Reference Model for service oriented Business Software based on Web Service Nets

Maik Herfurth
FZI Research Center for Information Technologies, Karlsruhe
herfurth@fzi.de

Thomas Karle
PROMATIS software GmbH, Ettlingen
thomas.karle@promatis.de

Frank Schoenthaler
PROMATIS software GmbH, Ettlingen
frank.schoenthaler@promatis.de

Abstract Configurable business software solutions are increasingly being implemented based on service oriented architectures (SOA). Hereunto model driven approaches for the definition and implementation of such software systems are indispensable. In addition reference models with pre-defined business processes can accelerate decisively the implementation of standard software solutions. In this paper we present a new integrated service oriented approach of a multi-layer model. XML nets, a kind of high-level Petri nets based on XML standards and Web Service nets that are additionally based on web service standards provide the creation of reference models for configurable business software solutions and can be used for the implementation. The reference models are structured in layers of different kind of granularity. Within these layers complex processes can be orchestrated using business process components. The business flows within the capsulated processes are defined with Web Service nets considering the respective standards of web services. With this hierarchical model, business processes of a configurable business software solution can be described from the rough process flow up to detailed functions and detailed instructions based on a formal model. Due to the use of web service standards, the process models might be assigned to implemented WS-BPEL processes of the business software or WS-BPEL processes might be generated using new or updated process models.

1 Introduction

New technologies and paradigms in IT demand new models and approaches when implementing business software. Up to now business software was mostly implemented based on the life cycle of software engineering in a classic manner, going from requirements engineering up to system maintenance. Modeling business processes in line with the requirements analysis is the basis for the systems design and architecture. One problem of this approach when implementing standard software is up to the deviation of specific business processes and the given processes within the standard business software solution. Business process models with a service oriented approach reduce the complexity and enable for a flexible business process

management by orchestrating services. The approach described here is a multi-layer model and enables to illustrate services on different levels, depending on the degree of abstraction. All services can be composed by means of web service standards and combined with each other vertically by using refinements.

2 Problems and Requirements

To reduce efforts and expenses for a system implementation a lot of companies decide to implement standard software e.g. for Enterprise Resource Planning (ERP) or Customer Relationship Management (CRM). In the following sections the problems and the resulting requirements of such implementations are discussed.

2.1 Problems with the Implementation of Standard Software

During the implementation companies might notice that the software's business processes do not correspond with their processes and are not appropriate to the operating procedures. The comprehensive functionalities of standard software and the resulting complexity make for a divergent understanding of the term *standard* in relation to processes. Heterogeneous IT-system landscapes are for the most part another reason for problems in implementing standard software. Additional expenses and efforts are created for complex interface solutions to integrate different systems. In the end and after all, a standard software solution should be less expensive than developing an individual software solution. Often software standard solutions offer the ability to be upgraded with modules which can be purchased separately. Usually when implementing business software, large parts of the company are affected because the business processes provided by the software are crossing divisions. Respectively standard software provides comprehensive functionalities and pre-defined business processes which can only be adapted within the limits given by the software itself. Lack of transparency of the business software's functionalities based on its complexity of such precast systems on one side and vague requirements on the division side, which shall use the systems, cause problems with the implementation. Vague requirements result from missing or very inexact, i.e. informal business process definitions. An aggravating factor: often users cannot give an exact process definition due to missing abstraction and structuring abilities. Even if processes incl. detailed requirements have been defined in the run-up, a direct – or if given – even an automated mapping on the processes and functionalities of standard software is quite difficult. Reasons for this are the conceptual and methodical differences between the used tools and procedure models in the analysis phase and in the documentation of the actual processes of the standard software. For users from different company divisions the implementation of the business software can be challenging as first of all totally different abilities are required than for the normal daily business tasks and on the other hand the project work has to be done in addition to daily standard business tasks. The familiarization with such software systems is difficult as the documentation will be very voluminous due to the software's complexity. Beyond that the ultimate surplus value of such solutions only becomes apparent by the interaction of several

corporate divisions, realized by the software. A comprehensive view on the solution remains concealed to most users. This problem often comes up in the system documentation when it is purely functional-oriented rather than process-oriented. The listed aspects all together tend to lead to long project terms and eventually to budget overruns. And often functions that are not covered by the standard software only come up when testing the system.

2.2 Current Requirements for a Process oriented Approach

The described problems result in requirements for a business software documentation that has to be oriented at the business processes. Beyond that users have to be lead through the processes via top down approach from rough general operational to detailed processes which describe the corresponding realization with the business software down to functional levels. This structure is thought to transfer general business terms from the higher, rough levels semantically to the terms of the standard software in detail processes. And by use of a formal model vague process definitions will be prevented. The possibly given informal requirements from the divisions should be assigned to certain positions in process models in order to be able to check the degree of coverage of the standard software in an early stadium. One solution approach for the described problems may be found in service oriented paradigms. Service oriented business software based on capsulated services eases the horizontal and vertical integration of different systems and allows for transparency. Process models should consider the standards of a service oriented architecture, i.e. the standards of web services, so that an orchestration of complex processes from existing services is allowed for different levels. Process models should be linked directly to corresponding Web Services Business Process Execution Language (WS-BPEL)[1] implementation of processes or enable for automated generation when making changes or new definitions. Unlike mere Enterprise Application Integration (EAI) solutions, a new generation of service integration apart from the integration of applications on a functional level shall be provided. Also integration on a higher level should be considered, i.e. integration on a business service level to allow for orchestration of complex SOA-based software components.
This would make it possible to assemble applications and realize business process driven model-based integration solutions. In the near future Enterprise Application Integration solutions will change to Enterprise Service Integration Solutions [SL03].

[1] The Web Services Business Process Execution Language (WS-BPEL) was defined by IBM, BEA and Microsoft in 2002. The current version is WS-BPEL 2.0, standardized by OASIS in 2007. WS-BPEL allows for a description of business process activities as web services and how to orchestrate them in order to accomplish a specific purpose (orchestration logic). It provides a language for the specification of executable and abstract business processes.

3 Existing Approaches in Literature

Oba and Komoda present in [OK01] a business process-based integration and method as a possibility for the implementation of Enterprise Application Integration (EAI), which builds up on workflow techniques. The business process-based integration separates Business Information Systems in business processes and logics that describe a system. The new type of a workflow for EAI uses a CORBA-based interface [OM03a] to integrate different system types and a statistical interim model to control processes dynamically. In [PS03] Pendyala, Shim and Gao describe an implementation of an Enterprise Application Integration totally based on an XML structure. It offers – unlike traditional EAI approaches with little flexibility by using a Black Box approach – a higher productivity and a high scalability as well as consolidation options of applications after company fusions. This is enabled by using XML schemas, which offer standardized and flexible data structures. In [KB03] the authors even go a step further and describe an MDA-based [OM03b] *Enterprise Model Integration (EMI)* approach, an exemplary system for meta model integration. This approach is based on object-oriented meta model concepts, which can describe context-specific, integrated model languages. Business Software development projects can integrate heterogeneous target systems by means of these model languages. However, all of these approaches are based on traditional EAI approaches and therefore not on the advanced approach for a service oriented architecture, which is based on XML standards. [RS03] present an EAI multi layer model which is allocating aspects of enterprise integration on top of each other and required activities accordingly. For the multi- layer model, XML is proposed for the data format of the transactions between the systems. [GL04] illustrate the main problems of design approaches of EAI architectures in their paper and declare the service oriented architecture to be the currently best architecture framework. In comparison web services do show advantages against traditional EAI solution. Contrary to typical EAI solutions they are easier to develop and maintain. Open standards exist for a broad adaption of web services and their flexibility in integration solutions is achieved by loosely coupling involved applications. Web services allow for breaking down complex applications into small independent logical units, which can be capsulated and developed as different business components of an ERP system. An approach to model cooperative business processes and the transformation into a service oriented architecture is also described in [SD05] where a continuous method for modeling cooperative, business-wide business processes and their transformation into an adequate IT architecture is described and evaluated. Zimmermann et al. introduce exemplarily in [ZD05] an order management scenario with the approach of service oriented architecture choreography of business processes. The paper gives an overview of WS-BPEL-based process choreography, the underlying architecture and beyond that introduces a component model as multi-layer model. Hamadi and Benatallah [HB03] propose a petri net-based model for web service composition. A Petri net-based algebra is used for modeling web services control flows. The model is expressive enough to capture the semantics of complex service combinations and their respective specificities. The obtained framework enables declarative composition of web services.

The existing approaches do not provide a comprehensive framework that covers process modeling based on a formal model, the use of current web service standards and the consideration of different roles for specific process steps within detailed layers.

4 Description of the Multi-Layer Model

With the multi-layer model, business processes of standard software can be described on several levels. With the multi layer model defined below SOA concepts are considered on all layers. In chapter 4, we present and explain the description of the mulit-layer model and its abstraction levels. We refer to the business process *Order2Cash* of our reference model as an example which is explained in detail in chapter 6. The business process *Order2Cash* describes the complete process from the assignment of an order to the payment. Fig. 1 shows the layers on different abstraction levels. The description of processes is done based on so called XML nets, a kind of higher Petri nets, and based on Web Service nets, which again are based on XML nets and contain all necessary information to define service-based processes. That way they may be used to generate practicable WS-BPEL code [KM05].

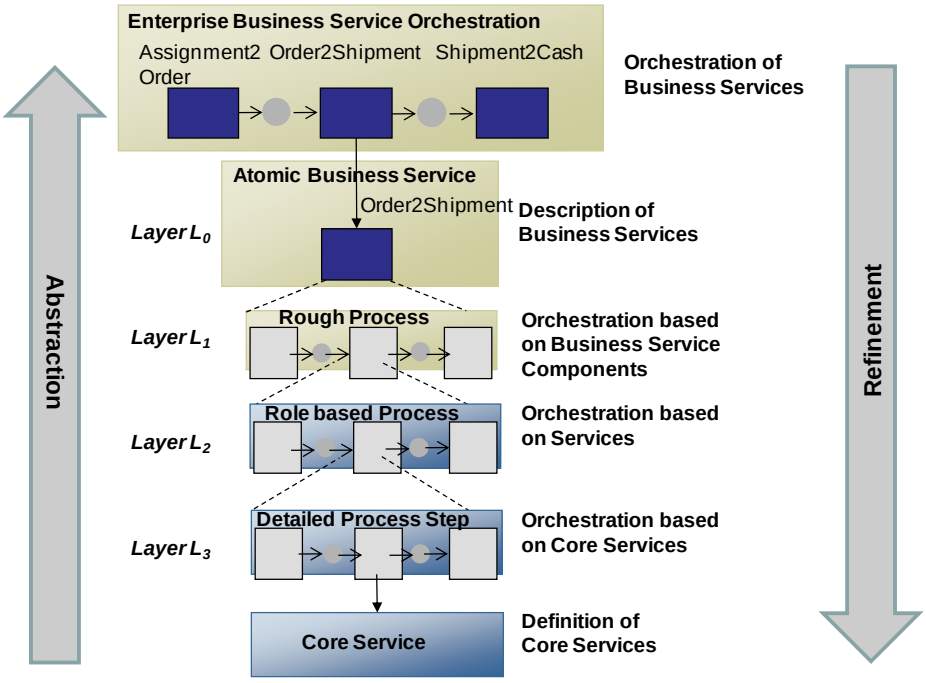

Fig. 1: Multi-Layer Model

The multi-layer model presented here is made up of four layers: *Business Services Description Layer* (L_0), *Business Service Components Orchestration Layer* (L_1),

Service Orchestration Layer (L_2) and *Core Service Orchestration Layer (L_3)*. On an upper level so called business services can be orchestrated to company-wide or company-spanning processes. A business service is an implemented self-contained process which illustrates the processing of one business object or business event to another business object or business event, for example: processing a *Sales Order* to *Customer Goods Receipt* within the business service *Order2Shipment*. These are business functionalities on a higher, super ordinate abstraction level. A business service cannot be assigned a uniform role within a company because one business function usually inherits several roles. The upper levels L_0 and L_1 are described as XML nets. At these places, XML schemas are deposited which describe the launch of a web service that starts the process in a business service. Details on the web services used are described in sub-levels. For transitions on higher levels there are no details being deposited, but rather a reference to the sub-level net. A single business service is described on layer L_0. Here the launch of the business service can already be deposited, i.e. the launch of the first web service, which illustrates the business service processing. Business services can be interleaved optionally, i.e. from orchestrating different business services, new business services are created, which might be used on other detailed levels in line with the enterprise orchestration. L_0 describes atomized business services. These are business functionalities being reusable process components that cannot be split any further on the company level.

Layer L_1 illustrates *Business Services components*. These business service Components are *rough* process steps within a business service. These are again services, which are only used within those business services. The description of these Business Service Components is done by business terms in substantiated form to point out the higher abstraction level. On this level the assignment of roles is not possible yet, as here as well in underlying detailed steps several roles can be involved. The levels L_2 and L_3 are described with Web Service nets and allow the assignment of certain roles to process steps. By these means Web Service nets can explicitly support role-based activities. Layer L_2 describes the process steps of a Business Service Component. These process steps, i.e. services, have to have corresponding roles assigned which use this service at exactly this point of process. The system fulfils this role when using automated process steps. Layer L_3 describes the detailed procedure within a service for exactly one role. Here SOPs (Standard Operating Procedures) are described for single roles that can be used for documentation, knowledge management and concrete personalization in an ERP solution. This describes from the point of view of a certain role how one or more services are connected and can then be used accordingly. For example here it is described how an employee with the role *Warehouseman* can create a *Picking List* with the appropriate services. Layer L_3 accesses so called *Core Services*, which are single web services that cannot be detailed any further. Core Services are the smallest unit for process steps in our reference model. A Core Service would be a *Create Picking List*, i.e. a web service that saves a given position to a *Picking List* in the data base. Core Services describe the system-based layer.

The choice for these four layers can be reasoned as follows: the on-top layer illustrates the orchestration of comprehensive business services on a company level. The low-level layer L_3 describes directly the system-based level. In L_3, detailed and concrete SOPs are described for determined roles. To absorb the discrepancy between

these two very different abstraction layers there are two layers L_1 and L_2 whereas L_1 enables for a rough understanding of services and on the other hand L_2 introduces the role-oriented layer for a first user relation.

5 Formal Description of the Reference Model

To model business processes on different levels XML nets and Web Service nets [KM05] with corresponding extensions are used. Web Service nets can be transferred directly into practicable WS-BPEL processes. When speaking of the multi-layer model the term S*ervices* can be used on every layer since all processes are capsulated and are defined based on Web Service nets. Each and every business process can be understood as self-contained service with input and output parameters. Within the service each transition represents an interface to a web service. If a transition occurs and delivers parameters via XML document, the call of a web service is defined. The web services interface is described by the XML schema valid for the XML document. The methods and parameters of the interface as well as the definition of formats and protocols for the use of the web service interface are described by WSDL [W301]. In the following first the XML nets, which are also the basis for Web Service nets, are described. Afterwards the extensions for Web Service nets and the transformation to WS-BPEL are explained.

5.1 XML Nets

XML nets are based on Petri nets, a formal graphic process description language, which combines the advantages of visual representation of processes with formal semantics. Petri nets consist of places and transitions. Places are illustrated by circles, transitions by rectangles. Directed edges connect places and transitions. The electronic exchange of process objects when processing operational business processes calls for a precise and integrated description as offered by XML format [WC04]. This format is platform-independent and can be presented in any form [BK06]. Especially efficient inter-organizational business processes in the field of e-commerce require the integration of electronic document interchange and inter-organizational process management [LO03]. Already in 1999 [LE99] XML was seen as integrating element for application integration. The reference model consisting of a derivate of XML nets, supports modeling and performance of services, based on XML. In XML nets [LK03] static components are expressed by XML schema diagrams [W304], and each component is represented by an XML schema. With XML nets places are defined as containers for a number of structured objects and combined for one common XML schema and tokens are restricted for valid XML documents concerning the schema. XML documents are processed by occurring transitions. Transitions represent activities, which define a class of operations on the XML documents of the adjacent places. Transitions might be inscribed by logical expressions of variables. This has to be fulfilled for the processing of the operation. The expression of an instance of a variable accepts the value *true* or *false*. On edges, filter schema diagrams express the transformation of XML documents. *XQuery*

expressions [WC05] define the data manipulation the processed object by the actions *Create, Change, or Delete.*
XML nets allow for a detailed analysis and simulation of distributed business processes. Furthermore XML nets support the process of finding relevant process fragments and their assignment to corresponding organizational divisions, in order to derive an improved, process-oriented organizational structure. XML nets enable for integrated modeling of procedures by means of their relevantly structured objects and a detailed description of document and data manipulation. Already during the development phase of business processes for business software, processes can be visualized. Beyond that a re-structuring of these processes can be made graphically at the process schema. The high formalization grade of XML nets can be used to support and steer partially automated processes, which e. g. are implemented via WS-BPEL [LK03].

5.2 Web Service Nets

The logic behavior of services can be described by *Web Service nets*. A Web Service net is an XML net, which has exactly one place without incoming edges, defined as *Input place* of the Web Service net and has exactly one XML document as initial configuration. Furthermore a Web Service net has exactly one place without outgoing edges, which describes the *outgoing place*. Every transition, which is included into the net, has to be connected on direct way from the input place all up to the output place. XML schemas of input and output places tell you which messages the service expects and which message it delivers back. Other XML schemas of pre-set places and post-set places of the transition describe the communication with a web service. However, only the XML schemas from the input place and the output place of a Web Service net are needed for the generation. A Web Service net has one token for the input place, defined by XML elements, which launches the service. WS-BPEL processes can be derived from Web Service nets [KM05]. The data structure to store Web Service nets is based on *Petri Net Markup Language (PNML)* [BC03], a standardized XML-based structure. The additional details for web services are assigned to the transitions as these have a relation to an external web service. If there is to be communication with a web service then it has to be assigned to a transition from an UDDI repository in order to access a corresponding WSDL. In the following the additionally required information, which are needed for implementation of practicable WS-BPEL processes are described.
The path to the WSDL file is shown within the element ***wsdl***. In addition the details ***partnerLinkType, portType*** and ***operation*** have to be named in order to know how to communicate with which web service. It is a precondition that in each WSDL-file the ***partnerLinkTypes*** are defined. Beyond that the ***activity*** has to be named. It identifies which basic activity of WS-BPEL this action complies with. In addition to ***reply***, ***receive*** and ***invoke*** activities, which communicate with a web service, ***empty*** and ***wait*** can also be provided. The last two activities do not call for ***wsdl***, ***partnerLinkType***, ***portType*** and ***operation***. When the activity ***wait*** is called upon, the element ***time*** expects a term enter. Under this element ***time*** there are the elements ***unit*** and ***duration***. ***Unit*** shows the time

unit and ***duration*** the term. The exact time duration has to be entered in the sub-element ***average***. The variables, needed by basic activities for input or output variables, illustrate the input and output places with their XML schema.

The basic structure of a Web Service net consists of two transitions and two places. The first transition has to be a ***receive*** activity and the second one has to be a ***reply*** or ***invoke*** activity. Between those two transitions then the procedure of the processes can be issued as desired. Fig. 2 shows such a basic structure of a Web Service net.

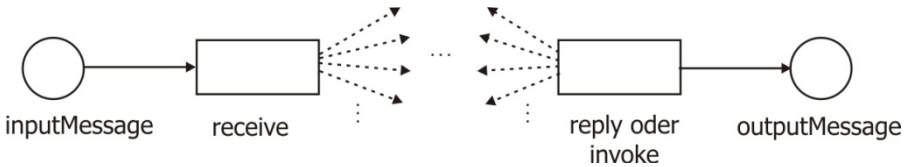

Fig. 2: Web Service Net Structure

Pre-defined WS-BPEL processes of the standard software can be assigned to Web Service nets. Changes to implemented WS-BPEL processes or new implementations of processes can be supported by a corresponding generator.

5.3 WS-BPEL Transformation

A Web Service net illustrates a process, which is orchestrated from different web services. A Web Service net makes up the basics for transformation into WS-BPEL. I.e. processes, which have to be illustrated as WS-BPEL processes are modeled as Web Service nets in our multi-layer model. Functionalities offered by web services can – as described before – be assigned to transition of a Web Service net. Transitions of Web Service nets are translated into basic activities of BPEL. A transition can therefore correspond to e. g. an ***invoke***, ***reply*** or ***receive*** activity. Transitions require input and output messages. Since a message can contain several parts and a transition several places within pre-set and post-set of the transition, and a part requires – just like a place – an XML schema, it would be contiguous that each place corresponds with a part of a message. The procedure of the XML net results in structuring activities. For the transformation into WS-BPEL and the handling of web services, the XML nets include an extension and the new nets are called Web Service nets. All services on layer L_2 and layer L_3 are modeled as Web Service nets. If a net should not fulfill all requirements valid for a Web Service net, it cannot be translated into WS-BPEL. To generate WS-BPEL processes from Web Service nets, a WS-BPEL generator was developed, which generates WS-BPEL code building up from a transformation algorithm by [VL05].

5.4 XML Framework INCOME2010

The software toolset INCOME2010 [AI07], which is currently being developed by the Institute AIFB at the University of Karlsruhe (TH) in cooperation with the FZI Research Center for Information Technology and PROMATIS software GmbH, is used for modeling the presented service oriented multi-layer model based on Web Service nets.

Based on XML nets and derivates of XML nets, INCOME2010 introduces a framework for modeling and analyzing XML nets, e. g. Web Service nets, and offers concepts for realization of service oriented architectures [KO08]. The storage of Petri nets in INCOME2010 is based on *Petri Net Markup Language (PNML)* [BC03], a standard, which defines an XML schema, by which Petri nets can be stored in XML data. It offers the ability for modeling procedures and functionalities of business objects such as *orders*, *invoices* or also *picking lists* based on XML documents. By means of this framework and the *XML Net Plug-in*, business objects based on XML nets or their derivates can be modeled graphically and described semantically. The structure of business objects can be defined with XML schema. Filter schema diagrams can be used to define the transformation of the business objects. Hereunto the framework uses the *Modeler* to graphically model business processes and organizations. Business objects such as XML filter schemas and transition inscriptions are developed and processed with the *OjectSchemaEditor*. To exchange XML nets, XNML is available, which was developed as an extension for the exchange format PNML and saves XML net-specific information such as transition inscriptions and schema paths. XNML can be graded up for Web Service nets.

With this framework now services based on Web Service nets can be modeled and orchestrated. Modeled process schemas can be added by service-specific parameters and exact information of attributes and values of business objects. The framework enables for simulation on different layers. The *Web Service Plug-in* supports the service oriented architecture concept by offering a *BPEL generator* with which XML nets can be converted into WS-BPEL code (BPEL4WS) [IM03] and other processes and services can be integrated [KO08]. In an upcoming version INCOME2010 will also support the new standard WS-BPEL 2.0 [OA07].

6 Example - Reference Model Order2Cash

In Fig. 3 a company-wide business process's service-orchestration *Order2Cash* is shown on the upper abstraction level of the multi-layer model: the *Enterprise Orchestration Level.* Orchestration defines a company process, which in its core is made up of three business services and describes the procedure from the triggering business cases on the left side (*Confirmed Quotation*, *Direct Order* etc.) up to the end result: *Invoiced Order.* The required interfaces to business services used in special cases which differ from the standard procedure of the Order2Cash-Process are accentuated in grey. The interfaces are defined by assigned XML schemas to the specific places.

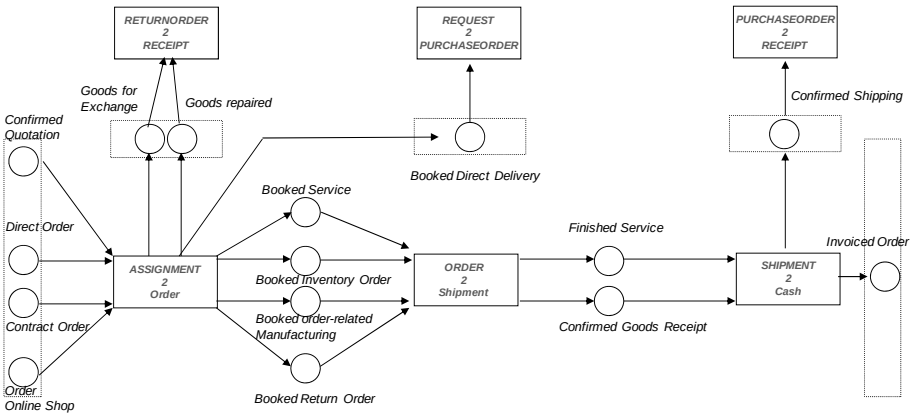

Fig. 3: Service Orchestration Order2Cash

In Fig. 4 the Business Service Layer L_0 for the business service *Order2Shipment* is explained and shows a reusable process component. This layer describes the business service and its interfaces. It provides an entry to detailed descriptions of the processes on the refined layers.

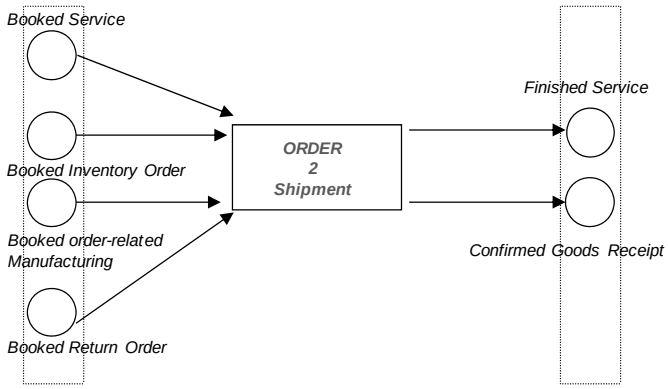

Fig. 4: Business Service Layer L_0 Order2Shipment

Fig. 5 shows Layer L_1, the layer *Business Service Component Orchestration* for Order2Shipment with a rough illustration of the service on a high, understandable level by using common business terms. The place *Planning & Structurizing successful* is described in the XML net by a XML schema diagram and the edge is provided with an XML filter schema as a graphical description. The technical specification behind is an XQuery routine for data manipulation of variables and constants corresponding to the schema of the place.

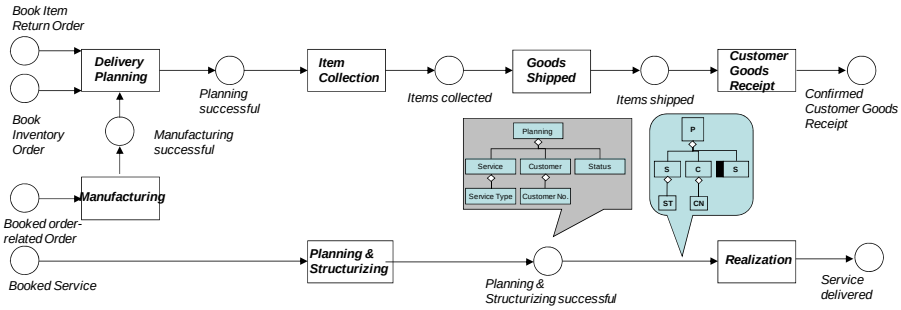

Fig. 5: Business Service Component Orchestration Layer L_1

Fig. 6 shows the detailed process design on Layer L_2, the role-based Orchestration. The simplified Web Service net shows XML schemas and filter diagrams for graphic notation, which are assigned to places and edges. The XML elements *Picking List created* is of the type *Boolean* and shows whether or not the picking list was created. As soon as the transition *Create Picking List* occurs, the XML documents, which contain *Picking List Number*, *Shipping Data*, *Item Numbers*, *Customer Numbers*, *Order Numbers* and an element *Picking List created* with the value *false* will be deleted of the place *Finished Planning* via an XQuery routine on the technical basis for the implementation according to the filter schema diagram of the incoming edge of the transition. After the transaction occurs, the XML documents are processed according to the specification of the filter schema diagram of the outgoing edge via an XQuery routine as the element *Picking List created* gets the value *true* and creates a *Picking List Number* within the XML schema in the following place.

The black bars of the filter schema diagrams represent the manipulation filter diagrams, to create or delete XML documents within this context, which also are represented via an XQuery routine. Elements which are illustrated with an *A* define place holders of the data type *AnyType* to instantiate elements of any type. The transition *Create Picking List* is assigned to the role *Warehouseman*.

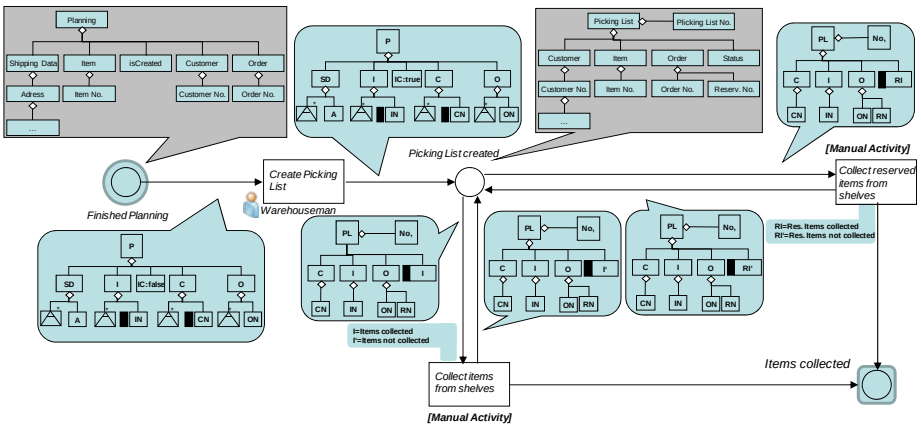

Fig. 6: Role-based Service Orchestration Layer L_2

7 Summary and Outlook

In this article we present a new approach to a service oriented multi-layer model for reference models. The multi-layer model is based on Web Service nets which enable for orchestration of services on different levels. Beyond that process descriptions can be illustrated directly in WS-BPEL in lower levels (Layer L_2 and Layer L_3). This enables for a direct assignment to pre-defined processes of standard software. Additionally newly designed or changed processes are supported and can be implemented by a corresponding generator. The multi-layer model is characterized by considering different abstraction and detail levels and therefore enables a controllability of the complexity of comprehensive business software solutions. By generating WS-BPEL processes via the framework INCOME2010 an implementation of services can be started right from modeling and business processes incl. organizational changes within the company can be applied. The presented approach supports role-based assignments to services considered on a detailed level which can also be realized immediately by upcoming standards such as BPEL4People [IS07]. The described approach is currently used and evaluated in several practice projects. The next planned step is to illustrate requirements based on this model in order to have an early audit on the coverage degree of the standard software. Furthermore an enhancement of the WS-BPEL integration is planned, i.e. the direct link from WS-BPEL-based processes of a standard software and their models. Another step is to upgrade the existing BPEL generator based on the model taking into consideration new standards such as BPEL4People.

References

[AI07] Institut AIFB Universität Karlsruhe (TH): INCOME2010, URL: http://www.aifb.uni-karlsruhe.de/Forschungsgruppen/BIK/income2010/index.htm, 2007.

[BC03] Billington J., Christensen S., van Hee K., Kindler E., Kummer O., Petrucci L., Post R., Stehno C., Weber M..: The Petri Net Markup Language: Concepts, Technology and Tools, ICATPN 2003, Eindhoven, Netherlands, 2003.

[BK06] Betz S., Karle T., Klink S., Koschmider A., Li Y., Mevius M., Oberweis A., Ried D., Trunko R., Zaich M.: Ein Framework zur Modellierung und Analyse von XML Netzen: Algorithmen und Werkzeuge für Petri Netze (AWPN'06), pp. 1-7, Hamburg, 2006.

[GL04] Gorton I., Liu A.: Architectures and Technologies for Enterprise Application Integration, Proceedings ICSE'04, p. 1, Scotland, UK, 2004.

[HB03] Hamadi R., Benatallah B.: A Petri Net-based Model for Web Service Composition, Proceedings, ADC'03, pp. 1-10, Australia, 2003.

[IM03] IBM, BEA, and Microsoft: Business Process Execution Language for Web Services Version 1.1.5.5.2003, URL: ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf, 2003.

[IS07] Active Endpoints, Adobe, BEA, IBM, Oracle, SAP AG: WS-BPEL Extension for People (BPEL4People), Version 1.0, July 2005, Update June 2007.

[KB03] Kühn H., Bayer F., Junginger S., Karagianis D.: Enterprise Model Integration, EC-Web 2003, LNCS 2738, pp. 379-392, Springer Verlag, 2003.

[KM05] Koschmider A., Mevius M. A Petri Net based Approach for Process Model Driven Deduction of BPEL Code, OTM Confederated International Conferences, Agia Napa, Cyprus, pp. 495-505, Springer Verlag, 2005.

[KO08] Klink S., Li Y., Oberweis A.: INCOME2010, a Toolset for Developing Process-Oriented Information Systems Based on Petri Nets: PNTAP 2008, pp. 1-3, Marseille, France, 2008.

[LE99] Lear A.: XML Seen as Integral to Application Integration, IT Professional, p. 1, 1999.

[LK03] Lenz K.: Modellierung und Ausführung von E-Business-Prozessen mit XML-Netzen, Dissertation, Verlag für Wissenschaft und Forschung, Frankfurt am Main, 2003.

[LO03] Lenz K., Oberweis A.: Interorganizational Business Process Management with XML Nets, in: H. Ehrig, W. Reisig, G. Rozenberg, H. Weber (Hrsg.): Petri Net Technology for Communication Based Systems. LNCS 2472, pp. 243-263, Springer Verlag, 2003.

[OA07] OASIS: Web Service Business Process Execution Language Version 2.0. Primer, URL: http://www.oasis-open.org/committees/download.php/23964/wsbpel-v2.0-primer.htm, 2004.

[OK01] Oba M., Komoda N.: Multiple Type Workflow Model for Enterprise Application Integration, Proceedings Hawaii International Conference on System Sciences, pp. 1-6, Hawaii, 2001.

[OM03a] OMG Object Management Group: CORBA, URL: http://www.omg.org/docs/ptc/06-08-03.pdf, 2003.

[OM03b] OMG Object Management Group: Model Driven Architecture (MDA), URL: http://www.omg.org/cig-bin/doc?ormsc/01-07-01.pdf, 2003.

[PS03] Pendyala S.; Shimj S., Gao J.: An XML Based Framework for Enterprise Application Integration, Proceeding, CEC‘03, California, USA, pp. 1-8, 2003.

[RS03] Reitz, D.; Schmietendorf, A.; Dumke, R.; Dimitrov, E.; Lezius, J.; Schlosser, T.: Aspekte des empirischen Software Engineering im Umfeld von Enterprise Application Integration Lösungen, Preprint Nr. 5, Fakultät für Informatik, Universität Magdeburg, 2003.

[SD05] Specht T., Drawehn J., Thränert M., Kühne S.: Modeling Cooperative Business Processes and Transformation to a Service Oriented Architecture, Proceedings CEC’05, pp. 1-8, Munich, Germany, 2005.

[SL03] Schmietendorf, A.; Lezius, J.; Dimitrov, E.; Reitz, D.; Dumke, R.: Aktuelle Ansätze für Web Service basierte Integrationslösungen, Preprint Nr. 10, pp. 1-57, Fakultät für Informatik, Universität Magdeburg, 2003.

[VL05] Van der Aalst, W.M.P., Lassen K.: Translating Workflow Nets to BPEL, BETA working Paper Series, WP 145, pp. 1-46, Eindhoven University of Technology, Eindhoven, 2005.

[W301] World Wide Web Consortium: Web Service Description Language (WSDL) 1.1. W3C Note, URL: http://www.w3.org/TR/2001/NOTE-wsdl-20010315.html.

[W304] World Wide Web Consortium: XML Schema Part =: Primer (Second Edition). W3C Recommendation, URL: http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/, 2004.

[WC04] World Wide Web Consortium: Extensible Markup Lanuage (XML) 1.0 (Fourth Edition). W3C Recommendation, URL: http://www.w3.org/TR/2006/REC-xml-20060816/, 2006.

[WC05] World Wide Web Consortium: XQuery 1.0: An XML Query Language W3C Candidate Recommendation 3. November 2005,
URL: htpp://www.w3.org/RE/2005/CR-xquery-20051103/, 2005.

[ZD05] Zimmermann O., Doubrovski V., Grandler J., Hogg K.: Service-Oriented Architecture and Business Process Choreography in an Order Management Scenario: Rationale, Concepts, Lessons Learned, OOPSLA'05, pp. 301-312, San Diego, California, USA, 2005.