- The ability to communicate design knowledge to practicing designers. This assumption concerns the tension between knowledge and ability. According to Russo and Stolterman (2000) it becomes more complicated to transfer knowledge of good design if design practice is considered to be an ability.
- The possibility to change the way practicing designers view the design process. A method is meant to change the behaviour of the method user. This assumption explains the large number of methods on the market.

Iivari (2000) presents the notion of information systems development (ISD) as knowledge work and distinguishes three components in information systems development knowledge: knowledge of information technology; knowledge of the application domain and ISD process knowledge. The concept does not refer to codified knowledge only. Iivari (2000) also provides a categorisation of ISD work into: routine, craft-like, professional, and creative knowledge work. Craft-like knowledge work is essentially skill-based and can only be learned through apprenticeship and practical experience. Skilful operation is characterised by an ability to recognise problems, diagnosing their cause and applying appropriate corrective procedures. Craft-like and professional knowledge work is further characterised by three features (Hirschheim and Klein, 2003): its close relationship to a person's identity which requires hard work and mistakes to acquire; its connection to personal emotions and interests which makes it dependent on social interaction and socialisation, and; its holistic nature which makes difficult to split into goals and means.

The above views of method use are characterised by the potential tension between the method in concept and the method as used. In order to characterise the use of methods three "tensions" are described; the concept usage tension, the concept implementation tension and the product usage tension (Lundell and Lings, 2004). For our work, the concept usage tension (between the method proponents and the developer) is of interest. This tension describes a discrepancy between how a method is described and how it is being used in practice. Figure 6 summarises these tensions.

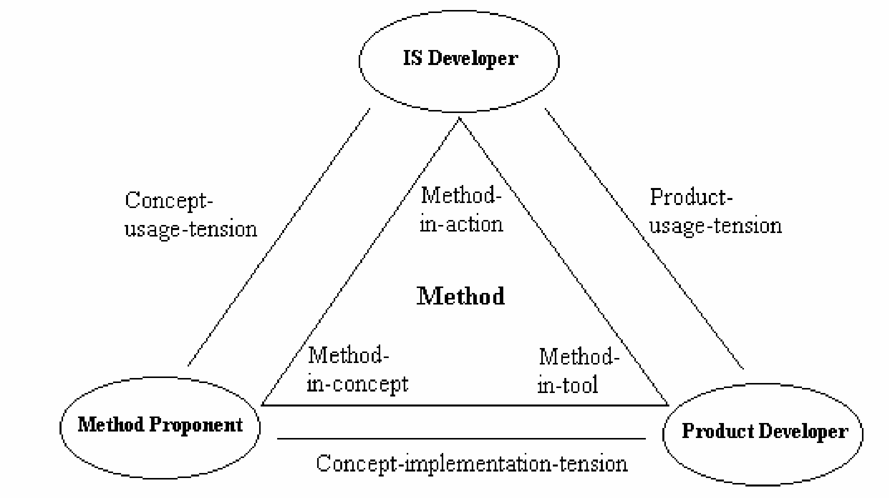


Figure 6: The stakeholder triangle and its tensions. From Lundell and Lings (2004).

A similar notion of the tension between method in concept and method in use is present in the method in action framework (Fitzgerald et al., 2002) Fitzgerald and O'Keane (1999) and Backlund (2004) also characterise the tension between method in concept and method in use.

The broad scope of development methods implies there are different roles involved in using them, e.g. analysts, developers, future system users, and project managers. There may be an explicit use of a specific method in an organisation or there may be implicit use of a method in terms of the internalised knowledge and ways of working among the developers. Finally, the method may serve different roles in systems development (Iivari and Maansaari, 1998). A method may serve as:

- a rule to determine or regulate action;
- a resource to support action;
- a reminder of actions to be taken;
- a model of the ideal process that may not be possible to follow in practice;
- a vehicle of learning.

We find it essential to create a method chunk repository which supports work in various ways since method use is a multifaceted concept.

II.2.3.1 Industry Aspects of Method Use

IT solutions have for the last 40 years been delivered to industry as application systems off the shelf. There are many business and technical reasons for this, such as general routines for solution versioning, deployment and training and customer support. This means that methods are programmed and hidden for users and part of the application code.

The consequences of this for industry and use of methods are two-fold:

1. Methods are not easily re-executed for iterations and impossible to modify or reuse in other solutions.

2. Methods cannot be adapted and combined into larger chunks of methods, and this cannot be performed by users as they perform design or engineering work that by nature have to be more cyclic to support the growing demand for collaboration and concurrency.

In ATHENA the first prototypes of Model-configured, User-composed IT solutions and services, built on integrated modelling and execution platforms, have been demonstrated and the application of this approach to method chunks in specific use-cases have helped us to identify some significant added values, such as those described below.

The value of methods to industry users is crucial. We emphasise the demands of industry to IT solutions and services to support:

- Growing ***stakeholder involvement*** in business and project initiatives;
- Traceable ***user interaction*** to support more collaboration and concurrency;
- Model-driven systems integration;
- Knowledge model ***Solutions interoperability*** by exploiting operational knowledge architectures.

In given industrial settings these criteria and qualities can be observed, measured and monitored in interactive dashboards. This is done by IT governance services as provided by eg. Troux Technologies.

II.2.4 Method Use Enabled by a Method Chunk Repository

We will identify a set of cases or scenarios (for an example, one case is presented in Annex 2) which will be used to illustrate, refine requirements and to facilitate training, learning and repository use. More specifically:

- To illustrate situation assessment in an enterprise and the activities of method chunk selection and assembly;
- To refine requirements for the Method Chunk Repository (MCR) (feeding into Task 2.2). To propose the architecture, contents and services for the INTEROP method chunks repository and to develop the prototype of the INTEROP method repository (Task 2.3);
- To facilitate training by providing case material to training courses that will result from Task 3.3. To establish, train and use the ATHENA Collaboration Platform, its Method Engineering approach and workplaces.

In a sense the cases and scenarios play a pivotal role through the MCR life cycle which can be modelled as in Figure 7 and Figure 8.

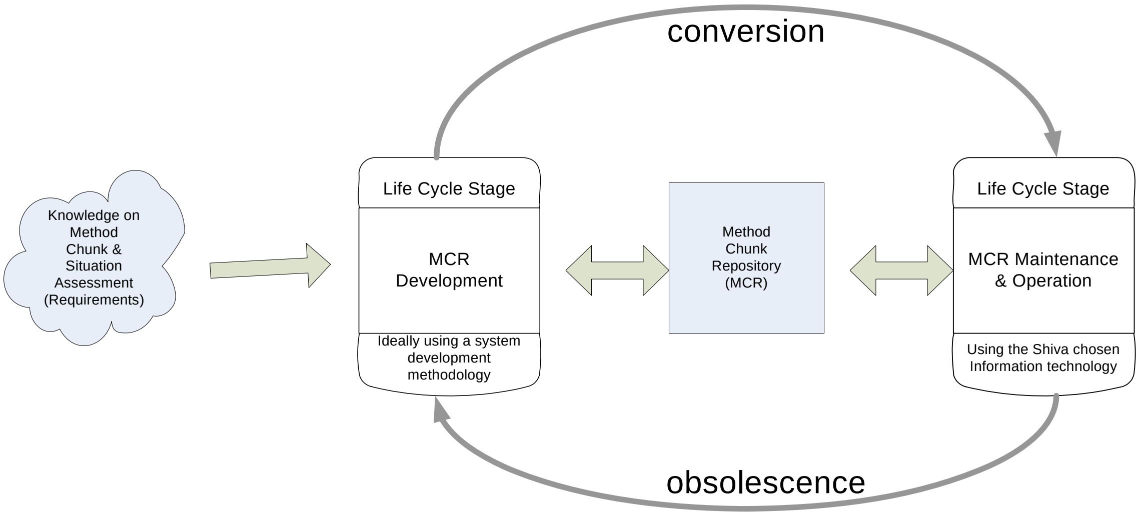

conversion
Knowledge on Method Chunk & Situation Assessment (Requirements)
Life Cycle Stage
MCR Development
Ideally using a system development methodology
Method Chunk Repository (MCR)
Life Cycle Stage
MCR Maintenance & Operation
Using the Shiva chosen Information technology
obsolescence

Figure 7: The Life Cycle of the Method Chunk Repository.

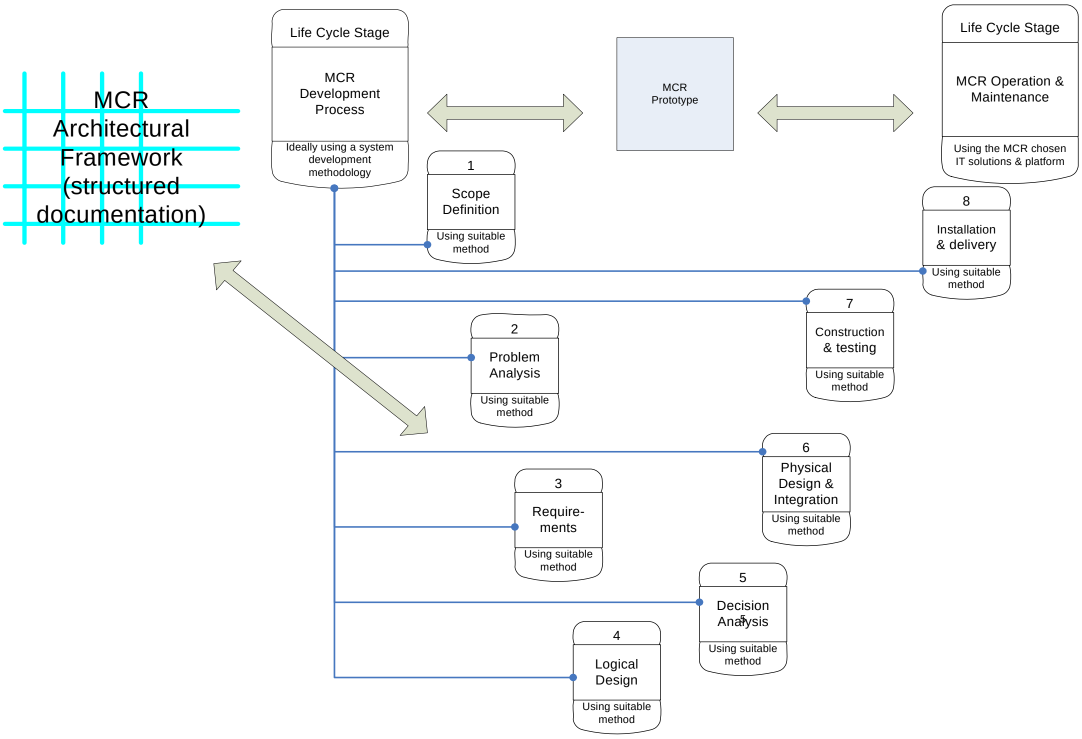


Figure 8: Development sub-processes of the Method Chunk Repository.

The MCR development sub-processes, i.e. the scenario analysis, is described in more detail in Annex 3.

To validate the method chunks and repository use we will analyse a number of scenarios. These scenarios are selected on the basis of an expectation that they will expose the practise of working with method chunks.

The remainder of this section will provide a more detailed account for the actual test cases that will be developed during the next phase of work.

II.2.5 References for Section II.2

ATHENA (2005a) ATHENA Mapping Approach Validation Report. WD B4.6.2. ATHENA Project No. 507849.

ATHENA (2005b) Global Processes of Reference. WD D.B4.2. ATHENA Project No. 507849.

ATHENA (2005c) MPCE Specification. Deliverable A1.5.1. ATHENA Project No. 507849.

Backlund, P. (2004) Development Process Knowledge Transfer through Method Adaptation, Implementation, and Use. Doctoral Thesis, Department of Computer and Systems Sciences, Stockholm University/Royal Institute of Technology.

BOC (2005). BOC and ADONIS Homepage. www.boc-eu.com, 2005.

Booch, G. (1991). Object Oriented Analysis and Design with Applications. Benjamin/Cummings.

Bajec, M., Vavpotic, D., Kirsper, M. (2004). "The scenario and tool-support for constructing flexible, people-focused system developement methodologies". Proceedings of the international conference on Information Systems Development, ISD 2004. Vilnius, Lithuania, September 2004.

Brinkkemper S., Saeki, M. & Harmsen, F. (1998). Assembly Techniques for Method Engineering. Proceedings of the 10th International Conference on Advanced Information Systems Engineering, CAiSE'98. Pisa, Italy, LNCS 1413, Springer.

Brinkkemper, S. (1996). Method engineering: engineering of information systems development methods and tools. Information and Software Technology, Vol. 38. Elsevier Science B.V. pp. 275-280

Bubenko, J., Persson, A., Stirna, J. (2001) EKD User Guide. HyperKnowledge IST–2000-28401. URL: ftp://ftp.dsv.su.se/users/js/ekd_user_guide_2001.pdf Last accessed 2005-10-04

Firesmith, D.G. and Henderson-Sellers, B., 2002, The OPEN Process Framework – An Introduction. Addison-Wesley, Harlow, UK

Fitzgerald, B., O'Kane, T. (1999) "A Longitudinal Study of Software Process Improvement," *IEEE Software*, pp. 37-45.

Fitzgerald, B., Russo, N.L., Stolterman, E. (2002) Information systems development: methods in action. London, McGraw-Hill.

Grundy, J.C. and Venable, J.R. (1996). Towards an Integrated Environment for Method Engineering. In Challenges and Strategies for Research in Systems Development. W.W. Cotterman and J.A. Senn (Eds.), John Wiley & Sons. Chichester, pp.45-62.

Harmsen A.F., Brinkkemper, S. and Oei, H. (1994). Situational Method Engineering for Information System Projects. In Olle T.W. and A.A. Verrijn Stuart (Eds.), Mathods and Associated Tools for the Information Systems Life Cycle, Proceedings of the IFIP WG8.1 Working Conference CRIS'94, pp. 169-194, North-Holland, Amsterdam.

Hirschheim, R. and Klein, H. (2003) Crisis in the IS Field? A Critical Reflection on the State of the Discipline, *Journal of the Association for Information Systems*, 4, 237-293.

IAA (2005) IBM Insurance Application Architecture – Insurance Business Models. IAA0001-MK. www.ibm.com, 2005.

Iivari, J. (2000) Information Systems Development as Knowledge Work: The body of systems development process knowledge In *Information Modelling and Knowledge Bases XI* (Eds, Kawaguchi, E., Hamid, I. A., Jaakkola, H. and Kangassalo, H.) IOS Press, pp. 41-56.

Jeusfeld, M.A., Jarke, M., Nissen, H.W.M., Staudt, H.W. (1998). ConceptBase - Managing Conceptual Models about Information Systems. *In P. Bernus, K. Mertins, G. Schmidt (eds.): Handbook on Architectures of Information Systems*, Springer-Verlag, 1998, ISBN 3-540-64453-9, pp. 265-285.

Karagiannis, D. and Kühn, H. (2002) Metamodelling Platforms. In: Bauknecht, K.; Min Tjoa, A.; Quirchmayer, G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France, September 2-6, 2002, LNCS 2455, Springer-Verlag.

Kelly, S., Lyyttinen, K. and Rossi, M. (1996). Meta Edit+: A fully configurable, multi-user and multi-tool CASE and CAME environment. Proceedings of the CAiSE'96 Conference, Heraklion, Crete, Greece, LNCS 1080, Springer Verlag.

Kumar, K. and Welke, R.J. (1992). Method Engineering, A Proposal for Situation-specific Methodology Construction. In Systems Analysis and Design : A Research Agenda, Cotterman and Senn (eds), Wiley, pp257-268, 1992.

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

Mirbel, I. and Ralyté, J. (2005). Situational Method Engineering: Combining Assembly-based and Roadmap-driven Approaches. *To be published in the Requirements Engineering Journal.*

Prakash, N. (1999). On Method Statics and Dynamics. Information Systems. Vol.34 (8), pp 613-637.

Ralyté, J. and Rolland, C. (2001a). An Assembly Process Model for Method Engineering. Proceedings of the 13th International Conference on Advaced Information Systems Engineering, CAISE'01, Interlaken, Switzerland LNCS 2068, Springer, pp. 267-283.

Ralyté, J. and Rolland, C. (2001b). An approach for method reengineering. Proceedings of the 20th International Conference on Conceptual Modeling, ER2001, Yokohama, Japan, LNCS 2224, Springer, pp.471-484.

Ralyté, J. (1999). Reusing Scenario Based Approaches in Requirement Engineering Methods: CREWS Method Base. Proc. of the First International Workshop on the Requirements Engineering Process - Innovative Techniques, Models, Tools to support the RE Process, Florence, Italy, September.

Russo, N. L. and Stolterman, E. (2000) Exploring the assumptions underlying information systems methodologies Their impact on past, present and future ISM research, *Information Technology & People*, 13,

Saeki, M., Iguchi, K., Wen-yin, K., Shinohara, M. (1993). A metamodel for representing software specification & design methods. Proceedings of the IFIP¨WG8.1 Conference on Information Systems Development Process, Come, pp 149-166.

SCOR (2005) Supply Chain Operations Reference Model. Supply Chain Council Homepage. www.supply-chain.org, 2005.

TMF (2005) URL: www.tmforum.org Last accessed 2005-10-04

UEML (2005a) UEML Homepage. www.ueml.org, 2005.

UEML (2005b) INTEROP Homepage. www.interop-noe.org, 2005.

II.3 Requirements on Methods for Interoperability

II.3.1 Introduction

This section will describe the requirements put on the INTEROP method chunk repository. We need to identify a set of requirements which have relevance both to industry and research. Hence it is our aim to let our analysis subsume both the work that has been done in INTEROP and ATHENA as well as other industry relevant perspectives.

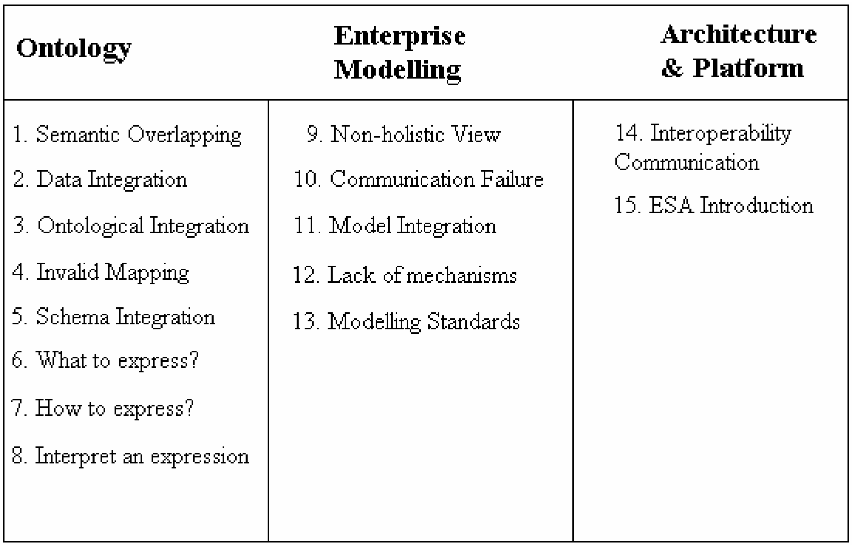
The section is organised as follows: We characterise interoperability problems from the INTEROP perspective (II.3.2.2). Section II.3.3.2 identifies and exemplifies a set of categories of interoperability issues. The reminding sections (to be completed in the next phase) include a requirements analysis of the derived interoperability issues from a method chunk repository point of view.

II.3.2 Characterisation of Interoperability Issues

This section presents a set of interoperability issues identified in the State-of-the-art (SoA) descriptions (Berio and Tami, 2004; Berre at al., 2004; CNR-IASI et al., 2004; Doumeingts and Berre, 2004; Petit et al., 2004) from year 1 of INTEROP. Furthermore, we are also concerned with identifying interoperability problems from the ATHENA project as well as from other areas of interest. The INTEROP method chunk repository will cover a subset of the issues identified.

II.3.2.1 Gathering and Analysing Interoperability Issues

The INTEROP NoE produced a set of SoA descriptions which were analysed to identify relevant interoperability problems to include in a method repository. The problems were elicited by means of an extensive literature survey and classified with respect to the sub-areas at the current time (Figure 9) of INTEROP: Ontologies, Enterprise Modelling and Architecture & Platforms. We note that this is not a classification to be proposed for the repository, it is rather to be perceived as a means for organising these results.



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

Figure 9: Summary of the elicited interoperability problems.

The interoperability problems identified are further described in the form of patterns (Alexander, 1999; Coplien and Schmidt, 2005) in Ottosson (2005), which may serve as descriptors for methd chunks. We note some problems concerning the identification and construction of these patterns. Firstly, it is problematic to select a useful level of granularity and level of abstraction for each interoperability problem. Secondly, the knowledge from these patterns has to be organised and related to each other in some way. These problems have been further analysed in Rolland et al. (2000). We propose a method chunk repository as a potential solution to some aspects of this problem. For example, a method repository provides the possibility to relate method chunks to each other n order to provide structure.

To complement the interoperability problems elicited from the INTEROP SoA reports we introduce the ATHENA perspective of the same issue. We identify 31 interoperability issues (Lillehagen, 2005) which are further described in ATHENA (2005a). Some examples are: data format interoperability, inconsistency of data and process interoperability. The initial analysis shows that there is some overlap between the two sources. However, this does not constitute a problem since the overlap is limited and the two sources represent two different perspectives of the interoperability domain. The INTEROP material provides a generic view of systems engineering problems, whereas the issues identified in ATHENA stem from a set of use cases in ATHENA (2005) and material in ATHENA (2005b).

By combining INTEROP sources with ATHENA sources we have produced an extensive characterisation of current interoperability problems.

II.3.2.2 Categories of Interoperability Issues

The interoperability issues are analysed according to the architecture knowledge layers and domains identified in Figure 2 of this report. For simplicity, they are sorted according to business, process, knowledge, information and data management. Most of the interoperability issues have relevance for more than one domain and their solution may impact the whole Enterprise Architecture; i.e. all knowledge dimensions and domains.

Table 1 presents an overview of the interoperability issues identified. Each issue is further described in the subsequent sections. The following abbreviations are used: Business management (BM), Process management (PM), Knowledge management (KM), Information management (IM), Software management (SM), Data management (DM).

Table 1: Categories of interoperability issues

Issue	BM	PM	KM	IM	SM	DM
Business Processes "hard-coded" in applications	BM 1					
Manual business processes.	BM 2					
Linking decision-making activities	BM 3					
Aggregated views of business information	BM 4					
Translating strategic goals	BM 5					
Inferring benefits of objectives	BM 6					
Support effective decision-making	BM 7					
Focus on process management		PM 1				
Process interoperability		PM 2				
Shorten time from order to delivery		PM 3				
Supplier rating		PM 4				
Product descriptions			KM 1			
Product related knowledge			KM 2			
Knowledge organisation based on domain standards			KM 3			
Knowledge governance processes			KM 4			
Enterprise description and knowledge management			KM 5			
Integrated application execution			KM 6			
Stakeholder involvement			KM 7			
Negotiation space based on models			KM 8			
Integration and federation of objectives			KM 9			
Link viewpoints for decision-making			KM10			
Identities are hard-coded				IM 1		
De-coupled layers					SM1	
Customisation of software products					SM2	
Auto descriptive applications					SM3	
Internal information model					SM4	
Documented publication of applications and services					SM5	
Data format interoperability						DM1
Distributed inconsistent data						DM2
Support for middleware framework						DM3

Business Management

BM 1: Business Processes "hard-coded" in applications: improvement of application deployment in terms of business internal "Time to Market" and high programming code reuse level.

BM 2: Repetitive manual business processes for regular bulk orders. Much of the manufactured products are generic and this involves repeated periodic processing of similar or identical orders.

BM 3: Link decision-making activities with strategic plans, development and operational results. Business decision-making activities are of paramount importance to enterprises, affecting both the 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 with strategic plans, by considering product/project development activities and results coming out of daily operations.

BM 4: Provision of (near) real-time aggregated views of key business information. Related to the above business decision-making activities, these aggregated views could be provided as services to the roles and actors required, 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.

BM 5: Target setting decomposition and objectives mapping to process roles. Among the major challenges large enterprises are facing today is the capability to translate strategic goals into detailed tactical and operational objectives and targets for every business unit and major business process. The timely execution of and the ability to re-adjust and fine-tune this activity upon fluctuations of market conditions can have a significant impact on the profitability and, in some occurrences, on the survival of these enterprises within the extremely competitive environment they operate. Therefore, there is a need to develop a mechanism to structure and facilitate that process of business strategy translation into business process objectives, attributable to the different roles of its key personnel.

BM 6: Objectives inferring to tangible benefits and expectations. One step further to the above issue, the detailed tactical and operational objectives per role have to be justified by the benefits and expectations one can bring back into each role, and all together into the business process/unit, thus justifying the budget and other resources to be allocated for its realisation. Therefore, there is a need to develop a similar to the above mechanism to inference role objectives into attainable benefits and expectations.

BM 7: Link program, product and enterprise viewpoint to support effective decision-making, strategic plans, development and operational results. Several viewpoints are structuring management of activities within an enterprise and a networked organisation. Each person has consequently different antagonist objectives, with sometimes unclear definition of priority. Linking program, product and enterprise viewpoints should allow more effective decision making, negotiation and activities within the collaborative product design within a networked organisation. It should also allow a better alignment between decision making, strategic plans, development and operational results.

Process Management

PM 1: Applications focus on transactions, not on processes: usage of graphical tools to manage process parameterisations and process management in a virtual enterprise context, gaining programming activities reduction/reset (code implementation).

PM 2: Process Interoperability: ability of a process or application to make "visible" the requested services/interfaces and the offered services/interfaces.

PM 3: Lag. Time from product order to delivery could be shorter. Shortening time from ordering to receiving raw materials from the supplier has a direct effect on the delivery date of the finished product.

PM 4: Time spent rating supplier. Many companies conduct tri-monthly reviews of their suppliers to ensure that standards are kept.

Knowledge Management

KM 1: Confusion resulting from poor product descriptions. Clients very often order the wrong products!

KM 2: Product related knowledge sharing within and between product life cycle phases. Adequate and common understanding of product and process information is required to support knowledge sharing between different product life-cycle phases, rather than merely transferring information between them.

KM 3: Domain standards based knowledge organisation. The knowledge meta-meta model should allow integration of concerned industrial sector meta-models, ICT used solutions meta-models and Enterprise Modelling used solutions meta-models.

KM 4: Establishment of Knowledge governance process, standards and best practices for a networked organization without governance and long term strategy, islandisation of knowledge applications will lead to non-interoperability.

KM 5: Enterprise description and knowledge management in various aspects and dimensions (organisation, role, decision, process, product, system) knowledge capture, assimilation and management are some of the most important assets of an enterprise, an asset that creates significant added value to any existing information and communication infrastructure.

KM 6: Ability of integrated applications execution via custom, adaptive and model generated environment. Legacy applications integration and interoperability: existing applications that provide access to enterprise data and facilitate analysis and decision-making should be integrated using a standard technology that allows composition at service level, thus providing the reusability and flexibility of customized services composition and deployment. Model driven generation of interoperable custom and role-based workplaces: Models mapping and integration at system level, as well as tools for the transformation of the provided models to interoperable service description interfaces would allow the interoperability of system models and model generated workplaces.

KM 7: Support for stakeholders' involvement and collaboration:

Communication / collaboration infrastructure integration: use of standard middleware and communication protocols would allow the seamless communication and interoperability of model-generated workplaces applications.

Shared data integration: Reconciliation of business level information exchanged between the stakeholders that would allow their collaboration and common understanding is required. This probably implies business data integration at semantic or business meta-models level with the use of reference ontologies and/or mapped meta-models.

Data and data access synchronization: Working concurrently on the same data requires a synchronization mechanism that preserves their consistency and validity and distributes their valid state to the interested stakeholders.

KM 8: Negotiation space based on objectives models used by the enterprises of the networked organisation. Enterprise Modelling tools are used to support elaboration of enterprise objectives and roles, supporting quality trends (ISO 9001, CMM, CMMI, etc), but level of maturity of enterprises for usage of such tools or quality approach is not the same. To benefits from modelling and models, but also to help the less mature enterprises to integrate such tools, some neutral standards are necessary in order to prepare negotiation workspace between the actors of Collaborative Product Design actors in a networked organization.

KM 9: Integration and federation of Objectives to tangible benefits and expectations used models. Different and heterogeneous tools (Activity Based Costing, System Engineering, Scorecard, etc.) are used to link objectives to tangible benefits in one hand, to the expectation and requirement in the other hand, often within the same enterprise. A negotiation and decision workspace should support quick and easy federation or integration of these tools to efficiently support enterprise and program management and decision making.

KM 10: Link program, product and enterprise viewpoint to support effective decision-making, strategic plans, development and operational results. Several viewpoints are structuring management of activities within an enterprise and a networked organisation. Each person has consequently different antagonist objectives, with sometimes unclear definition of priority. Linking program, product and enterprise viewpoints should allow more effective decision making, negotiation and activities within the collaborative product design within a networked organisation. It should also allow a better alignment between decision making, strategic plans, development and operational results.

Information Management

IM 1: Identity and identification schemes are hard-coded. This makes cooperation and collaboration process modelling and execution very difficult. Inbound and outbound logistics have to be designed from knowledge structures, and services provided to decode and align logistics schemes.

Software Management

SM 1: De-coupled application layer and technical layer. In order to support agility of global information system, and independence between business logic and technical solutions implementing the awaited functionality. It should allow interchange-ability of software product components and real governance of the information system by enterprise and networked organisation.

SM 2: Easy customisation of the software product and automatic reorganization of the technical interfaces. As enterprises are more and more using Commercial of the Shelves, 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.

SM 3: Auto descriptive applications. Capability from the software product solution to extract the business logic as business or enterprise model, with a standard and open format, in order to support custom, adaptive and model generated collaboration environment parameterisation.

SM 4: Internal information model of software products and applications based on standardised information models capacity for the networked organisation to rely on stable and software product independent business models to establish their collaboration, with minor impact of technical solutions evolution.

SM 5: Documented publication of applications and software products services. The idea is to allow easy usage of these applications when willing to establish collaboration, without used solution experts. It should be done through Application Programming Interfaces and service description according the numerous interface description standards (IDL, WSDL), for IT department and ICT integrators. It should be done through well structured and agile documentation that should be reusable by knowledge models. All these interfaces should be coherent and easily reflect customisation, an automated way.

Data Management

DM 1: Data Format Interoperability: 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.

DM 2: Distributed inconsistent data: ability of a solution to guarantee data consistency and distributed data alignment in a virtual enterprise context.

DM 3: Support of the main technical middleware framework a coherent way. It includes STEP ISO information technical platform, CORBA and OMA, eBusiness infrastructure (Web services), Wfmc standards, J2EE and .NET. It aims to be able to easily collaborate with partners that have made some choices based on these technical platforms a seamless way.

II.3.3 Cases and Scenarios of Interoperability

This section will be finalised during M24-M36.

This section will identify and describe a set of cases for method chunks selection and assembly. The process of identifying and working with the cases is further described in II.2.4 and Annex 3.

II.3.4 Situation assessment

This section will be finalised during M24-M36.

Responsible person: Jolita Ralyte

This section will be based on the strategy for situation assessment presented by Jolita in Geneva during the meeting in conjunction with the I-ESA Conference.

The section will contain a classification of situations and method chunks for interoperability which can be applied to the cases we collect in the next phase.

II.3.5 Requirements for ME for interoperability

This section will be finalised during M24-M36.

Responsible person: Jolita Ralyte

Contributions: Per Backlund

We need an initial repository structure (compare Frank Lillehagens presentation from ATHENA RequirementsElaboration_20050706, which can serve as a good starting point)

In the ATHENA WD B 4.2 (ATHENA, 2005b) chapter 7 there is a list of criteria for task and service assessment, and we have adapted and extended this list to help us define user values most important criteria are:

- Repeatable approach
    - Planning process well defined
    - Planning process as work patterns for establishing
    - Criteria for measuring progress
    - Training services on the reference model
- Compliant methodologies
    - Language of the POP* family
    - Language extension services to cover new uses and users
    - Consistent with reference model source definition
- Interactive adaptation and composition (open platform and repository services)
    - Services for systems integration and data capture

  - Services for model transformation and mapping
  - Runtime plug-and-play software components
  - Compatible infrastructures and IT solutions
- User-composed tasks and services (work management support)
  - Task definition, task lists, trees and nested structures
  - Task swim-lanes for work monitoring
  - At least two kinds of execution platforms

These requirements are derived from an analysis of the findings in section II.2.2.3.

II.3.6 Gap analysis

This section will be finalised during M24-M36..

Responsible person:

Contributors: All

Identify and analyse the gap between the interoperability issues we have identified in our characterisation of interoperability and the existing methods/method chunks.

What are the requirements for formulating methods for interoperability?

II.3.7 Method chunk requirements defined

This section will be finalised during M24-M36..

Responsible person:

Contributors: All

A compiled list of the problems our chunks will solve (i.e. requirements for the content of the chunks).

II.3.8 References for Section II.3

Alexander, C. (1999) The Origins of Pattern Theory The Future of the Theory, and the Generation of a Living World, *IEEE Software*, vol. 19, pp. 71-82.

ATHENA (2005a) ATHENA Mapping Approach Validation Report. WD B4.6.2. ATHENA Project No. 507849.

ATHENA (2005b) MPCE Specification. Deliverable A1.5.1. ATHENA Project No. 507849.

Berio, G., Tami, C. (2004) Mapping Representation Languages. WP 5, INTEROP Network of Excellence - Contract no.: IST-508011

Berre, A., Hahn, A., Akehurst, D., Bezivin, J., Tsalgatidou, A., Vermaut, F., Kutvonen, L., Linington P.F., (2004) State-of-the art for Interoperability architecture approaches" - Model driven and dynamic, federated enterprise interoperability architectures and interoperability for non-functional aspects. WP 9, INTEROP Network of Excellence - Contract no.: IST-508011

Chen, D. (2004) Design Princilples for Interoperability. WP 6, INTEROP Network of Excellence - Contract no.: IST-508011

CNR-IASI, UNIROMA–DIS, UNITILB, KTH. (2004) State of the art and state of the practice including initial possible research orientations. WP 8, INTEROP Network of Excellence - Contract no.: IST-508011

Coplien, J.O., Schmidt, D.C. (1995) Pattern Languages of Program Design. Reading, Massachussets: Adison-Wesley Publishing Company.

Doumeingts, G., Berre, A. (2004) State of the art of methods to derive IT specifications from models and to develop IT specific components based on IT modelling. WP 7, INTEROP Network of Excellence - Contract no.: IST-508011

Lillehagen, F. (2005) Interoperability Issues, copied from the three matrices. Internal documentation.

Ottosson, A. (2005) An Analysis of a Content of a Method Chunk Repository concerning Interoperability Problems. Master Thesis HS-EA-DVA-2005-001, University of Skövde.

Petit, M., Krogstie J., Sindre, G (2004) Knowledge map of research in interoperability in the INTEROP NoE, 1st version. WP 1, INTEROP Network of Excellence - Contract no.: IST-508011

Rolland, C., Stirna, J., Prekas, N., Loucopoulos, P., Persson, A., Grosz, G (2000) Evaluating a Pattern Approach as an Aid for the Development of Organisational Knowledge: An Empirical Study. In the proceedings of CAiSE 2000 (Eds.) Wangler, B., Bergman, L.

Ralyté, J. (2002). Requirements Definition for the Situational Method Engineering. IFIP TC8/WG8.1 Working Conference on Engineering Information Systems in the Internet Context, Kanazawa, Japan.

Annexes

Annex 1

Case Study: ATHENA MDI Framework for SOA and Web Services

This section presents the model-driven interoperability (MDI) framework for service-oriented architecture (SOA) and Web services that is currently being developed in the ATHENA A5 and A6 projects. We believe this methodology will provide a useful case study for INTEROP TG6 in which ATHENA can use the framework of TG6 to define method chunks that addresses how SOA and Web services should be designed, developed and integrated in order to achieve interoperability.

1. Introduction

System interoperability is a growing interest area, because of the continuously growing need of integration of new, legacy and evolving systems, in particular in the context of networked businesses and eGovernment. Enterprise applications and software systems need to be interoperable in order to achieve seamless business across organisational boundaries and thus realise virtual networked organisations. We see SOA and Web services as important enabling technologies to achieve the goals of these networked organisations.

In the ATHENA A5 and A6 project we are studying how the model-driven development (MDD) approach can help us to design and develop interoperable service-oriented architecture (SOA) and adaptive software architecture (ASA) system solutions. We are defining a model-driven interoperability (MDI) framework that will provide guidelines for how MDA should be applied to develop interoperable Web services.

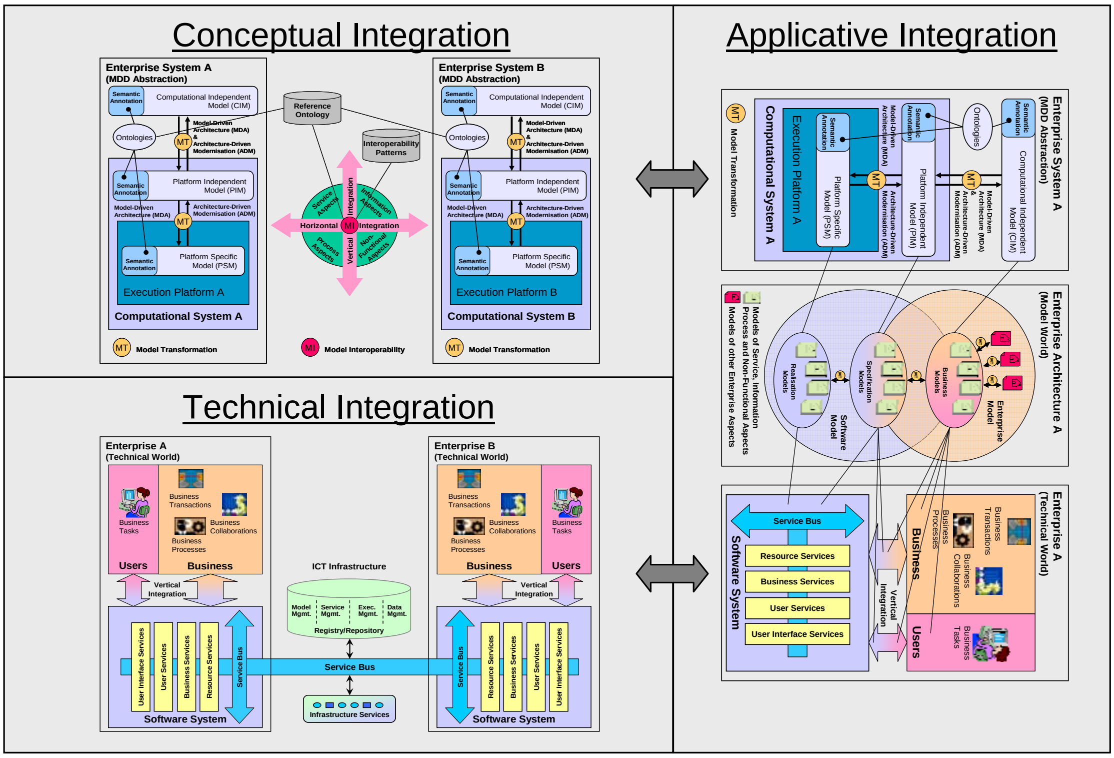


Figure A1.1: ATHENA MDI framework

The ***ATHENA MDI Framework*** [1] depicted in Figure A1.1 is structured according to three main integration areas; conceptual integration, technical integration and applicative integration. The conceptual integration focuses on concepts, metamodels, languages and model relationships to systemize software model interoperability. The technical integration focuses on the software development and execution environments. The applicative integration focuses on methodologies, standards and domain models. It provides us with guidelines, principles and patterns that can be used to solve software interoperability issues.

2. Conceptual integration

The MDI framework defines a platform-independent metamodel for service-oriented architecture (*PIM4SOA*) and *Web service metamodel* for the most important Web service specifications. The metamodels are formalized using the meta-object facility (MOF). This allows us to develop model transformations according to the OMG Model Driven Architecture (MDA) approach where transformations between models are defined using their metamodels. Amongst the WS-* standards covered are:

- Web service descriptions (WSDL [2, 3])
- Web service compositions (WS-BPEL [4], JACK BDI [5])
- Web service addressing (WS-Addressing [6])
- Web service security (WS-Security [7], WS-SecureConversation [8], WS-Trust [9], WS-SecurityPolicy [10])
- Web service registry (UDDI [11])

- Semantic annotation of Web services (WSDL-S [12], OWL-S [11]).

3. Applicative integration

The *ATHENA baseline methodology for SOA* [13] introduces a model-driven development (MDD) approach to specifying interoperable service-oriented architectures realized as Web services. In model-driven development one uses models to describe business concerns, user requirements, activities, information structures, components and component interactions. These models govern the system development in that they can be transformed to program code. We aim to develop tools to automate model transformations for service-oriented architectures. Hence, the term model-driven development in our context encompasses both the development of models, and tools for model transformation.

The models are expressed in UML, and supported by UML profiles for SOA and Web services. The baseline methodology provides guidelines for how to develop the different kinds of models recommended for SOA. Some of them lay the basis for automated code generation; all of them contribute to the understanding and specification of the system or services to be developed.

Figure A1.2 illustrates the ATHENA MDI methodology, covering enterprise models, SOA models and Web service models. The focus of the ATHENA A5 and A6 projects is on the SOA and Web service models, supporting both a top-down as well as a bottom-up approach, for developing new and integrating existing Web services in a SOA supporting the enterprise.

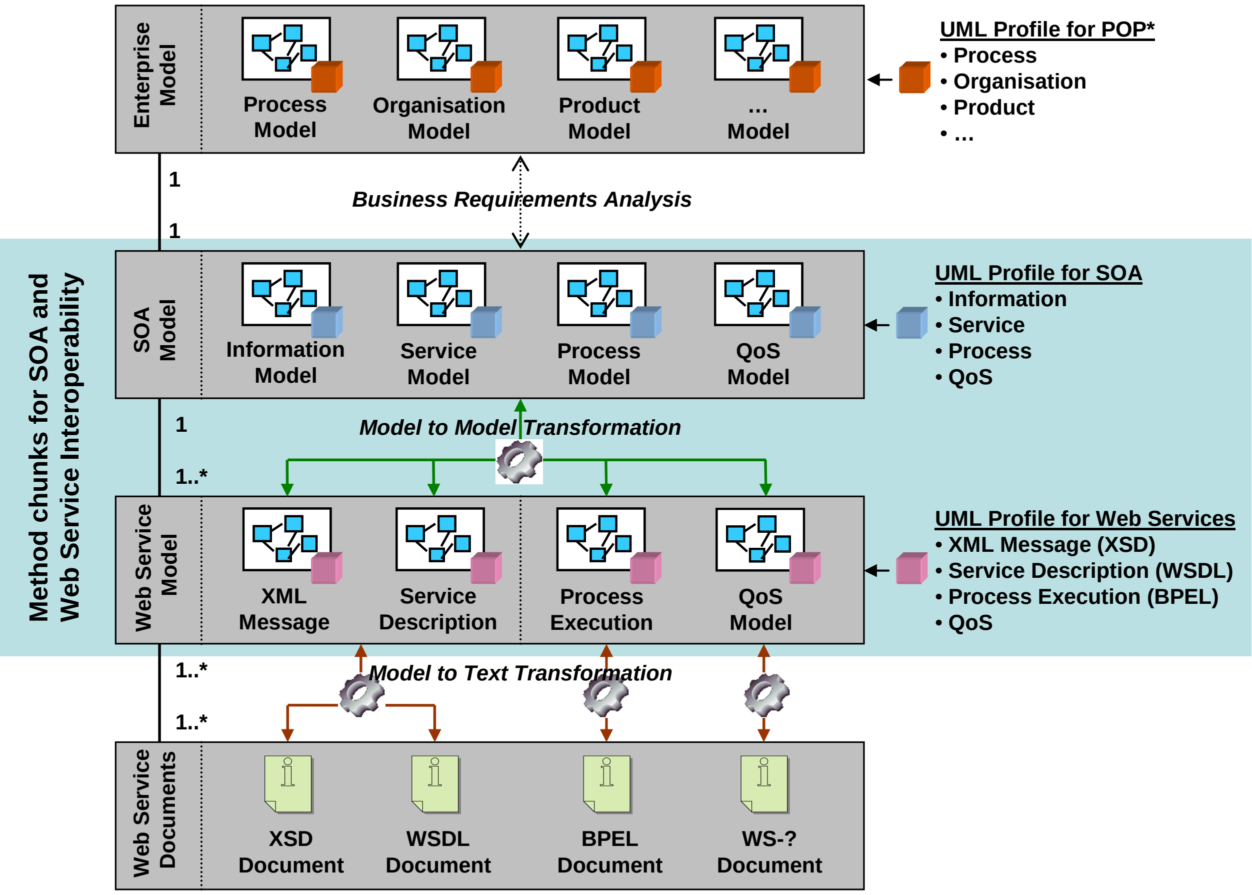


Figure A1.2: ATHENA MDI methodology for SOA and Web services

4. Technical integration

ATHENA aims to provide development and integration tools to support MDD for SOA and Web services. As part of this goal, we have developed a *UML profile for SOA* and a *UML profile for Web services*. A UML profile is an extension of the UML metamodel that represents a level of abstraction between the model and the implementation environment. A UML profile uses UML model elements that have been customized for a specific domain or purpose using extension mechanisms as stereotypes, tagged values and constraints. The UML profile supports the process from an idea to a Web services system and allows the analyst and the developer to work in an organised way. The SOA and Web service profiles and corresponding model transformation tools have been implemented as a feature for Rational Software Modeler (RSM) [14].

5. Relevance for INTEROP TG6

The INTEROP TG6 provides a method engineering (ME) framework that can be used to structure and define reusable method chunks. In the development of the ATHENA MDI methodology we aim to use the templates provided by TG6 to define a set of method chunks (guidelines and techniques) that addresses how SOA and Web services should be designed, developed and integrated in order to achieve interoperability.

References

[1] ATHENA A6, "D.A6.1: Specification of a Basic Architecture Reference Model", ATHENA IP, Deliverable D.A6.1, 2005. https://www2.mycomputas.com/Team/Repository/Projects/Project_209/Upload/Attachments/WPs/WPA6.1/D.A6.1/D.A6.1.doc

[2] W3C, "Web Services Description Language (WSDL) 1.1", World Wide Web Consortium (W3C), W3C Note, 15 March 2001. http://www.w3c.org/TR/wsdl

[3] W3C, "Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language", World Wide Web Consortium (W3C), W3C Working Draft, 3 August 2004. http://www.w3.org/TR/2004/WD-wsdl20-20040803/

[4] BEA Systems, IBM, Microsoft, SAP AG, and Siebel Systems, "Business Process Execution Language for Web Services Version 1.1", 5 May 2003. ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf

[5] AOS, "JACK Intelligent Agents", The Agent Oriented Software Group (AOS). http://www.agent-software.com/shared/home/ (accessed: 2005).

[6] W3C, "Web Services Addressing Working Group", World Wide Web Consortium (W3C). http://www.w3.org/2002/ws/addr/ (accessed: 2005).

[7] IBM, Microsoft, and VeriSign, "Web Services Security (WS-Security) Version 1.0", 5 April 2002. ftp://www6.software.ibm.com/software/developer/library/ws-secure.pdf

[8] Actional, BEA Systems, Computer Associates, IBM, Layer 7 Technologies, Microsoft, Oblix, OpenNetwork Technologies, Ping Identity, Reactivity, RSA Security, and VeriSign, "Web Services Secure Conversation Language (WS-SecureConversation)", February 2005. ftp://www6.software.ibm.com/software/developer/library/ws-secureconversation.pdf

[9] Actional, BEA Systems, Computer Associates, IBM, Layer 7 Technologies, Microsoft, Oblix, OpenNetwork Technologies, Ping Identity, Reactivity, RSA Security, and VeriSign, "Web Services Trust Language (WS-Trust)", February 2005. ftp://www6.software.ibm.com/software/developer/library/ws-trust.pdf

[10] Microsoft, VeriSign, IBM, and RSA Security, "Web Services Security Policy (WS-SecurityPolicy)", Draft, 18 December 2002. http://www-106.ibm.com/developerworks/webservices/library/ws-secpol/

[11] OASIS, "UDDI Version 3.0.2", Organization for the Advancement of Structured Information Standards (OASIS), UDDI Spec Technical Committee Draft, 19 October 2004. http://uddi.org/pubs/uddi_v3.htm

[12] LSDIS, "METEOR-S: Semantic Web Services and Processes", Large Scale Distributed Information Systems (LSDIS). http://lsdis.cs.uga.edu/projects/meteor-s/ (accessed: 2005).

[13] ATHENA A5, "D.A5.2: Model and Specification of Service Descriptions and Usage as well as Advanced Concepts", ATHENA IP, Deliverable D.A5.2, 2005. http://athena.troux.com/Team/Repository/Projects/Project_208/Upload/Attachments/Results/Deliverables/D.A5.2/D.A5.2.doc

[14] IBM, "Rational Software Modeler". http://www-306.ibm.com/software/awdtools/modeler/swmodeler/ (accessed: 2005).

[15] ATHENA A6, "D.A6.1 Annex 3: Evaluation of Existing Tools (UML 2.0) for System Specifications", ATHENA IP, Deliverable D.A6.1-A3, 2005.

https://www2.mycomputas.com/Team/Repository/Projects/Project_209/Upload/Attachments/WPs/WPA6.1/D.A6.1/D.A6.1-A3_UML20.doc

[16] Eclipse.org, "Eclipse.org Main Page". http://www.eclipse.org/ (accessed: 2005).

Annex 2

Interoperability problem: Distributed Federated Simulation of Supply Chains

Name and institution of the author: Frank-Walter Jaekel (IPK)	
Origin of the knowledge captured: WP 6 - D6.1 / II.3.3	
1.	Interoperability level
- PROCESS; DATA	
2.	Interoperability barrier
- CONCEPTUAL; TECHNOLOGICAL	
3.	Interoperability problem
Distributed Federated Simulation of Supply Chains: The large enterprises are interested in expanding their own approaches in the field of "Digital Factory" to their suppliers and the entire value chain. This can already be seen in the grown pressure on the suppliers to increase transparency of their processes and to supply planning data. A selection of problems are: CONCEPTUAL: (*) Different representations in different simulation packages, (*) different understanding of the simulation context, (*) different ways for simulation, (*) trust between companies and data quality, (*) different terminologies such as “entity” versus “object”, etc.. TECHNOLOGICAL: (*) different simulation tools, (*) different simulation technologies, (*) high diverse of animation and monitoring features, (*) different ways of model distribution, (*) different maturity of the models, (*) different data descriptions (Frames, Objects,…), etc..	
4.	Interoperability knowledge
The global concept based on the results of the European MISSION Module, the High Level Architecture (HLA) and on an extension of the Enterprise Modelling Method IEM (Integrated Enterprise Modelling). It illustrates an instance of distributed federated simulation of supply chains. The models can be located within different enterprises and executed within different Off-the-Shelf Simulation Packages. Therefore, two levels of interoperability are required. First the Off-the-Shelf Simulation Packages has to provide the opportunity of being interoperable and second the models executed within the tools have to be synchronised and interoperable. Today several Off-the-Shelf Simulation Packages supports the required opens. On the base of the open API the interfaces can be developed which are specified in the MISSION approach.	

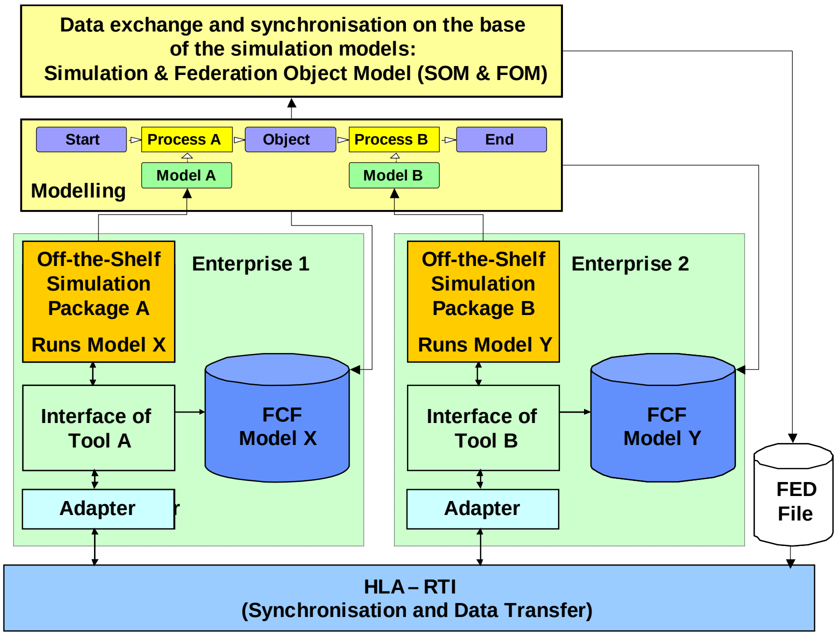

Data exchange and synchronisation on the base of the simulation models: Simulation & Federation Object Model (SOM & FOM)
Start
Process A
Object
Process B
End
Model A
Model B
Modelling
Off-the-Shelf Simulation Package A
Runs Model X
Enterprise 1
Off-the-Shelf Simulation Package B
Runs Model Y
Enterprise 2
Interface of Tool A
FCF Model X
Interface of Tool B
FCF Model Y
FED File
Adapter
Adapter
HLA – RTI
(Synchronisation and Data Transfer)

Figure: Holistic concept from the modelling approaches to the execution across different models located within different enterprises (see also DTG1.1).

The connection of different models requires a synchronisation of models independent from the technical realisation of the connection. This synchronisation enables the models to run in one whole scenario. This requires to add interface elements within the models and the definition of objects which have to be exchanged. Moreover, the overall monitoring needs additional information from the simulation models to measure a global view of the operating features. It collects and calculates statistical data generated by the simulation models. After the overall scenario model is created, the files (exchange data model) for the HLA (SOM, FOM → FED) have to be generated. At the same time the configuration files for each model within a Off-the-Shelf Simulation Packages are generated.

The knowledge described is concerned with the development of interoperability to improve the analysis and monitoring facilities across companies as well as to arrange different model providers together.

5. Example (optional)

An industrial case can be the following: Three simulation models (part production, assembly, testing) can be run separately. The simulation of the supply chain including the different interfaces needs the combination of these models. Each model is provided by a independent service provider (separate companies). The objectives are:

- Identifying of inconsistencies between the interfaces of the three models
- Evaluation of the whole supply chain instead of only checking separate models
- Realisation of an environment to integrate supplier simulation models
- Evaluation of different supply chain scenarios
- Detecting the influence of changes and interferences within the supply chain.

6. Remark

The required tasks cover the connection of simulation models, the monitoring of the whole scenario and the definition of different scenarios. However, some optional tools could be provided to simplify the creation of scenarios, the reuse of simulation models, the expandability of the monitoring and the influence in the scenario of the user at runtime such as:

- Configurable Visualisation and Monitoring Components,
- Simulation Manager,
- Scenario management.

7. Reference

- MISSION; 01.11.2003: http://www.ims-mission.de/.
- Kuhl, F.; Weatherly, R.; Dahmann, J.: Creating Computer Simulation Systems. An Introduction to the High Level Architecture, Prentice-Hall, 1999.
- IEEE 1516-2000 Standard for Modeling and Simulation (M&S) High Level Architecture (HLA).
- Mertins, K.; Rabe, M.; Jäkel, F.-W.: Neutral Template Libraries for Efficient Distributed Simulation within a Manufacturing System Engineering Platform. In: Winter Simulation Conference, Orlando (USA) 2000, pp. 1549-1557.
- Rabe, M.; Jäkel, F.-W.: Simulation for Globally Distributed Enterprises. In: 12th European Simulation Symposium (ESS), Hamburg 2000, pp. 322-327.
- Jäkel, F.-W.; Arroyo Pinedo, J.S.: Development of a Demonstrator for Modelling and Simulation of Global Distributed Enterprises. In: Mertins, K.; Rabe, M. (Eds.): The New Simulation in Production and Logistics. 9th ASIM Dedicated Confer-ence on Simulation in Production and Logistics, Berlin 2000, pp. 375-384.
- XMSF 01.11.2003: http://www.movesinstitute.org/xmsf/xmsf.html.
- HLA-CSPIF 01.11.2003: http://www.brunel.ac.uk/~csstsjt/HLA CSPIF/index.html.
- SISO 01.11.2003: http://www.sisostds.org/.

Annex 3

Process for analysing interoperability scenarios

In the context of TG 6 work however, the adaptive process of the scenarios will be walked through with a purpose to feed a number sub-processes of the MCR development process. This is illustrated in Figure A3.1- 3. The relationships are as follows:

- *MCR1: Scope Definition*: Scenarios should be concerned with interoperability of enterprise software applications. MCR1 therefore affects the selection of the cases or scenarios.
- *MCR2: Problem Analysis*: The scenarios should expose interoperability problems (as causes of) enterprise problems *(E problem)* that necessitate a development in which method chunks need to be assembled following a situational assessment. The scenarios should illustrate the problems that are seen as the justification for developing and operating a MCR.

  The *MCR problem* should be expressed in reference to a *Method Engineering (ME) decision frame* (objectives on the development process, performance indicators and action means).

  It should be understood that in many cases interoperability problems will not surface for the enterprise management, they only appear as the *enterprise problem* is analysed, when a *technology opportunity* is evaluated, or when the implementation of a *directive* is explored.[1] The enterprise problem is expressed in reference to an (*E*) *Enterprise decision frame.*
- *MCR3: Requirements.* By simulating the scenario development path (some of processes S1-S8) we will be in the position to articulate more fine-grain requirements for the MCR. The scenarios should be selected in such a way that a range of MCR services are addressed or applied. Performance values for the E development path (without MCR use) must be expressed w.r.t. to the ME decision frame.
- *MCR7: Constructing and Testing.* Following the operationalization of the MCR prototype the task group will be in the position to re-run the scenario development path with the use

[1] Whitten et al. (2004) define problem, opportunity and directive as follows. ***Problem***: an undesirable situation that prevents the organization from fully achieving its purpose, goals, and/or objectives. Usually, problems are manifest in production or logistic processes. ***Opportunity****:* a chance to improve the organization even in the absence of an identified problem. Often an opportunity comes from the availability of new technology. ***Directive***: a new requirement that is imposed by management, government, or some external influence. A directive may also come from science, government, or other governance entities (the societal drivers).

of the MCR prototype. Performance values for the MCR enabled E development path must show improvements when comparing with values without use of MCR.

- *MCR Operation & Maintenance:* Eventually most of the scenario analysis must fit into this phase. For the first couple of scenarios, the focus will be on:
  - Scenario 1: Scope definition
  - Scenario 2: Problem Analysis: expressing the E decision frame and the enterprise problem in reference to it.
  - Scenario 3: Requirements: assessing the enterprise situation; on the basis of the assessment, configure the methods for the achieving sub-processes.

These phases will provide the basis for the situational assessment that will feed the MCR prototype use, and on the basis of which the MCR will offer method chunks for assembling a path for the remaining phases S4-S8.

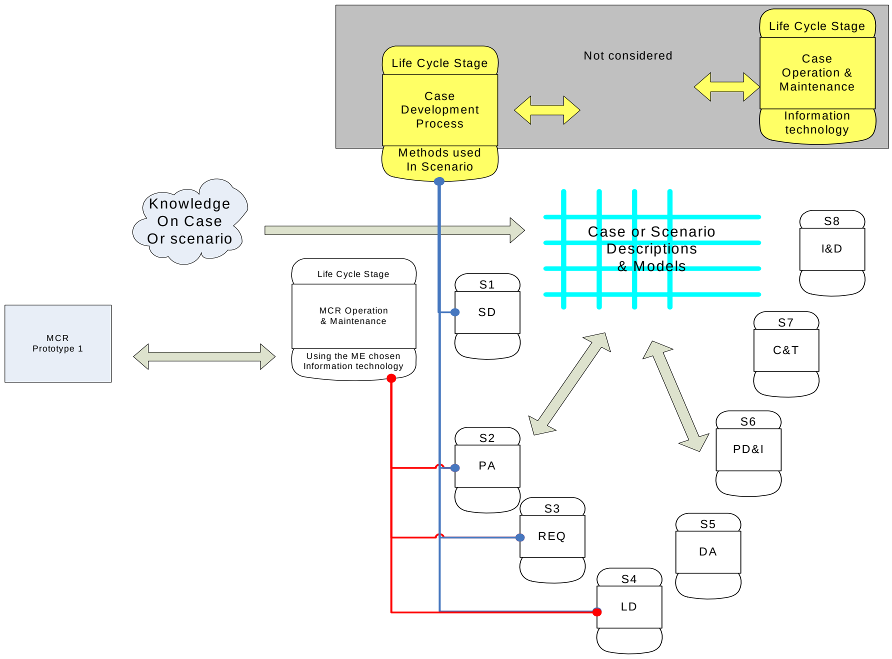


Figure A3.1: Scenarios and MCR Operation& Maintenance.

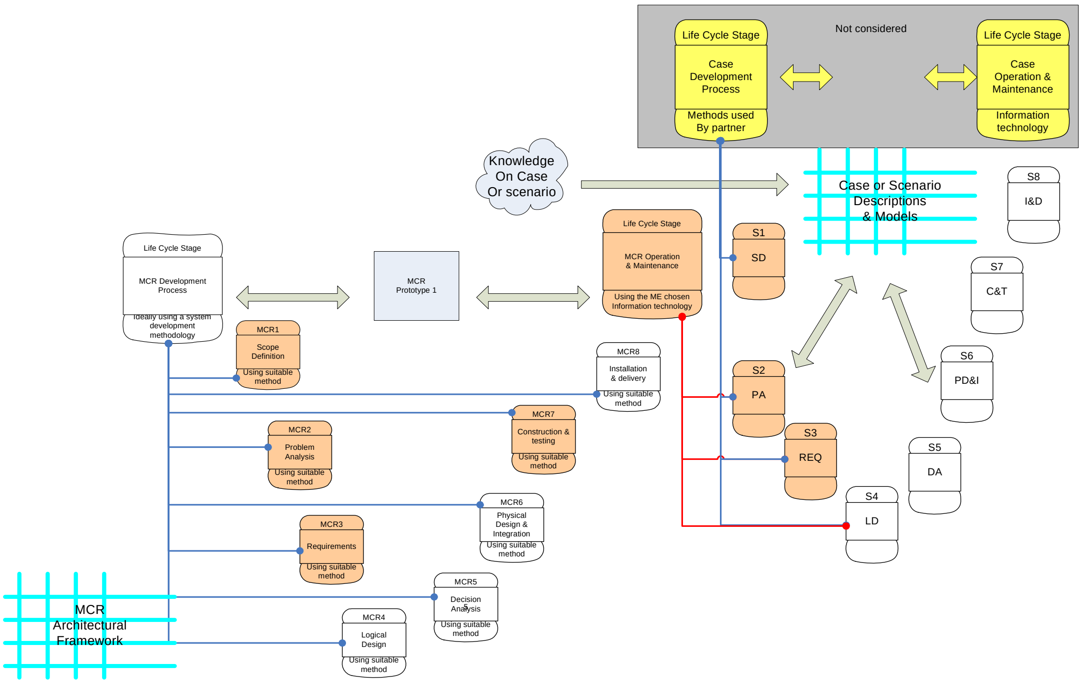


Figure A3.2: Where S-activities affect MCR-activities.

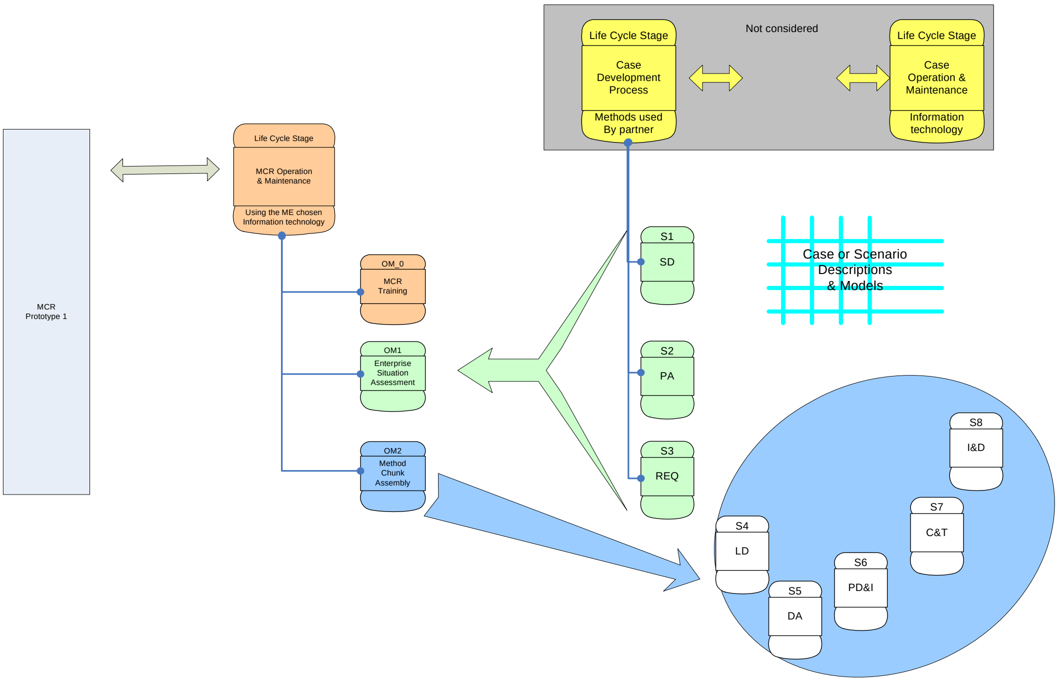


Figure A3.3: Enterprise Development Process enabled by Method Chunk Repository.