A Customizable Methodology for the Model-driven Engineering of Service-based System Landscapes

Michael Stollberg
SAP Research CEC Dresden
Chemnitzer Str. 48, 01187 Dresden, Germany
michael.stollberg@sap.com

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

Victor Shafran and Roman Magarshak
SAP Research Israel
15 Hatidhar St., 43665 Ra'anana, Israel
{victor.shafran, roman.magarshak}@sap.com

Abstract

Sophisticated engineering environments for modern business solutions need to combine several specialized techniques in order to meet the growing industrial requirements. Integrated and extensible technology platforms appear to be suitable for this, which however become complex and hence are difficult to use. This paper presents a tool-supported methodology framework for supporting engineers in the proper and beneficial usage of such complex technology environments by automatically created custom methodologies. We use and extend the Eclipse Process Framework (EPF) to define reusable methods and processes, and on top of this provide a tooling infrastructure for generating methodologically valid procedures for individual system development projects that contain detailed guidance only for the relevant engineering techniques. For illustration, we define concrete methods for an integrated model-driven engineering environment for service-based system landscapes, and evaluate the usage and business benefits of our framework for real-world industrial system engineering.

1 Introduction

In times of global markets, the competitiveness of companies in various industries is more and more determined by the efficiency of its business processes. In order to provide adequate IT support for this, sophisticated business technologies need to support the specification of efficient business processes, the integration of business-relevant systems, and the flexible adaptation of the processes along with the related technical implementations.

There is a wealth of technologies and research activities addressing this challenge. Most prominent from an industrial perspective is the establishment of service-orientation as a central design principle for enabling system integration and interoperability among legacy systems, and Model-Driven Engineering (MDE) for facilitating efficient and flexible system development by the successive modelling from high-level business perspective down to technical implementation models and automated code generation. However, an MDE infrastructure for service-oriented system design appears to be not sufficient because most industrial solutions require the integration with other business technologies and demand advanced engineering support.

A suitable approach for this is to provide an extensible MDE framework that integrates several business-relevant engineering techniques. This can serve as a generic development infrastructure for various industrial applications, supporting efficient system design in a structured and integrated manner. However, such comprehensive technology environments become very complex, and typically only a part of the provided engineering techniques is needed for developing a particular solution. In consequence, a sophisticated methodology is needed that – in addition to the tool-supported MDE infrastructure – guides system engineers in the proper and beneficial usage of the engineering framework. This should support the identification of the relevant techniques and the definition of the overall methodological procedure for a particular engineering project along with detailed guidance for the individual engineering tasks.

This paper presents a framework that addresses these requirements by enabling the customization of methodologies for complex MDE infrastructures. As the core constructs, we define methods that provide structured guidance for individual engineering tasks and partial processes that specify suitable engineering procedures by aggregating methods and defining the dependencies among them; the methodology content is organized by a reference matrix. We use and extend the Eclipse Process Framework (EPF) for implementing the methodology framework, and provide tool support for creating workable engineering procedures that are customized for particular system engineering projects. While the methodology framework is in general applicable to various integrated engineering environments, we illustrate it for an extended MDE infrastructure for service-based system engineering throughout the paper.

The paper is structured as follows: Section 2 introduces the extended MDE infrastructure for service-based system engineering. Section 3 specifies the methodology framework and its implementation in EPF, and Section 4 defines concrete methods for the extended MDE infrastructure and illustrates the tool-supported creation of customized methodologies for this. Section 5 evaluates the methodology within real-world industrial use cases, and Section 6 positions our approach within related work. Finally, Section 7 concludes the paper.

2 Extensible MDE Infrastructure for Service-based System Engineering

In order to motivate the need for customizable methodologies, the following introduces an MDE framework for service-based system landscapes that integrates several business-relevant engineering techniques and has been developed in the European research project SHAPE. Referring to [16] for further details, we here provide a brief overview that appears to be sufficient for the purpose of illustration throughout this paper.[1]

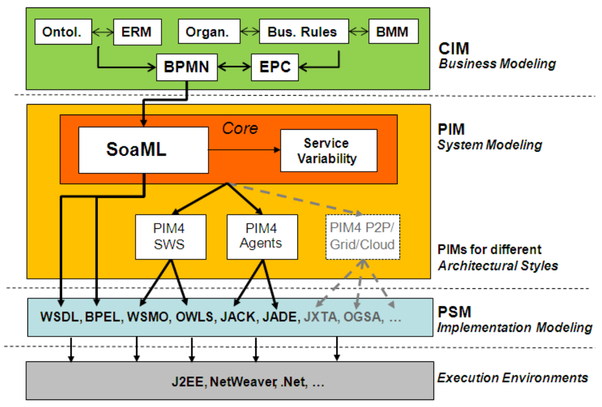


Figure 1. MDE for Service-based System Engineering – Overview

[1]SHAPE Project Website: `www.shape-project.eu`.

The aim is to provide an integrated infrastructure for the model-driven development of service-based system landscapes with support for various business-relevant technologies that have been identified in industrial use cases.

Figure 1 shows a high-level overview of the framework. It follows the OMG Model Driven Architecture (MDA) approach that distinguishes three levels for modelling IT systems with automated model transformations between them [11]. Starting at the top, on the level of computational independent models (CIM) several prominent techniques for modelling the business perspective are integrated. These are transformed into skeletons of platform-independent models on the next level (PIM) that describe the IT system architecture for supporting the execution of the business processes. Here, SoaML – a UML Profile and Metamodel standardized by OMG [12] – is used to describe the services and the overall system architecture. This is extended with various techniques that appear to be desirable for real-world system engineering. The existing framework integrates a service customization technique by creating personalized variants [15], semantic technologies for facilitating interoperability in heterogeneous environments (via PIM4SWS [4]), and agent technology for designing automated planning systems on top of service-based system infrastructures (via PIM4Agents [5]); this is extensible for other technology platforms like P2P, Grid, or Clouds [10]. Finally, the architecture models are transformed into platform-specific models (PSM) from which the implementation code for the respective execution environments is generated.

This provides an integrated development environment for the model-driven engineering of service-based systems with support for various business-relevant technologies. It however is very complex, and for most applications only a part of the available engineering techniques is needed. For example, the German steel manufacturer Saarstahl AG wants to integrate an automated planning system that manages the production line with the order and customer management systems; for this, service modelling with SoaML and the agent extension is needed. At the Norwegian oil & gas company Statoil, the service customization technique and the semantic technology extension are needed for integrating legacy systems that work on different industrial standards and expose a high internal complexity.

In order to support such diverse application scenarios, we have developed a methodology framework that allows system engineers to create customized methodologies that define the overall procedure for a particular system development project and merely contain the relevant engineering techniques as illustrated in Figure 2. Such a methodological support on top of the technological infrastructure appears to be necessary in order to facilitate the proper and beneficial usage of comprehensive engineering environments as the one introduced above. The remainder of the paper presents

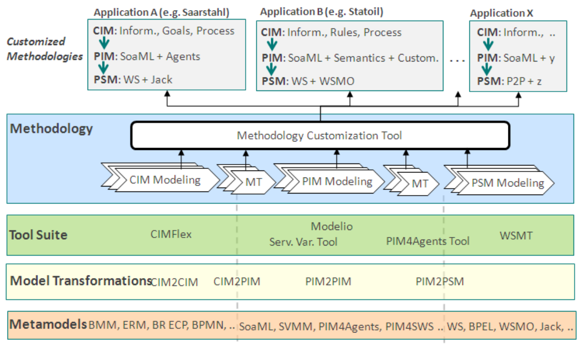


Figure 2. Methodology Overview & Purpose

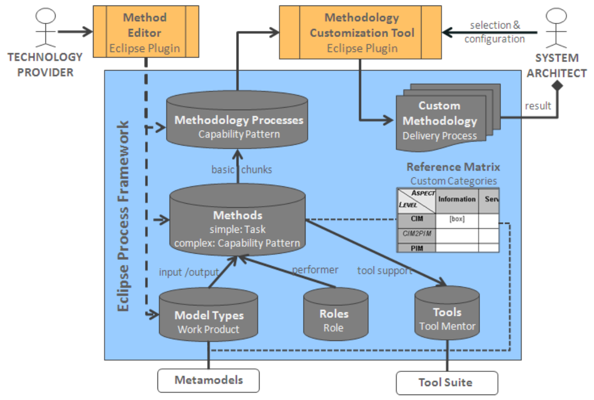


Figure 3. Methodology Framework

the technical realization of the methodology framework in detail, and we discuss its usage and benefits for the mentioned industrial use cases in Section 5.

3 Methodology Framework

This section presents the technical realization of the methodology framework. We specify the necessary elements and explain the implementation within the Eclipse Process Framework (EPF), an open-source infrastructure for defining software engineering processes. While the following focuses on the generic architecture of the methodology framework that can be applied to various MDA engineering frameworks, we will present concrete methodology content and illustrate the customization support for the model-driven engineering framework for service-based systems introduced above in the next section.

3.1 Element Specification

Figure 3 shows the overall structure of the methodology framework, indicating the EPF constructs used for the technical implementation as explained below in more detail. The first main element are *Methods* that provide detailed user guidance for creating a specific model. A method defines the necessary engineering steps in a structured manner, and is associated with the following elements: the *Model Types* as the resulting output and the models required for the engineering task, the *Roles* which perform the method, and the *Tools* to be used for this. We distinguish two kinds of methods, depending on the complexity of the necessary user guidance: while for automated model transformations simple instructions are sufficient, several engineering tasks require a more detailed procedural description – e.g. the creation of SoaML models for a complete system landscape. We further use a *Reference Matrix* as the underlying organization scheme of the methodology content: its vertical axis defines the MDA levels, and the horizontal axis defines the aspects relevant for service engineering (information, services, processes, rules, events, organization, goals, and non-functional aspects). The methods, processes, and the model types are categorized according to this matrix.

The second main element are *Methodology Processes*. These define methodologically workable procedures over conjoined methods for the separate phases of the intended overall engineering process, therewith providing the higher-level guidance for how the various engineering techniques can be used. The methodological logic is defined in terms of processes that define the order of the engineering tasks. Therein, specific methods can be declared to be mandatory or optional with respect to the intended usage of the engineering techniques as supported by the existing tools and model transformations. On this basis, a *Custom Methodology* can be created, which defines the overall engineering procedure for a particular system development project. The *Methodology Customization Tool* supports the system architect in choosing the appropriate methodology processes and selecting the relevant methods, and it ensures that the resulting process description is methodologically valid with respect to the logic defined in the methodology processes and the dependencies of the chosen methods. In addition, the *Method Editor* supports technology providers in the specification of valid methodology contents.

3.2 Implementation in EPF

We have implemented the methodology framework within the Eclipse Process Framework (EPF), an open-source technical infrastructure for defining customizable software engineering processes that provides a specification framework for methods and processes along with editing and content management facilities (see `www.eclipse.org/epf`). As the conceptual framework is similar, the mapping of our framework to the EPF constructs is straightforward as shown in Figure 4 (adapted from [6]).

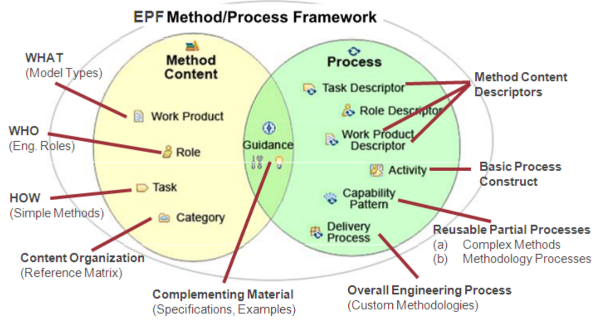


Figure 4. EPF Constructs and Usage

We define our models as work products, and specify the reference matrix in terms of EPF custom categories. A main difference is the differentiation of simple and complex methods. In EPF, detailed stepwise user guidance that is associated with work products and roles can only be defined as *Tasks*. These however do not support the definition of a procedural guidance that consists of several ordered engineering tasks. This is needed for our complex methods, which appear to be necessary in order to properly guide engineers through more complex engineering tasks, such as the modeling of a comprehensive system architecture in SoaML. We hence specify both complex methods and methodology processes as *Capability Patterns* that define processes over tasks, and distinguish them by additional custom categories in EPF. Our custom methodologies are presented as *Delivery Processes* that are defined by aggregating capability patterns and tasks in EPF.

The tool support for our methodology framework is implemented in form of Eclipse plugins on top of the open-source EPF implementation. These implement the necessary algorithms for ensuring the validity of newly created methodology content and custom methodologies, and guide the user stepwise via UIs in form of wizards. Unfortunately, the EPF release (version 1.5.0.2) that was available at the time of our implementation did not provide an API for the development of external tools. Hence, our prototypes are tightly connected to the EPF implementation.

4 Service Engineering Methods and Tool-Supported Methodology Customization

After having the defined the generic structure of the methodology framework, this section illustrates its instantiation and usage for the model-driven engineering of service-based systems. For this, the following first presents the concrete methods and methodology processes for the extended MDA framework for service-based system engineering introduced above, and then explains the tool support for creating customized methodologies in detail.

4.1 Methods and Methodology Processes

The following presents the actual methodology content for the extended MDA framework for the model-driven development of service-based systems introduced above in Section 2 (*cf.* Figure 1). The purpose is to provide a principle overview on the intended usage of the various integrated techniques and to illustrate the definition of reusable methods from which system architects can create custom methodologies for a particular service engineering project. As a concise overview that appears to be sufficient for illustration, we here present the methodology processes that specify workable procedures for using the available engineering techniques along with the existing tooling infrastructure on each of the distinct MDA levels; the complete methodology content is available online at: `www.shape-project.eu/download-area/SHAPE-Methodology_OnlineLibrary_final/`.

4.1.1 Integrated Business Modelling (CIM level).

As the most abstract perspective in the MDA approach, the CIM level is concerned with describing a company or an economic ecosystem from the business perspective independent of how this is supported by IT systems. For this, our engineering framework integrates various modelling techniques for business-relevant aspects and enables efficient modelling by providing integrated tooling support.

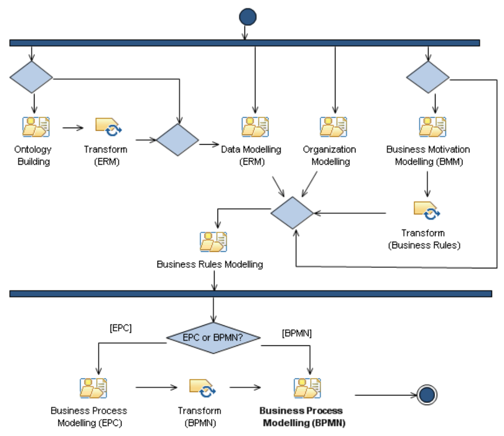


Figure 5. Business Modelling Methodology Process

Figure 5 shows the methodology process for this. It supports several entry points: the business analyst can commence with defining a common terminology in form of an ontology, or with modelling the data model and the organizational structure, or with the business motivation in BMM. On this basis, the constituting conditions and rules of busi-

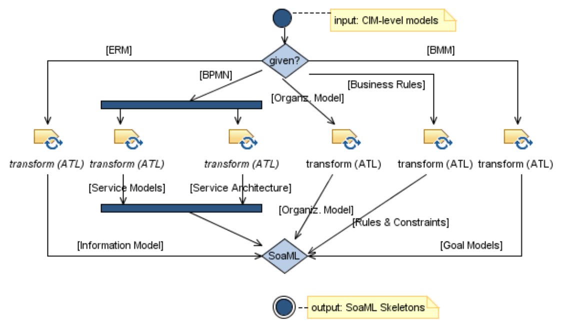


Figure 6. CIM2PIM Transformations

ness can be defined, which in turn are the basis for specifying efficient business processes modelled in EPC (Event Process Chains) or BPMN. Partial transformations support the sequential use of the various modelling techniques.

The design of an IT system that properly supports the business aims requires an appropriate reflection of the processes and requirements defined on the CIM level within the technical system models. Commonly referred to as the business-IT gap, the main challenge for this is to bridge the substantially different concepts between the business and the technical perspective. To support this, our methodology encompasses automated model transformations defined in ATL (see `www.eclipse.org/m2m/atl/`) that generate skeletons of PIM level models from the given CIM level models as shown in Figure 6. In our methodology, the main model types are BMPN for business process modelling on the CIM level and SoaML for service and architecture modelling on the PIM level. Hence, these are defined as the only mandatory activities in the methodology processes, indicated by bold labelling in the figures.

4.1.2 System Architecture Modelling (PIM level).

Subsequent to the business modelling on the CIM level, the PIM level is concerned with describing the components and architecture of the IT system that shall support the proper and efficient execution of the business processes. For this, our MDA framework supports the modelling of services and service-based architectures in SoaML as well as the optional usage of additional engineering techniques.

Figure 7 shows the methodology process for the system modelling on the PIM level. At first, the basic service descriptions and the overall system architecture is modelled in SoaML. This is the only mandatory activity, and the user can choose which of the other engineering techniques shall be used for a particular engineering project. Currently, the framework integrates the following extended service engineering techniques (see Section 2):

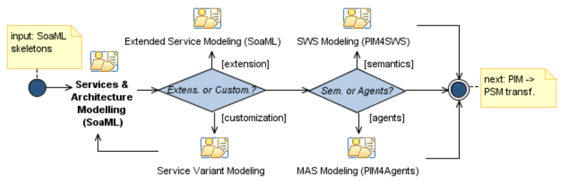


Figure 7. System Architecture Modelling Methodology Process

- Extended Service Modelling: behavioural and abstract functional descriptions in SoaML, serving as the basis for advanced service engineering techniques (e.g. composition, discovery, mediation)
- Service Variant Modelling: enable customization of services and simplified consumption by creating preconfigured and personalized variants (see [15])
- MAS Modelling: integrated modelling of multi-agent systems that work on top of a service-oriented system landscapes (see [5])
- Semantic Technologies Modelling: extend service and architecture models with semantic annotations to enable heterogeneity handling and automation of service consumption tasks (see [4]).

Note that Figure 7 only shows the high-level procedure for selecting the desired techniques: each activity of the methodology process is a complex method that consists of several engineering tasks. Moreover, there are various dependencies among the model types; these are defined via the input- and output work products of the individual methods.

4.1.3 Implementation Modelling (PSM level).

As the most detailed modelling perspective, the PSM level is concerned with the detailed implementation models for specific technology platforms from which actual code can be generated. Exploiting the advantages of the model-driven approach, the skeletons of the implementation models are generated by fully automated model transformations.

On the PSM level, we hence define simple methods that guide developers in refining the generated skeletons along with recommendations for the deployment on respective execution environments. Currently, our framework provides PIM2PSM transformations and PSM methods for the following technology platforms:

- Web Services & Processes: WSDL, XSD, BPEL
- Multi-Agent Systems: JACK, JADE
- Semantic Web Services: WSMO, WSMX, IRS-3.

4.2 Methodology Customization Tool

We now turn towards the creation of custom methodologies for individual system engineering projects. As already outlined above, a custom methodology defines the overall engineering procedure for a specific system development project, merely containing the relevant techniques and methods out of the available ones. To support system architects in the creation of custom methodologies, we provide the Methodology Customization Tool that guides the identification and selection of the relevant engineering methods and generates a valid overall engineering procedure with respect to the procedural constraints and dependencies that are defined within the methodology content.

The Methodology Customization Tool is implemented as an EPF plugin and works on reusable methodology content as e.g. the methods and processes explained above. As we shall illustrate for concrete use cases below in Section 5, the workflow for creating a custom methodology is as follows:

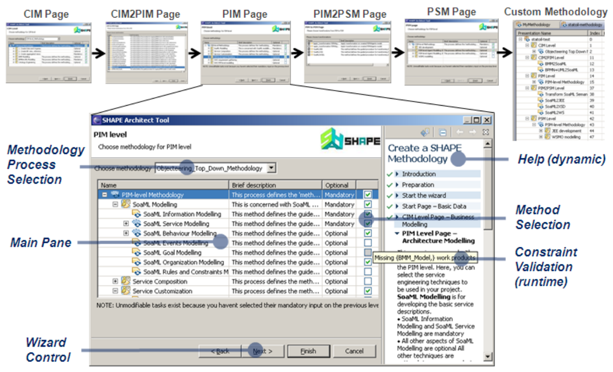
1. **Identification of relevant Engineering Techniques** by analyzing requirements and consulting the methodology & tool documentation

2. Tool-supported **Custom Methodology Creation**:

   - Configuration of EPF infrastructure for project
   - Wizard for selecting required engineering methods with including runtime validation of user choices (see below)
   - Generation of valid custom methodology as EPF Delivery Process

3. **Refinement** of custom methodology via EPF editing facilities (optional)

4. **Publication** of final custom methodology as a website (optional).



PIM level
Choose methodology for PIM level
Choose methodology: Objecteering_Top_Down_Methodology

Name	Brief description	Optional	
PIM-level Methodology	This process defines the 'meth...	Mandatory	✓
SoaML Modelling	This is concerned with SoaML ...	Mandatory	✓
SoaML Information Modelling	This method defines the guide...	Mandatory	✓
SoaML Service Modelling	This method defines the guide...	Mandatory	✓
SoaML Behaviour Modelling	This method defines the guide...	Optional	✓
SoaML Events Modelling	This method defines the guide...	Optional	
SoaML Goal Modelling	This method defines the guide...	Optional	
SoaML Organization Modelling	This method defines the guide...	Optional	✓
SoaML Rules and Constraints M	This method defines the guide...	Optional	
Service Composition	This process defines the meth...	Optional	
Service Customization	This process defines the meth...	Optional	✓

NOTE: Unmodifiable tasks exist because you havent selected their mandatory input on the previous leve

Create a SHAPE Methodology
- Introduction
- Preparation
- Start the wizard
- Start Page – Basic Data
- CIM Level Page – Business Modelling
- PIM Level Page – Architecture Modelling

Missing {BMM_Model,} work products

the PIM level. Here, you can select the service engineering techniques to be used in your project. **SoaML Modelling** is for developing the basic service descriptions.
• SoaML Information Modelling and SoaML Service Modelling are mandatory
• All other aspects of SoaML Modelling are optional All other techniques are

Presentation Name	Index
statoil-test	0
CIM Level	1
Objecteering Top Down f	2
CIM2PIM Level	11
BMM2SoaML	12
BPMN+UML2SoaML	13
PIM Level	14
PIM-level Methodology	15
PIM2PSM Level	37
Transform SoaML Seman	38
SoaML2JEE	39
SoaML2XSD	40
SoaML2WS	41
PSM Level	42
PSM-level Methodology	43
JEE development	44
WSMO modeling	47

Figure 8. Methodology Customization Tool

Figure 8 illustrates the tool support for defining custom methodologies (*cf.* step 2). The wizard guides the system architect stepwise from the CIM level down to the PSM level. On each page, the user can choose one of the predefined methodology processes and select the specific engineering methods that are required for the project. The tool ensures the methodological validity with respect to the constraints and dependencies that are defined within the methods and methodologies, and finally generates an EPF delivery process that contains the selected methods and defines a coherent overall engineering procedure for the project. The system architect can optionally refine this – e.g. add specific instructions for the application scenario – and publish the custom methodology as a website that provides the central guidance for all roles involved in the engineering project.

The tool implements the following central algorithms for supporting the creation of custom methodologies and ensuring their methodological validity:

- *Selection of Methods*: all mandatory methods are activated by default; on the transformation levels (CIM2PIM and PIM2PSM) the methods are activated for which the required inputs result from the methods chosen on the preceding level; the user can change this when the suggested procedure shall not be followed

- *Runtime Validation*: user selections are validated at runtime with respect to the dependencies among the methods, e.g. methods can not be selected when a mandatory input does not exists within the preceding user selections; constraint violations along with resolution options are provided in the user interface

- *Generation of Custom Methodology*: generation of an EPF delivery process that contains the selected methods and defines a coherent overall engineering procedure, including a final validation of method dependencies and the refinement of the procedures.

With this, we provide an easy-to-use tool that allows system architects to adapt comprehensive engineering technologies to their individual needs. To our knowledge, comparable tools for EPF do not exist at this point in time (see Section 6 for a more detailed discussion).

5 Evaluation

In order to evaluate the usability and business relevance of the methodology framework, we have applied it to the two industrial use cases already mentioned above. In addition, we conducted an end-user survey where several technology experts and system engineers tested and assessed the methodology infrastructure, content, and tool support. The following summarizes the central findings.

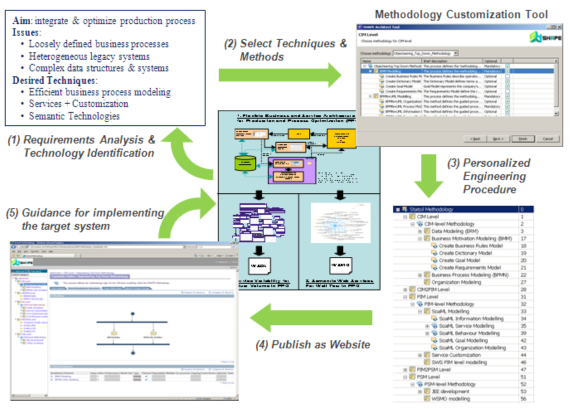

Aim: integrate & optimize production process
Issues:
• Loosely defined business processes
• Heterogeneous legacy systems
• Complex data structures & systems
Desired Techniques:
• Efficient business process modeling
• Services + Customization
• Semantic Technologies
(1) Requirements Analysis & Technology Identification
(2) Select Techniques & Methods
Methodology Customization Tool
(3) Personalized Engineering Procedure
(4) Publish as Website
(5) Guidance for implementing the target system
Statoil Methodology
CIM Level
CIM-level Methodology
Data Modeling (ERM)
Business Motivation Modeling (BMM)
Create Business Rules Model
Create Dictionary Model
Create Goal Model
Create Requirements Model
Business Process Modeling (BPMN)
Organization Modeling
CIM2PIM Level
PIM Level
PIM-level Methodology
SoaML Modelling
SoaML Information Modelling
SoaML Service Modelling
SoaML Behaviour Modelling
SoaML Goal Modelling
SoaML Organization Modelling
Service Customization
SWS PIM level modelling
PIM2PSM Level
PSM Level
PSM-level Methodology
JEE development
WSMO modelling

Figure 9. Usage of Methodology Framework

The first use case is concerned with optimizing the processes for reservoir and production management at the Norwegian oil & gas company Statoil. The aim is to provide an integrated solution for optimizing the reservoir performance, intelligent wells and production processes among the numerous offshore production sites. The processes are currently performed by production experts supported by an IT landscape which consists of various isolated and heterogeneous legacy systems of a considerably high complexity. Hence, the following techniques appear to be suitable for enhancing the IT-supported business efficiency: techniques for modelling efficient business processes, services for integrating the legacy systems, semantic technologies for handling the heterogeneous resources, and customization techniques for enabling the effective usage of the complex systems in different application contexts.

Figure 9 illustrates the usage of the methodology framework as explained above. Referring to [7] for details, the following experiences have been reported. Comprehensive engineering environments as the one presented here are highly desirable for designing value-added business solutions in an integrated manner. The concept of customized methodologies is considered to be highly valuable for enabling system engineers to effectively apply the engineering techniques in a specific context. The tool-supported methodology framework is reported to be well usable by knowledgeable system architects, and the detailed guided procedures along with illustrative examples are suitable for guiding also non-technical experts in performing the engineering tasks. Particularly useful in this scenario are the automated BPMN->SoaML transformation for bridging the gap between the production and the IT experts, and the web-publication facility for enabling the collaborative system design among locally distributed engineers.

Similar benefits are reported from the second use case at the German steel manufacturer Saarstahl AG. Here, the central challenge is to integrate an agent-based planning system for optimizing the production line capacity with the previously isolated order- and customer management systems. While the integration is achieved by providing the legacy systems as services, the main merit of the integrated MDA framework is that the interactions between the services and agents can be specified on the architecture level (PIM) in such detail that the system implementation can be generated and maintained with minimal effort [5]. For this, the detailed methodological guidance is essential to assist the system engineers in designing the system. Besides, the customized methodology is useful for explaining the system benefits to business stakeholders as well as for coordinating the system development among the involved departments.

In our end-user survey, most participants consider integrated and extensible frameworks for the model-driven engineering of service-based systems to be highly desirable, and substantiate the need for customizable methodological support. The concept of custom methodologies as well as the overall design of the tool support and infrastructure for creating and using them is mostly regarded as a suitable and useful technical solution for this, and we have implemented various improvements on the tool usability on the basis of the user feedback. However, we consider the Methodology Customization Tool as a research prototype that requires further development to reach industrial strength.

6 Related Work

Methodological guidance is commonly considered as an essential part of Software Engineering, and there is a wealth of approaches and tools for this. Our framework belongs to the category of *engineering methods* that provide guidance for using a particular technology, which are in general complementary to *development methodologies* like SCRUM or more classical techniques that are concerned with planning and controlling the actual development process [14].

There are numerous methods for technologies related to service-based system engineering; [3] provides a comprehensive overview on the most prominent ones. However, most of the existing methodological frameworks are dedicated to a specific technology and define static procedures that can hardly be combined into a comprehensive methodology for integrated engineering frameworks. To overcome this, our approach follows the idea of Situational Method Engineering (see [9] and [2]) where reusable method chunks are assembled into customized engineering methods for particular application scenarios.

Recent approaches – mostly notably around the Eclipse Process Framework (EPF) – aim at providing a generic infrastructure for customizable software engineering methodologies. Notably, OpenUP provides an open-source implementation of the Unified Process – a generic framework for iterative software engineering processes [8] – within EPF, and the IBM Rational Method Composer (`www-01.ibm.com/software/awdtools/rmc`) provides a commercial tool with IBM's own SoaML-based Service-Oriented Modeling and Architecture (SOMA) methodology [1]. Although the infrastructures are extensible with methods for specific engineering techniques, the customization of the generic processes for a specific scenario is left to manual inspection. For this, our tool-supported framework provides a complementary infrastructure for selecting the desired methods and automatically creating methodologically valid engineering procedures.

A specific feature of our framework is that it can be reused to define customizable methodologies for other engineering technologies. The initial investigation for the ISE Workbench – an integrated service engineering environment [13] – has revealed that merely methods and methodology processes for the respective techniques need to be defined; the technical infrastructure as well as the Methodology Customization Tool can be used without any changes.

7 Conclusions

This paper has presented a tool-supported framework for customizing methodologies for comprehensive model-driven service engineering environments. Such methodological support appears to be highly desirable for integrated engineering environments that are employed for developing modern business solutions.

Following the Situational Method Engineering approach, we have defined a framework and the necessary constructs for enabling the customization of reusable method chunks. The methods and processes are defined within the Eclipse Process Framework (EPF). On top of this, we provide new tool support for the automated creation workable engineering procedures that merely contain the techniques relevant for individual system development projects.

While the presented methodology framework and tool support is in general applicable to any model-driven engineering technology, we have presented an instantiation for an extended MDA framework for service-based system engineering. For this, we have defined an integrated set of reuable methods and methodology processes around BPMN and SoaML, and discussed its business relevance within two industrial use cases. For the future, we plan to apply the framework to other engineering technologies and to align it with the ongoing EPF developments.

Acknowledgements. This paper is mainly based on works supported by EU funding under the SHAPE project (FP7 - 216408). The authors would like to dedicate special thanks to Dmytro Panfilenko, Christian Hahn, Davide Cerri, Marisa Escalante, and Arne-Jørgen Berre for the seminal collaboration and fruitful discussions.

References

[1] J. Amsden. Modeling with SoaML. Technical article, IBM, 2010. Online: `www.ibm.com/developerworks/rational/library/09/modelingwithsoaml-1/index.html`.

[2] M. Brinkkemper S., Saeki and F. Harmsen. Assembly Techniques for Method Engineering. In *10th Conference on Advanced Information Systems Engineering, CAiSE'98, LNCS 1413*, pages pp. 381–400. Springer, 1998.

[3] B. Elvesæter and S. G. Johnsen (eds.). Model-driven Methodology and Architecture Specification. Deliverable D2.1, SHAPE Project, 2009.

[4] D. Fensel, H. Lausen, A. Polleres, J. de Bruijn, M. Stollberg, D. Roman, and J. Domigue. *Enabling Semantic Web Services. The Web Service Modeling Ontology*. Springer, 2006.

[5] C. Hahn, C. Madrigal-Mora and K. Fischer. Interoperability through a Platform-Independent Model for Agents. In K. M. J. P. Müller and M. Zelm, editors, *Enterprise Interoperability II - New Challenges and Approaches*. Springer, 2007.

[6] P. Haumer. Eclipse Process Framework Composer. Part 1: Key Concepts. EPF Documentation, 2007. Online: `www.eclipse.org/epf/composer_architecture`.

[7] S. Jacobi (ed.). Case Study Execution and Validation. Deliverable D1.3, SHAPE Project, 2009.

[8] P. Kroll and B. MacIsaac. *Agility and Discipline Made Easy: Practices from OpenUP and RUP*. Addison-Wesley, 2006.

[9] K. Kumar and R. Welke. Method Engineering: A Proposal for Situation-specific Methodology Construction. In Cotterman and Senn, editors, *In Systems Analysis and Design : A Research Agenda*, pages pp. 257–268. Wiley, 1992.

[10] P. K. McKinley, F. A. Samimi, J. K. Shapiro, and C. Tang. Service clouds: A distributed infrastructure for constructing autonomic communication services. In *Proc. of the 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing (DASC '06)*, pages 341–348, Washington, DC, USA, 2006. IEEE Computer Society.

[11] S. J. Mellor, K. Scott, A. Uhl, and D. Weise. *MDA Distilled – Principles of Model-Driven Architecture*. Object Technology Series. Addison-Wesley, 2004.

[12] OMG. Service oriented architecture Modeling Language (SoaML) – Specification for the UML Profile and Metamodel for Services (UPMS). FTF Beta 1, 2009.

[13] G. Scheithauer, K. Voigt, V. Bicer, M. Heinrich, A. Strunk, and M. Winkler. Integrated Service Engineering Workbench: Service Engineering for Digital Ecosystems. In *Proc. of the International ACM Conference on Management of Emergent Digital Ecosystems (MEDES), 27-30 October 2009, Lyon (France)*, 2009.

[14] I. Somerville. *Software Engineering*. Addison Wesley, 8. edition, 2006.

[15] M. Stollberg and M. Muth. Service Customization by Variability Modeling. In *Proc. of the 5th International Workshop on Engineering Service-Oriented Applications (WESOA09), Stockholm, Sweden*, 2009.

[16] M. Stollberg (ed.). SHAPE Whitepaper. Technical Report, Version 3.2, SHAPE Project, 2010. Online: `www.shape-project.eu/download`.