Aligning Business and IT Models in Service-Oriented Architectures using BPMN and SoaML

Brian Elvesæter
SINTEF ICT
P. O. Box 124 Blindern
N-0314 Oslo, Norway
+47 22 06 76 74
brian.elvesater@sintef.no

Dima Panfilenko
DFKI IWi
Stuhlsatzenhausweg 3, Campus D3.2
D-66123 Saarbruecken, Germany
+49 681 85775 7777
dima.panfilenko@dfki.de

Sven Jacobi & Christian Hahn
Saarstahl
Bismarckstraße 57-59
D-66333 Voelklingen. Germany
+49 6898 10 3476
{sven.jacobi | christian.hahn}
@saarstahl.com

ABSTRACT

In this paper, we introduce the new Service oriented architecture Modeling Language (SoaML) and describe how the language can be used to align business models and IT models. In particular we provide a mapping specification from BPMN models to SoaML models.

Categories and Subject Descriptors

D.2.11 [**Software Architectures**]: Service-oriented architectures. D.2.12 [**Software Engineering**]: Interoperability.

General Terms

Design, Standardization, Languages, Theory.

Keywords

Business modelling, service modelling, business and IT alignment, BPMN, SoaML.

1. INTRODUCTION

There is an industrial interest in ensuring a good connection and mapping between business models as expressed in enterprise architectures and IT models as expressed in technical system architectures, which are commonly realised as service oriented architectures (SOAs). The increasing popularity of the SOA paradigm relies on its closeness to business models, in particular business processes. The concepts of SOA apply both to business architectures as well as system architectures. From a business perspective the SOA describes the business-critical processes, contracts, information and capabilities of the enterprise. From an IT perspective the SOA describes the software components, their service interfaces and how these components can be coupled to form a technical system architecture that supports the business requirements of the enterprise.

Although SOA concepts, business models and service technologies has been a hot topic the last few years, the alignment of business and IT models still remain a challenge. Furthermore, although modelling is now an integrated part of software engineering approaches, standardised modelling languages to support SOA has been lacking. SHAPE (Semantically-enabled Heterogeneous Service Architecture and Platforms Engineering) (ICT-2007-216408) (http://www.shape-project.eu/) is a European Research Project under the 7th Framework Programme that has developed an infrastructure for model-driven engineering (MDE) for SOA with support for various technology platforms [1]. The SHAPE technologies resolve around the new Service oriented architecture Modeling Language (SoaML) specification [2] from the Object Management Group (OMG). SoaML aims at providing a common modelling language to business and system architects. In the SHAPE project we have defined an MDE approach to SOA that incorporates the use of business modelling formalisms such as BPMN and provide mappings to SoaML to help the business and system stakeholders to align their business requirements and IT system implementations.

This paper is structured as follows: In Section 2 we give an overview of the SoaML language. Section 3 describes our requirements, mapping rules and tool support for the business and IT alignment between BPMN and SoaML. In Section 4 we present an illustrative example taken from one of the industrial use cases in the SHAPE project. Section 5 discusses our results and findings. Finally, Section 6 concludes this paper.

2. SoaML

The Service oriented architecture Modeling Language (SoaML) specification [3] defines a UML profile and a metamodel for the design of services within a service-oriented architecture. The goals of SoaML are to support the activities of service modelling and design and to fit into an overall model-driven development approach. The SoaML profile defines extensions to UML to support the range of modelling requirements for service-oriented architectures, including the specification of systems of services, the specification of individual service interfaces, and the specification of service implementations. This is done in such a way as to support the automatic generation of derived artefacts following an MDA based approach.

According to the specification, SoaML has been designed to support both an IT and business perspective on SOA. Our

MDI2010, October 5, 2010, Oslo, Norway.

experiences with the SoaML language, in the context of tool and method implementation in the industrial use case, have suggested that a clearer separation of the business-level and IT-level concepts are needed. In the context of SHAPE we have made these levels more explicit. Figure 1 illustrates the separation.

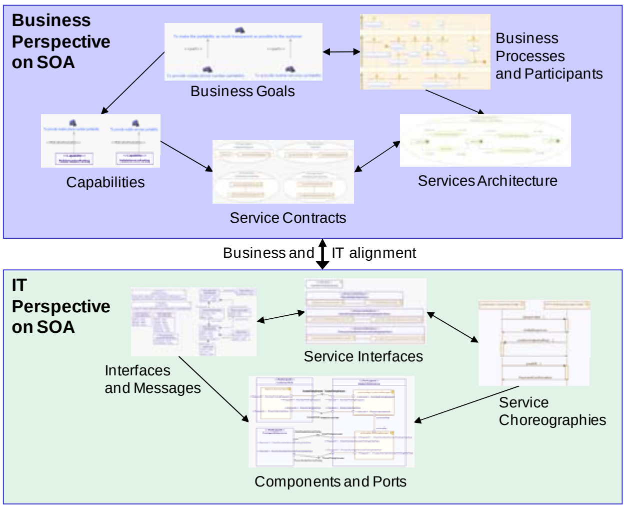


Figure 1. Business and IT concepts of SoaML

In the business perspective on SOA we suggest to integrate the use of the SoaML language with the Business Motivation Model (BMM) language [4] to define business motivation models and the Business Process Model and Notation (BPMN) language [5] to define business processes. Motivation models and business processes are important aspects to be included when modelling the business perspective on SOA. The SoaML specification defines relationships to BMM and the BMM specification defines relationships to BPMN which allows for this integration of languages. In this paper we focus on the relation between BPMN and SoaML in order to align process models from BPMN with service models for SOA. The language constructs from SoaML that are most suitable at the business level are *participant*, *services architecture*, *service contracts* and *capability* (see Figure 2).

Participants are used to define the service providers and consumers in a system. A participant may play the role of service provider, consumer or both.

Services architectures are used to define how a set of participants works together for some purpose by providing and using services. A services architecture describes how participants work together by providing and using services expressed as service contracts.

Service contracts are used to describe interaction patterns between service entities. A service contract is used to model an agreement between two or more parties. Each service role in a service contract has an interface that usually represents a *provider* or a *consumer*.

Capabilities represent an abstraction of the ability to affect change. Capabilities identify or specify a cohesive set of functions or resources that a service provided by one or more participants might offer. Capabilities can be used by themselves or in conjunction with participants to represent general functionality or abilities that a participant must have.

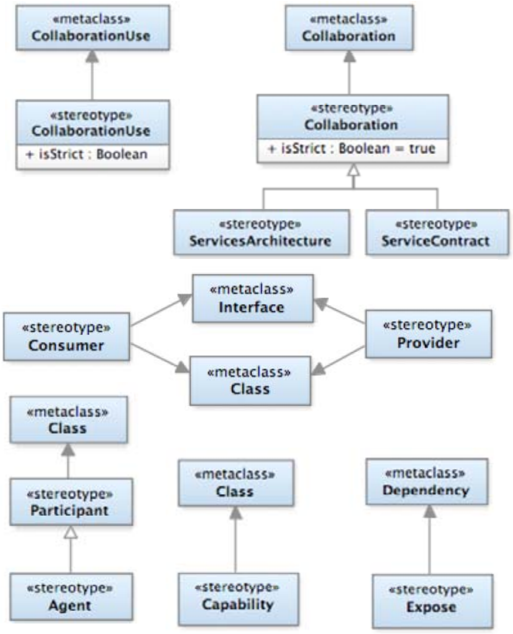


Figure 2. UML extensions for business concepts

The language constructs from SoaML that are most suitable at the IT level are *service interface* and its behaviour (i.e. service choreography), *interface, message type, components* (i.e. participants) and *service* and *request ports* (see Figure 3).

Service interfaces are used to describe the operations provided and required to complete the functionality of a service. A service interface can be used as the protocol for a service port or a request port.

Service data are used to describe service messages and message attachments. The *message type* is used to specify the information exchanged between service consumers and providers. An *attachment* is a part of a message that is attached to rather than contained in the message.

It should be noted that some of the language constructs defined in SoaML fit on both the business and IT level. In particular this applies to *participants* that are used to define the service providers and consumers in a system. At the business level the participants typically represent business organization units or roles, whereas on the IT level the participants typically represent IT systems or software components. When a participant acts as a provider it contains *service ports*, and when a participant acts as a consumer it contains *request ports*.

SoaML is agnostic to the choice of modelling formalisms to define behaviour. The specification states than any UML behavioural constructs can be used to describe behaviour such as service choreographies, but also other formalisms such as BPMN can be used.

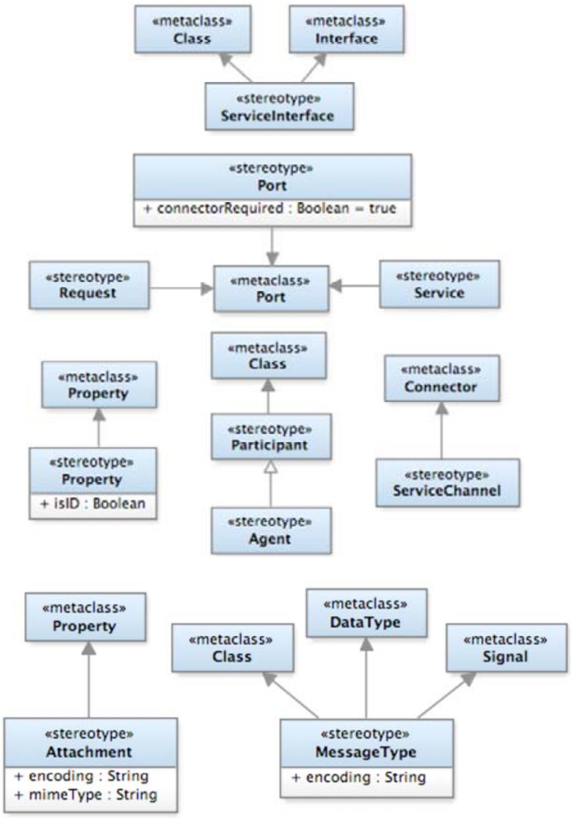


Figure 3: UML extensions for IT concepts

3. ALIGNING THE BUSINESS AND IT PERSPECTIVES ON SOA

3.1 Requirements

For the support of the different roles in a collaborative modelling project one can think of appropriate modelling formalisms. In the aligning of the business and IT perspectives there are obviously at least two roles that can be considered – business architect and system architect. They both are experts in their area but are not necessarily using the same notations for representing the same concepts, For that reason the two formalisms are described in the following that can be used by these respective users for modelling.

Business users may use a business process modelling formalism such as Event-driven Process Chain (EPC) [6] to represent their workflows. Process chains describe the sequencing and interaction between data, process steps, IT systems, organisational structure and products. An EPC always starts and ends with events, which define the state or condition under which a process starts and the state under which it ends. An event may initiate multiple functions at the same time; similarly, a function may result in multiple events. To represent these branches and processing loops in an EPC, a connector (or rule) is used. However, instead of acting simply as graphical connections, the connectors also define the logical links between objects, such as “and” or “either/or.” EPCs are typically used at the higher levels of the process hierarchy. If more technical details of business processes need to be described, other methods, such as BPMN, UML, or BPEL, are used instead of EPCs. The reference models provided by SAP are also defined using EPC methodology. EPCs offer a variety of ways to analyse processes and identify both quantitative and qualitative improvement options.

The Business Process Management Initiative (BPMI) (http://www.bpmi.org/) developed an initial standard called Business Process Modelling Notation (BPMN) that was adopted by the OMG and renamed to Business Process Model and Notation (BPMN) [5]. The primary goal of BPMN is to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes. Thus, BPMN creates a standardised bridge for the gap between the business process design and process implementation. Another goal, but no less important, is to ensure that XML languages designed for the execution of business processes, such as BPEL4WS (Business Process Execution Language for Web Services), can be visualised with a business-oriented notation. Furthermore you have the possibility to create organisational units. With pools and lanes you can manage your organisation view of the process. Another aspect is that you are able to communicate between pools and lanes.

In general the BPMN and SoaML models can be seen as different architectural viewpoints on the enterprise model, and coupled to the enterprise and information, and computational viewpoints respectively from the Reference Model for Open Distributed Processing (RM-ODP) [7-10]. Indeed, BPMN is focused rather on the enterprise processes and information, whereas SoaML primarily describes the structure of the service architecture. The models we create with the BPMN and SoaML standards could be seen as architectural viewpoints according to IEEE 1471 [11], which suggests a viewpoint-based modelling approach for supporting different stakeholders in the system development process.

3.2 BPMN to SoaML Mapping Rules

In this section the mapping rules for the model transformation between BPMN and SoaML are presented. The challenge here is in transforming BPMN models to SoaML in order to generate the appropriate system relevant constructs for SoaML according to the generic business context on the computation independent model (CIM) level. The tool support for that is implemented within CIMFlexMT (see Section 3.3), which supports in its initial version the model-to-model transformation by making use of the Atlas Transformation Language (ATL) [12]. First the simple one-to-one rules are presented and then patterns for recognizing the SoaML service contracts are introduced.

Mapping Rule 0: Process to Services Architecture

A *services architecture* has components at two levels of granularity: The community services architecture is a ”top level” view of how independent participants work together for some purpose. The services architecture of a community does not assume or require any one controlling entity or process. A participant may also have a participant services architecture, which specifies how parts of that participant (e.g., departments within an organization) work together to provide the services of the owning participant. Participants that realize this specification must adhere to the architecture it specifies.

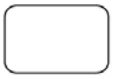
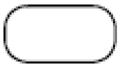
The *services architecture* is aligned with the business process, and the *participants* and *service contracts* can be derived from the pools or lanes and activities in the business processes respectively following these guidelines:

- Identify public and collaborative business processes that involve interactions and potential usage of software services between different business organizations. These processes are candidates for public community-level services architectures in SoaML that describe the service contracts between the business organizations.
- Identify private business processes for the business entities under your ownership control that are involved in the services architecture under consideration. These processes are candidates for private participant-level services architectures in SoaML that describe the service contracts between the internal organizational roles or units within the business organization.

Mapping Rule 1: Task to UML Action

A *task* describes an activity that is possibly providing a useful output that could be consumed by the participants of the process. It can be then mostly closely assigned to an *action* construct in UML as it gives the abstract interface for the job done and at the same time does not give further specification of the workflow implementing this task. In the CIM manufacturing example it means all three Tasks “Prepare Order”, “Purchase” and “Receive Order” are mapped to *actions*. Table 1 illustrates the mapping of the notation.

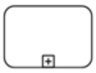
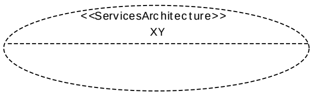
Table 1. Task to UML Action

	BPMN	SoaML
Construct	Task	Action
Notation		

Mapping Rule 2: Sub-Process to Services Architecture

A *sub-process* represents a more complex process than a simple task, but still can be seen as a whole. It can be assigned to a lower-level, e.g. *participant-level services architecture* that details the roles and tasks of the sub-process. It should be mentioned, though, that this *services architecture* is not necessarily the bottom level and can be subdivided further (through roles). Table 2 illustrates the mapping of the notation.

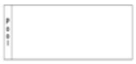
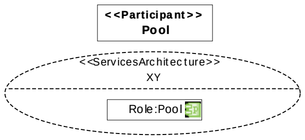
Table 2. Sub-Process to Services Architecture

	BPMN	SoaML
Construct	Sub-process	Services Architecture
Notation		

Mapping Rule 3: Pool to Participant (Community-level)

A *pool* in BPMN stands for a business entity or a participant of a process, on the one hand. It also can be structured with respect to further participants of the process, thus creating a participants’ hierarchy. These two points together to map the pool onto a role in a *community-level services architecture* that has a *participant* type matching the pool. Table 3 illustrates the mapping of the notation.

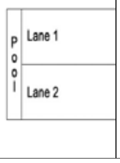
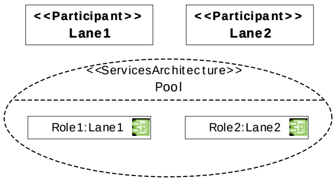
Table 3. Pool to Participant (Community-level)

	BPMN	SoaML
Construct	Pool	Participant Role in a Community-level Services Architecture
Notation		 <<Participant>> Pool <<ServicesArchitecture>> XY Role:Pool

Mapping Rule 4: Lane to Participant (Participant-level)

A *lane* represents a participant or a department in BPMN and is situated in a pool, thus showing the two-tier hierarchy. In order to show the possibility for further subdivision (which is also ongoing in the current BPMN2 proposals), the lane is mapped to a role in a *participant-level services architecture* that has a *participant* type matching the lane. The participant-level services architecture must adhere to the community-level services architecture for which the corresponding pool participants (see rule 2) belongs. Table 4 illustrates the mapping of the notation.

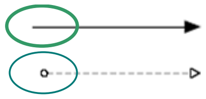
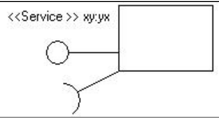
Table 4. Lane to Participant

	BPMN	SoaML
Construct	Lane	Participant
Notation		 <<Participant>> Lane1 <<Participant>> Lane2 <<ServicesArchitecture>> Pool Role1:Lane1 Role2:Lane2

Mapping Rule 5: Message “Begin” to Service

The *beginning point* of each and every message in BPMN has the following semantics – it should be the starting end of the data channel between two participants or pools. This exact meaning also has the *service port* in SoaML, which finds its accordance in this mapping point. The participants in SoaML are using this construct in order to provide services for other participants in the modelled architecture. Table 5 illustrates the mapping of the notation.

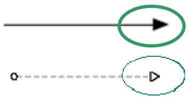
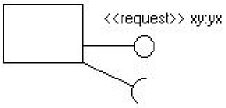
Table 5. Message "Begin" to Service

	BPMN	SoaML
Construct	Message "Begin"	Service
Notation		

Mapping Rule 6: Message "End" to Request

The *ending point* of each and every message in BPMN has the semantics that looks very alike with the message beginning point, but is situated on the other end of the communication channel. The similar semantics of the *request port* in SoaML offers this construct to be mapped to the messaging end from the BPMN. The aim of this mapping is the reflexion of the data channel target in the service consumption of the modelled architecture. Table 6 illustrates the mapping of the notation.

Table 6. Message "End" to Request

	BPMN	SoaML
Construct	Message "End"	Request
Notation		

Mapping Rule 7: Process fragment (pattern) to Service Contract

There is no single construct in BPMN that resembles a service contract. You need to analyze the BPMN processes and identify process fragments that can be mapped to *service contracts*. A service contract defines a service specification that defines the roles each participant plays in the service, and the interfaces they implement to play that role in the service. We can however, define a pattern of BPMN constructs that can be mapped to a service contract.

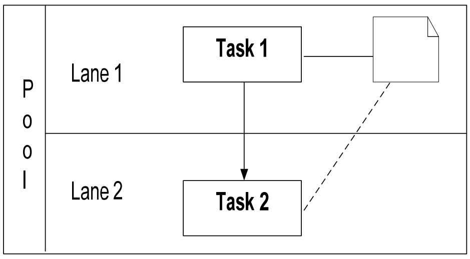
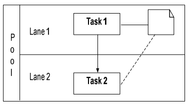
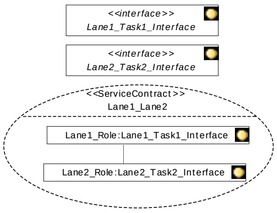


Figure 4. Rule 7 Transformation Pattern

The pattern (see Figure 4) describes a task sequence connected by a sequence flow, but the participants are represented through different lanes in the same pool. The two tasks that belong to a service contract also share a data object. Table 7 illustrates the mapping of the notation.

Table 7. Process fragment (pattern) to Service Contract

	BPMN	SoaML
Construct	Lane1→ Lane2	Service Contract
Notation		 <<interface>> Lane1_Task1_Interface <<interface>> Lane2_Task2_Interface <<ServiceContract>> Lane1_Lane2 Lane1_Role:Lane1_Task1_Interface Lane2_Role:Lane2_Task2_Interface

3.3 Tool Support

In last section we provided mapping rules of the high CIM-level service modelling with the aid of the BPMN. This notation is well-known and established since the beginning of the 21st century, moreover it has been standardized and there are more than 50 products, both commercial and open-source, providing the implementation of this standard [13]. The particular considerations with respect to modelling services by the business users are that there is a little awareness of the services by CIM-level users, on the one hand, and even if there would be any knowledge about it, there are no direct constructs describing the services on the CIM-level in the BPMN notation anyway. Of course the upcoming BPMN 2.0 [14] standard includes the services modelling and the according constructs for it, but it only rules out the second, more technical problem, and not the first one – understanding.

For the solution of this problem we propose a semi-automated approach in this section based on a model-to-model (M2M) transformation from CIM-level BPMN models to PIM-level SoaML-based models. Those models on the higher abstraction level in BPMN would be analysed through a set of mapping rules and would result in a service model representing according constructs and architectures needed for the comprehensive PIM-level model as a basis for the further transformation to the PSM-level. The further section content comprises the manufacturing example and the mapping rules identified and needed for the services mapping from CIM- to PIM-level models. In addition there are technical details of the transformation presented for the BPMN to SoaML mapping set giving a short insight into the serialisation of the models during transformation.

As an example of the technical solution we consider the mapping rule 7 for service contract (also see Figure 4). In the following we show how the pattern identified for the recognition of the service contract on the CIM-level is technically transformed into the corresponding PIM-level construct. We consider the specific function names in ATL transformation file out of scope and concentrate on the XML representation of the source and target models. Through the rule 7 eight objects of the BPMN model are being translated into six objects of SoaML model (see Table 8). The graphical representation of the SoaML input models is taken from the SoaML Editor developed in SHAPE project.

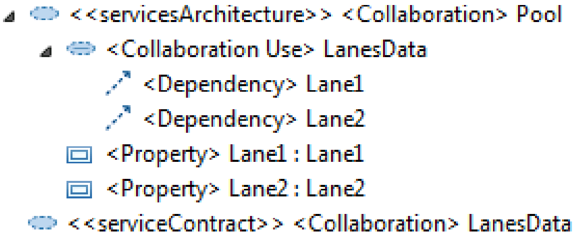
Table 8. Transformation XML mapping

BPMN	SoaML
Lane1	Property Lane 1, Dependency1
Lane2	Property Lane 2, Dependency2
Association1	-
Association2	-
SequenceFlow	-
Task1	-
Task2	-
DataObject	<<ServiceContract>>, <<Collaboration>>, <<CollaborationUse>>

For the technical realisation of the transformations following agreements are valid in the ATL transformation implementation

- CollaborationUse is an element of a ServicesArchitecture. The Properties are elements of a ServicesArchitecture as well. Dependencies are assigned CollaborationUse as children. (One can see the hierarchy graphically in the SHAPE SoaML Editor).
- The directions in which Associations are showing are of no importance, they should only connect the two Tasks in the different Lanes with a DataObject.
- The objects possess hierarchy structure relations, in particular CollaborationUse containts a reference to the according ServiceContract, Properties a reference to the according Participant, Dependencies a reference to the according Properties.

The transformed ServiceContract element according to the rule 7 can be seen in SHAPE SoaML Editor, which shows not only the structure of the transformed element and accompanying relations and properties but also the SoaML stereotypes applied during the transformation (see Figure 5).



<<servicesArchitecture>> <Collaboration> Pool
<Collaboration Use> LanesData
<Dependency> Lane1
<Dependency> Lane2
<Property> Lane1 : Lane1
<Property> Lane2 : Lane2
<<serviceContract>> <Collaboration> LanesData

Figure 5. Hierarchy of objects in SHAPE SoaML editor

4. ILLUSTRATIVE EXAMPLE

The CIMFlex editor is a tool developed in SHAPE project. The CIMFlex editor allows the user to create and refine a semi-formal model of a business process, an organisational structure, a data structure or business rules based on the input coming from the domain users. The editor is able to create, change and store these types of models in EPC or BPMN notation. As storage format XML files are generated. The target users of this component are the domain user and especially the business analysts. From an architectural point of view the component has two interdependencies with other components for its output. The information, which is required for the creation of a CIM model, will be derived from the use cases by the domain users. The output of the CIM level editor can have two different forms depending on its purpose. On the one hand, a model on CIM level in BPMN notation can be used as the technical information description draft, giving a starting point for the transformation into BPEL for further execution of the resulting model or the enrichment with further technical information. On the other hand the output of the CIM level editor is the starting point for the CIM to PIM transformation. In this case the editor does not provide the models in BPMN notation, but transforms them into SoaML models. The conceptual and technical details of this transformation are described in the Section 3. The prototypes of this transformation are available on the SHAPE website (http://www.shape-project.eu/).

Figure 5 depicts the Saarstahl Manufacturing example modelled in the CIMFlex editor, which partly implements the BPMN notation. There is a pool named *Manufacturing* representing the cooperation between two counterparts of the process, namely *Customer* and *Manufacturer* represented by BPMN lanes. The starting event is followed by a BPMN task on the *Customer* side fulfilling the purpose of order preparation. As soon as the *Order* represented by a BPMN data object reaches the *Manufacturer*, it performs a purchase operation and leads the way to the receiving order by the *Customer*. The process ends with a BPMN end event. In the following we apply a set of mapping rules to illustrate the transformation from this BPMN model to SoaML.

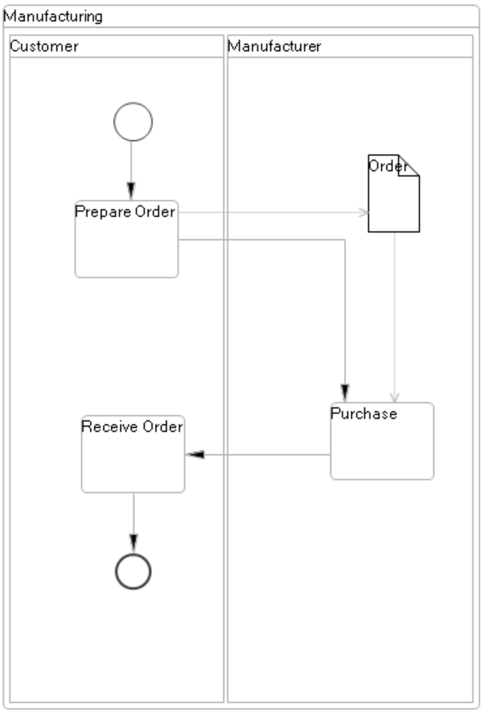


Figure 6. Manufacturing process – input model

After the transformation application the following model would emerge through the rules described before:

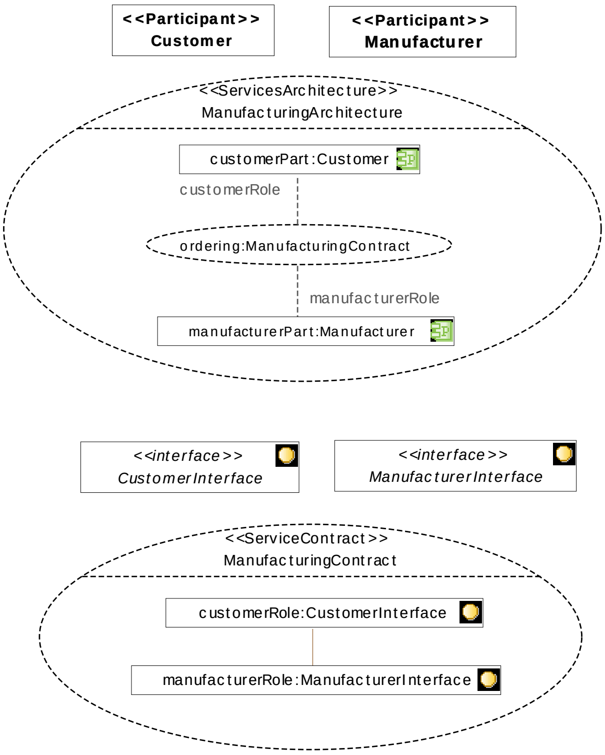


Figure 7. SoaML services architecture – output model

As we can see, the *lanes* constructs from the BPMN notation example are translated into the *participants* constructs in SoaML (rule 4). At the same time a pattern identified by the rule 7 translates the interaction between *Customer* and *Manufacturer* into a *service contract* within the services architecture.

5. DISCUSSION

There is an industrial interest in ensuring a good connection and mapping between business models as expressed in enterprise architectures and IT models as expressed in technical system architectures, which are commonly realised as service oriented architectures (SOAs). The gap between these models is not trivial to close and we believe this stems from the fact that this is not only a technical task, but also one that requires collaborations and decisions to be made by both business and system stakeholders. Obviously, modelling guidelines, mapping rules and software tools, as those developed in SHAPE, to model and execute semi-automated model transformations can be used in the alignment of business and IT models, in particular for simple one-to-one mappings.

However, for more complex mappings, as evident in the mapping to service contracts, it is more of a business and IT design choice. Although we have presented a pattern for identifying service contracts from analyzing BPMN processes, the choice of which tasks to include into a service contract is still not clear. This relates to the service choreography that defines the behaviour of the service contract. The issue is to include all tasks and all interactions that make up a suitable choreography. This choreography may include several interactions and passing of messages across two or more pools in the case of multi-tier service contracts. This is a business design choice which ultimately depends on the people involved and how they best understand the business operations.

The overall approach presented by SHAPE is how to model your processes starting on CIM, over PIM down to PSM yielding to some system which reflects the processes described on CIM level. For green field projects this ‘top-down approach’ might be a suitable approach. In the Saarstahl use case they benefited from improved practices for business and IT modelling to improve communication and synchronisation between business requirements and IT solutions. However, Saarstahl also noted that most companies have already an existing IT landscape and running systems modelling their processes. A reverse engineering or bottom-up approach should be investigated to cover this missing part.

6. CONCLUSION AND FUTURE WORK

In this paper we have presented an overview of the SoaML modelling language and its application for describing both a business and IT perspective on SOA. Furthermore, we have defined a set of model transformation rules that can be used to map BPMN models to SoaML models. The application of these mapping rules have been tested in industrial use cases in the SHAPE project with the objective of aligning business and IT models. The SHAPE technologies improved practices for business and IT modelling and improved communication between business requirements and IT solutions.

One aspect of our guidelines that requires further work is to identify and describe additional patterns and guidelines for mapping to service contracts. In particular better support for multi-tier service contracts requires additional work. Furthermore, the mapping rules defined must also be updated and aligned with the ongoing BPMN 2.0 specification, which introduces some new process and service language constructs.

7. ACKNOWLEDGMENTS

This research was co-funded by the European Union in the frame of the SHAPE FP7 project (ICT-2007-216408). The authors would like to express their acknowledgments to SHAPE colleagues.

8. REFERENCES

[1] M. Stollberg (ed.), "SHAPE Project Whitepaper", SHAPE STREP, 9 June 2009. http://www.shape-project.eu/wp-content/uploads/2008/01/shape_whitepaper.pdf

[2] OMG, "Service oriented architecture Modeling Language (SoaML), FTF Beta 2", Object Management Group, OMG Document ptc/2009-12-09, December 2009. http://www.omg.org/spec/SoaML/1.0/Beta2/PDF/

[3] OMG, "Service oriented architecture Modeling Language (SoaML), FTF Beta 1", Object Management Group, OMG Document ptc/2009-04-01, April 2009. http://www.omg.org/spec/SoaML/1.0/Beta1/PDF/

[4] OMG, "Business Motivation Model, Version 1.0", Object Management Group (OMG), OMG Document formal/2008-08-02, August 2008. http://www.omg.org/spec/BMM/1.0/PDF/

[5] OMG, "Business Process Model and Notation (BPMN), Version 1.2", Object Management Group (OMG), OMG Document formal/2009-01-03, January 2009. http://www.omg.org/spec/BPMN/1.2/PDF

[6] A.-W. Scheer and M. Nüttgens, "ARIS Architecture and Reference Models for Business Process Management", 2000. http://www.wiso.uni-hamburg.de/fileadmin/wiso_fs_wi/EPK-Community/LNCS_Geschaeftsprozessarchitektur.pdf

[7] ITU-TS, "Basic Reference Model of Open Distributed Processing - Part 1: Overview and guide to use the Reference Model", Rec.X901 (ISO/IEC 10746-1), 1995.

[8] ITU-TS, "Basic Reference Model of Open Distributed Processing - Part 2: Descriptive model", Rec.X902 (ISO/IEC 10746-2), 1995.

[9] ITU-TS, "Basic Reference Model of Open Distributed Processing - Part 3: Prescriptive model", Rec.X903 (ISO/IEC 10746-3), 1995.

[10] ITU-TS, "Basic Reference Model of Open Distributed Processing - Part 4: Architectural Semantics", Rec.X904 (ISO/IEC 10746-4), 1995.

[11] IEEE, "IEEE Std 1471-2000: IEEE Recommended Practice for Architectural Description of Software-Intensive Systems", IEEE, October 2000.

[12] INRIA & LINA, "ATLAS Transformation Language (ATL) Project Documentation". http://www.eclipse.org/gmt/am3/doc/ (last visited 2010).

[13] BPMI, "Current Implementations of BPMN", Business Process Management Inititative (BPMI). http://bpmn.org/BPMN_Supporters.htm (last visited 2010).

[14] OMG, "Business Process Model and Notation (BPMN), FTF Beta 2 for Version 2.0", Object Management Group (OMG), OMG Document dtc/2010-05-03, May 2010. http://www.omg.org/spec/BPMN/2.0/Beta2/PDF