Semi-automatic generation of Web services and BPEL processes - A Model-Driven approach

Rainer Anzböck[1], Schahram Dustdar[2]

[1] D.A.T.A. Corporation,
Invalidenstrasse 5-7/10, 1030 Wien, Austria
ar@data.at
[2] Distributed Systems Group, Vienna University of Technology
Argentinierstrasse 8/184-1, 1040 Wien, Austria
dustdar@infosys.tuwien.ac.at

Abstract. With the advent of Web services and orchestration specifications like BPEL it is possible to define workflows on an Internet-scale. In the health-care domain highly structured and well defined workflows have been specified in standard documents. To reduce the complexity of creating Web service orchestration specifications, we provide a model-driven design approach, consisting of manual and automatic transformations. Security and transaction requirements are covered additionally. The resulting Web services can be bound dynamically at run-time. Therefore, we gain the flexibility to integrate processes that are already established with specific business protocols. Parts of this approach should be applicable to other domains, too.

1 Introduction

1.1 Motivation

Healthcare workflows are distributed across several locations and executed by a large number of applications. Most scenarios in the past covered the inter-hospital execution of processes, such as exchanging patient information or diagnosis data. With the advent of Web services the Internet is capable of providing an infrastructure including a platform for patient social security cards and for supporting healthcare workflows. One way to reach this goal is to enable existing business protocols (HL7 [1], DICOM [2]) to be executed using Web services. Furthermore, the definition of Web service processes is a time-consuming and error-prone task. A semi-automatic, model-driven approach reduces the steps involved. Dynamic run-time behavior of a process enables a single Web service to bridge existing business protocols between business partners which further reduces the complexity of the solution. For our solution we focus on Web service standards that turn out to receive most support from the industry, for example BPEL [6], WSDL [5], WS-security [12], WS-transaction [13] and WS-policy [14].

1.2 Goals

In this paper we provide a model-driven approach for semi-automatic Web service descriptions with run-time binding and a Web service process. The goals we want to reach are, (i) to define a modeling process for Web service orchestration. The steps in the modeling process are supported by automatic transformations to reduce the effort that has to be put into the process; (ii) to specify the Web service orchestration in a way that it can be dynamically invoked by all applications that currently interact using established processes with specific business protocols; (iii) to integrate additional security and transaction properties of the orchestration, to satisfy requirements of real-world scenarios; and (iv) to complement this design-time process with a run-time perspective, to gain a better understanding of the execution of the orchestration.

Overall, this approach supports the implementation of Internet-scale healthcare workflows by reducing the complexity of creating Web service orchestration specifications. Although, we use an example from the healthcare domain, valuable parts of this model-driven approach should be applicable to other domains, too.

The paper is structured as follows. Section 2 introduces an example and the basic idea of our model-driven approach. Section 3 describes the modeling process in an overview and with each step in detail. Section 4 provides additional information about the run-time behavior of the Web service orchestration. Section 5 concludes the results and states topics of further research.

1.3 Related work

Our previous work covers interorganizational workflow in the medical imaging domain [4]. The paper covers the separation of a workflow layer using WSDL and BPEL, and a domain layer using DICOM and HL7. Subsequent papers then focused on Web service modeling and the mapping between BPEL activities and DICOM and HL7 messages [9, 10]. Besides our work, Artemis [26], an EU supported project, develops Semantic Web services for the healthcare domain. Its main focus is semantic mediation of services, in contrast to our process modeling oriented approach.

Furthermore, one paper [9] compares classical workflow models for medical imaging with Biztalk. This work is related to the middleware paradigm in an intranet based environment. One paper on application integration [10] helps understanding the “large picture” of medical workflows and the IHE framework but does not focus on modeling Web services. Another paper covers a model-driven approach for Web service transactions that supports more sophisticated scenarios of interaction [23].

Finally, there is work related to the medical industry and Web services standards as referenced throughout this paper. However, the focus of our paper on modeling BPEL processes based on the IHE framework is, to the best of our knowledge, not covered in the literature so far.

2 Example workflow

2.1 Overview

In the healthcare domain highly structured and well defined workflows have been specified through the HL7 and the DICOM protocol standards. Those standards have been extended with the IHE (Integrating the Healthcare Enterprise) [3] framework that defines scenarios and profiles for these standards and specifies the most common application roles and workflows in detail. This is the source for our modeling process. In other domains similar specific sources have to be identified or created. Most IHE roles used in the workflow are covered by HIS and RIS (Hospital and Radiology Information System) and PACS (Picture Archiving and Communication System) applications. They are comparable to ERP (Enterprise Resource Planning) or SCM (Supply Chain Management) applications. For a more detailed description of the domain and the capabilities of these applications refer to [9, 10].

2.2 Example

For our example we focus on a specific workflow within the IHE framework, the IHE *administrative workflow*. Figure 1 shows an overview of roles and transactions, the grey-shaded area, the *patient registration* IHE transaction, is where we dive into. The lighter-shaded area is also mentioned as it is part of the same workflow. An IHE transaction is comparable to a BPEL process (Appendix C, Table 1 [27]).

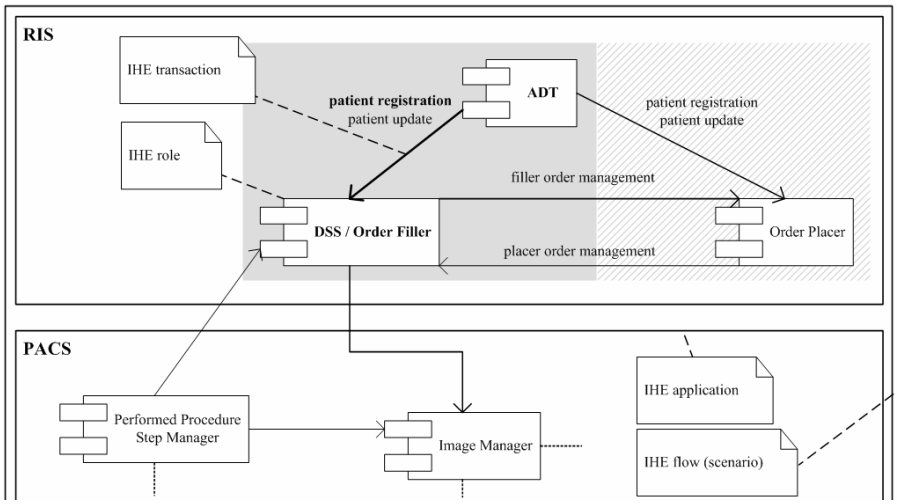

Fig. 1. IHE *administrative workflow* - focus of our example

The *patient registration* transaction is performed between two systems: *ADT* (Admission, Discharge and Transfer) and the *DSS* (Department System Scheduler). The

ADT corresponds to an administration application that provides patient data to different subsystems. The *DSS* is responsible for scheduling medical examinations. The transaction transfers patient registration information from the *ADT* to the *DSS*. The *ADT* and the *DSS* are IHE roles, an application (IHE actor) can act as several IHE roles. There is a direct relation to the BPEL partner model (Appendix C, Table 1 [27]). As shown in Figure 1, several roles and transactions are involved in a specific implementation scenario. Our model-driven approach is appropriate for this environment in general. Next, we provide an overview of the source and result of our modeling approach (see Figure 2).

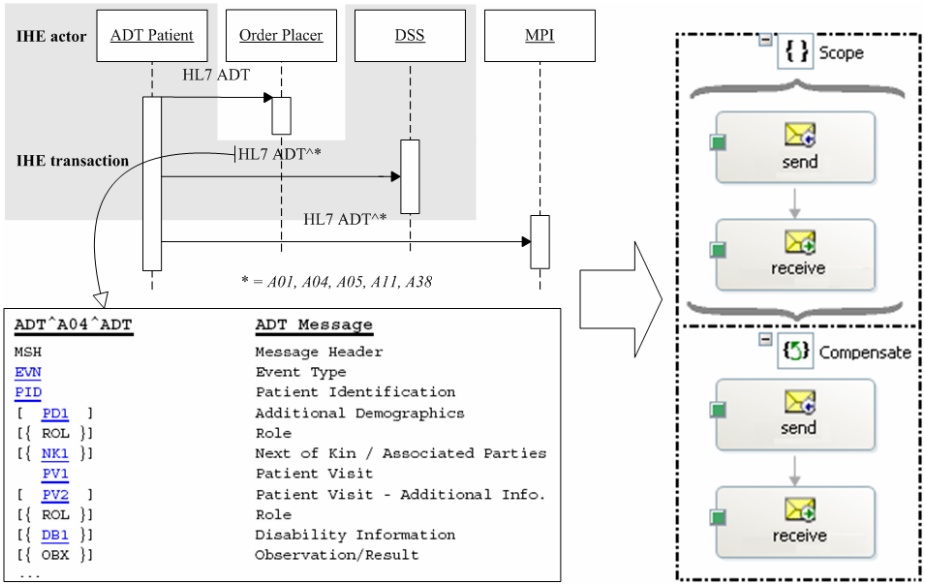

Fig. 2. source and result of our model-driven approach

The HL7, DICOM and IHE standard documents serve as the starting point for the modeling process. In our example, the two main sources are the IHE transaction UML sequence diagrams and (for our example) the HL7 message scheme definition (on the left side). The modeling result should be an executable BPEL process (on the right side). Of course, this basis cannot be mapped directly to a Web service orchestration. It provides process information, structure and data which have to be extended manually and transformed automatically in several steps. From the sequence diagram we derive the business process. This source has to be transformed and extended with orchestration flow constructs and security and transaction requirements manually. From the HL7 message we extract the structure and data for message correlation, business partner and communication port configuration. It also provides data to control the orchestration itself. In non-Internet environments, application providers implement the workflow according to the IHE framework sequence diagrams. They provide a native HL7 (over TCP/IP) interface business protocol. The outcome of our approach is to execute the processes and exchange the messages using Web services and BPEL orchestration. Section 3 describes the modeling process in detail.

2.3 Requirements of further workflows

Healthcare workflows have different requirements. A model-driven approach should be evaluated using several examples with different requirements. For example, the IHE framework contains transactions using the DICOM protocol, where large amounts of medical image data (more than 100MB per transaction) have to be transferred and, therefore, be compressed. In [9, 10] we investigated the requirements of healthcare workflows in a Web service environment. Conclusions have been considered in our approach.

3 Modeling process

In this chapter we provide the modeling process for our orchestration. We use our example, the IHE *patient registration* transaction (BPEL process). First, we provide an overview with a short description before we show the modeling steps in detail.

3.1 Process overview

The modeling steps are described throughout the sections as shown in Figure 3.

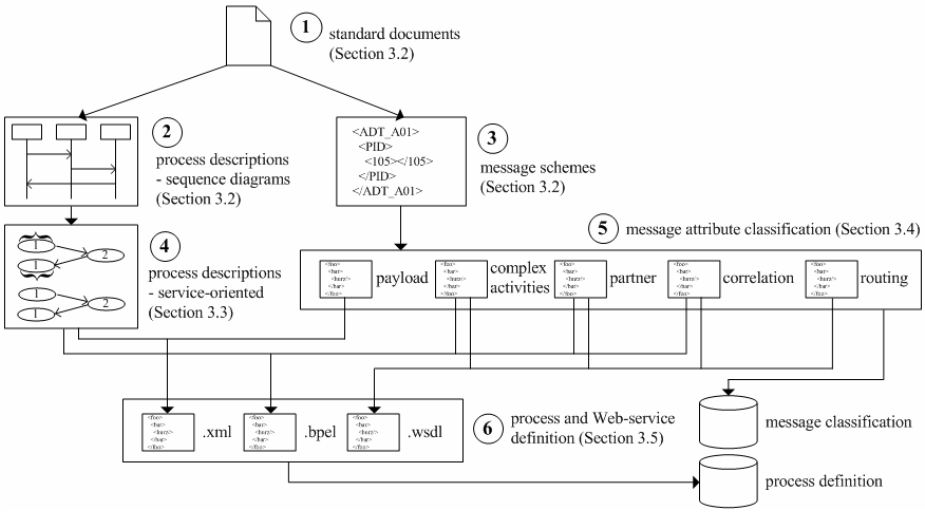

Fig. 3. Modeling process steps

The modeling process occurs at design-time. The run-time behavior of an executing BPEL process is shown in Section 4. The process starts with the available standard documents (Step 1) as shown in the introduction of the example. In Step 2 and 3 the UML sequence diagrams and HL7 message schemes are stored in a database or file-system. Step 4 converts the diagrams to a process oriented UML activity diagram and applies transaction and security concerns. Step 5, on the other hand, classifies the

message attributes. The classification covers *structured activities* (for structured activities, like BPEL *switch* statements), the business *partner* and partner-link definitions (corresponding to the sections in the BPEL process definition), *correlation* attributes (used in BPEL correlation-sets) and *routing* attributes that define the destination of the messages and are used to configure the ports in the BPEL process. The *payload* contains the whole HL7 message that is sent as an attachment to the SOAP [25] message by the Web service. Step 6 merges the information of the process and the message attributes and generates three output files, a BPEL process description, a WSDL file that defines the communication end-points and an XML containing additional security and transaction properties using WS-policy.

3.2 Step 1-3: Digital source representation

We start with a digital representation of the IHE transaction and HL7 message. Figure 4 shows the source representation of the *patient registration* transaction.

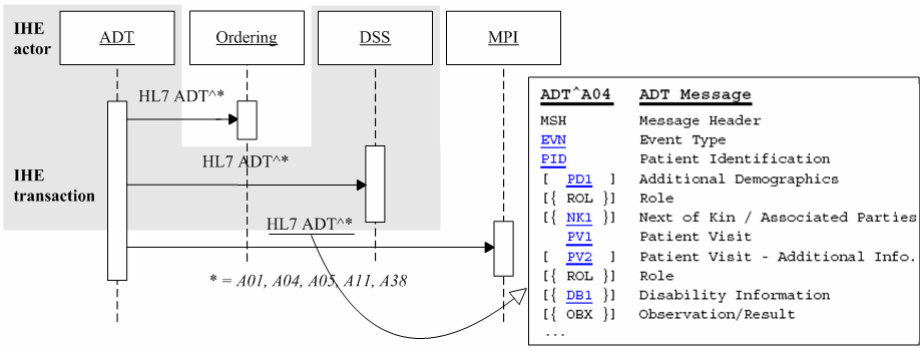

Fig. 4. IHE *patient registration* transaction and HL7 message

In general, from the sequence diagram we extract the process, from the message we extract a design-time configuration and run-time properties of the process. The information has to be digitized into a file-based or database storage.

The UML diagrams of the IHE framework are of proprietary file format (in our case Microsoft Visio). The diagrams contain several ambiguities and errors that have to be resolved. For example, the arrows represent more than one message. Furthermore, a sequence diagram is not appropriate for a BPEL process as it contains more than two business partners for which an interface should be defined. In the next section we show how an activity diagram and several adaptations solve these problems.

The HL7 message has a hierarchical format that consists of several modules (which are reused between messages) and each module consists of a set of attributes. The attributes are of specific (simple and complex) data types that can be represented in XML [11]. DICOM messages for comparison contain data and service descriptions, but nevertheless, can be broken down to data types and payload data (images, documents). Also some XML messages carry documents as attachments. However, to setup our process we are only interested in those parts of the messages that contain

information to identify partners, configure and control our process and route messages. All other data resides in the HL7 message, which is sent as an attachment of the SOAP message.

3.3 Step 4: Service-oriented process description

In this step we convert the sequence diagram into an activity diagram.

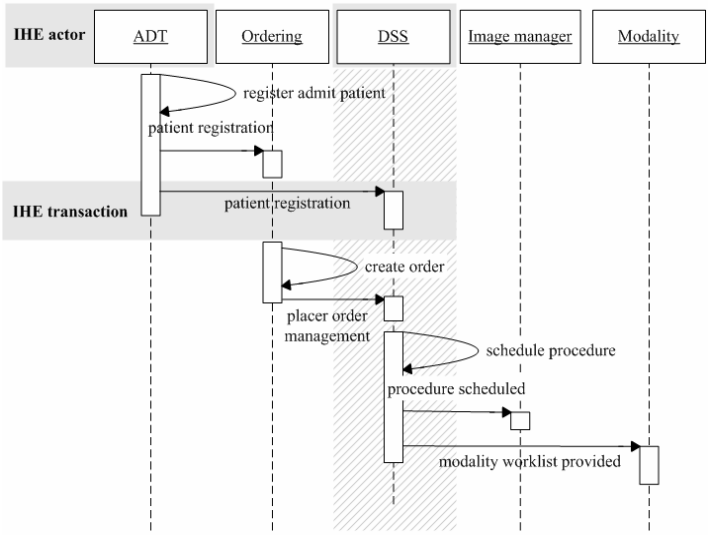

Fig. 5. IHE administrative flow

The following sub-steps are performed during the manual transformation:

- resolve errors in the standard document
- select partner to define a public process
- convert to an activity diagram and skip private activities
- apply security requirements

(repeat the next steps for each IHE transaction)

- select a specific IHE transaction
- extend the diagram to represent different control flows
- extend the diagram to represent acknowledgement messages
- apply transaction requirements

Each sub-step is described in more detail throughout the rest of this section. In contrast to the introduction the sequence diagrams of the IHE standard are provided at two levels of detail. We start with the *coarse-grained level* to perform several sub-steps of the transformation for multiple transactions at once. Figure 5 shows the sequence diagram of the IHE *administrative workflow* which is the "large-picture" where the *patient registration* transaction (grey-shaded are) is performed. From here we start with the following changes.

Substep 1: Resolve errors in the standard document: In this diagram of the IHE framework we found, that a transaction has been drawn in the wrong direction (*modality worklist provided* transaction). Manually created sources always have to be reviewed in detail.

Substep 2: Select partner to define a public process: As we model executable BPEL processes, it is necessary to select an IHE actor (BPEL partner), whose public process has to be represented. We select the *DSS* actor and skip all activities that are not sent or received by this actor.

Substep 3: Convert into activity diagram and skip private activities: Compared to the sequence diagrams, each arrow (IHE transaction) is represented with two activities, one BPEL *invoke* and *receive*. For each IHE actor a lane is generated. Furthermore, internal activities, activities that are performed by an actor on itself, are skipped, as no BPEL process related activity is necessary. The resulting diagram is shown in Figure 6.

Substep 4: Apply security requirements: Next, security requirements between business partners are defined. In [8] we defined security zones and boundaries to represent groups of applications that trust each other. The organizational trust information might be modelled using WS-Trust and stored globally in a database for all modelled processes. However, this is out of the scope of this paper. Figure 6 shows the result after these four sub-steps.

In our case, the *DSS* actor is in the same zone as several other actors but in a different one as the *ADT* actor. Therefore, we require message encryption using WS-security when executing the *register patient* transaction.

Substep 5: select a specific IHE transaction: We select the *patient registration* transaction and now focus on the sequence diagram provided by IHE on the *fine-grained level* (diagram shown in Figure 4).

Substep 6: Extend the diagram to represent different control flows: In our case the transaction consists of sending one of three HL7 messages (*ADT_A01*, *ADT_A04* and *ADT_A05*). Which message is sent depends on the class of patient which is represented as the *PatientClass* attribute within the HL7 message. This decision can be modelled in BPEL using a *switch* structured activity. It is represented in the activity diagram accordingly.

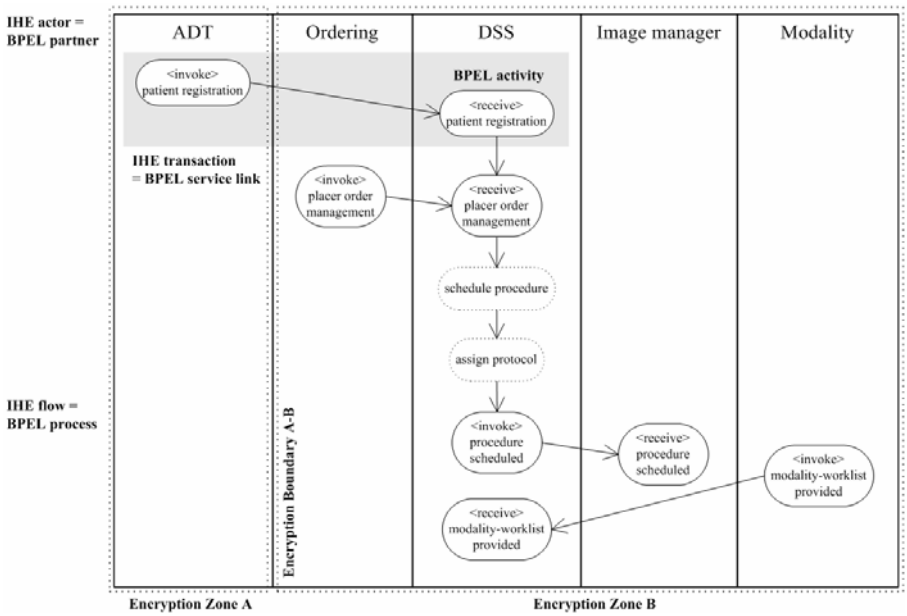

Fig. 6. Administrative flow - BPEL public process of *DSS* (Department System Scheduler)

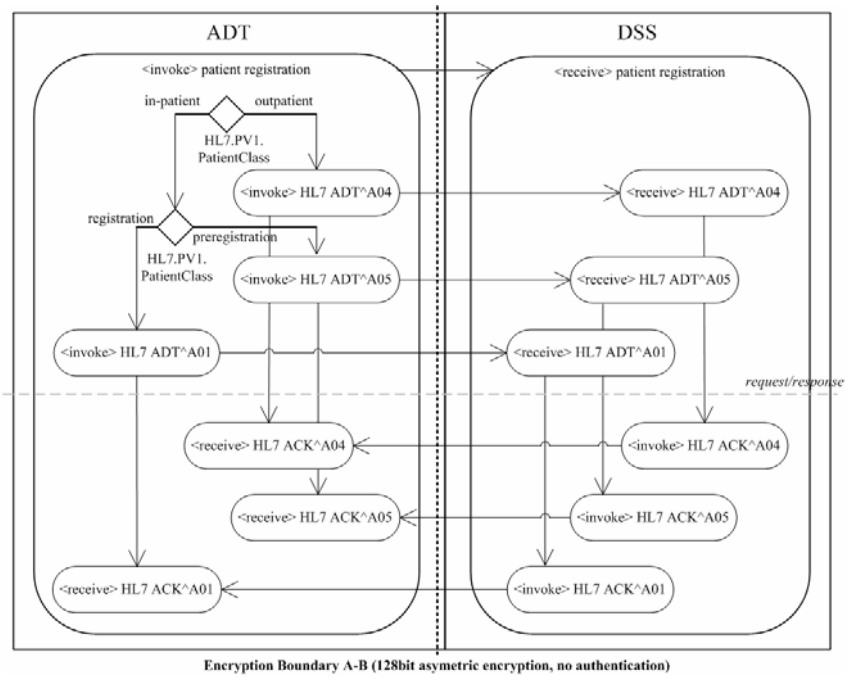

Fig. 7. *patient registration* transaction - public process

Substep 7: Extend the diagram to represent acknowledgement messages

In HL7 each message sent is followed by receiving an acknowledgment message (*ACK_A01*, *ACK_A04*, *ACK_A05*). Therefore, the diagram has to be extended to represent this behavior. DICOM uses status messages to represent similar behavior. Figure 7 shows the resulting diagram. The outer frame corresponds to the original invoke-receive pairs (compare to Figure 6), which has been extended through the substeps 6 and 7.

Substep 8: Apply transaction requirements

Referring to the requirements stated in Section 3.2 we integrate transaction requirements using compensation activities. Atomic transactions are currently not considered, although, there are several operations in the DICOM standard suited for it. We currently refer to the work presented in [23] for an extended transaction modeling approach. Figure 8 shows the resulting diagram.

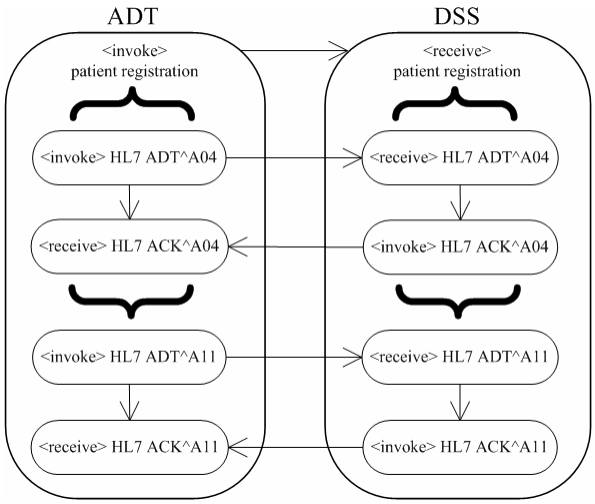

Fig. 8. Compensation-based transaction for the *patient registration* transaction

For HL7 negative acknowledge messages (*ACK_A11*) are generated in case of errors. Transactional behavior is directly expressed in the process definition, as it is part of the BPEL specification, while security parameters have to be configured for interfaces and are later on stored in the policy file (see Section 3.5.3).

3.4 Step 5: Message attribute classification

In the next step we turn to the HL7 message itself. Not all parts of the message are equally important for a process definition. We can distinguish the following five categories for message attributes (which are also valid for other business protocols):

- Class 1: attributes required for *binding* (WSDL interfaces binding information)

- Class 2: attributes required for *partner* definition (BPEL partner definitions)
- Class 3: attributes required for *complex activities* (complex BPEL activities)
- Class 4: attributes required for correlating the message (BPEL *correlation-sets*)
- Class 5: other message attributes (used by the underlying business protocol)

As stated in Section 3.2 the hierarchical structure of message attributes can be represented in XML. Related to our example we classify the HL7 ADT_A04 message as shown in Figure 9. The structure of all ADT messages is the same regarding attribute classification.

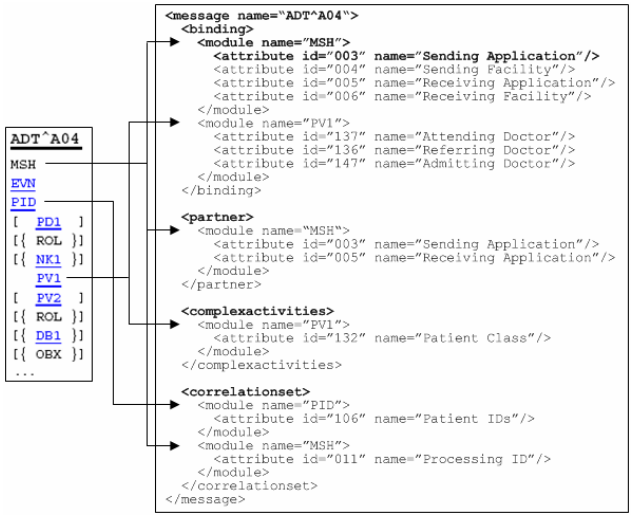

Fig. 9. Message attribute classification of HL7 ADT^A04

On the left, the original message structure is shown. Through an analysis of the attribute descriptions in the HL7 standard and the IHE framework the attributes listed on the right side have been selected for each class respectively. An XML scheme file for this structure can be found in Appendix A [27]. The result of this step is a classification file for each message used in the process. The file is required in the next step of orchestration definition and during run-time execution (Section 4) for message parsing. It is stored together with the message scheme files in a database.

The Class 1-3 attributes are the same for every business partner. For Class 4 it is possible that the receiving partner requires different attributes for BPEL complex activities. Therefore the analysis has to take into account the processing of all partners involved in the transaction. In our case no extension to the definition is necessary. A further conclusion is that Class 1-4 attributes have to be part of the SOAP message and Class 5 attributes reside in the attachment. However, the security requirements (see previous section) are always defined for the whole SOAP message (the parameters and the attachments).

3.5 Step 6: Orchestration definition

The orchestration definition step is split into three sub-steps which are described in detail throughout this section. We present parts of the files for illustration, a complete listing can be found in Appendix B [27].

3.5.1 BPEL process definition

The first sub-step is the definition of the BPEL process. Figure 10 points out which parts of the model contribute to the content of the file and how the file is structured.

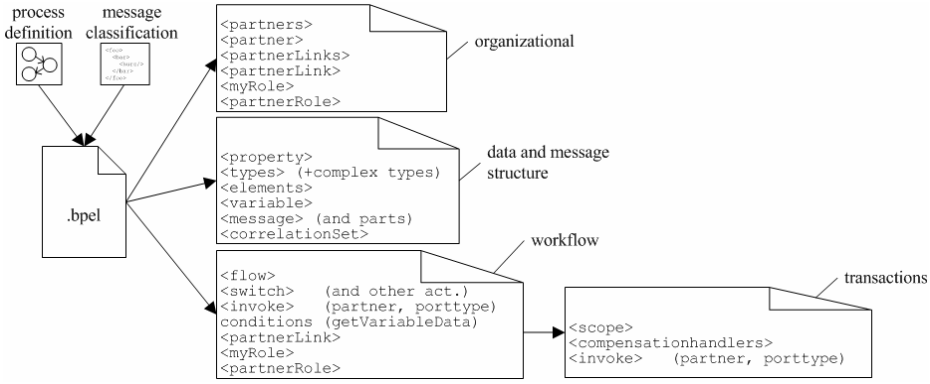

Fig. 10. BPEL process definition

One part of the BPEL definition is created by converting the activity diagram into BPEL constructs. Several conversions are performed to create parts of the *flow* section of the BPEL file from the process (see Table 2 in Appendix C [27]). The second part is extracted from the message attribute classifications partner and binding sections. Therefore, the table also lists the mapping between message attributes and BPEL sections and tags. Conversions of IHE activity diagrams into BPEL flows have been covered in detail in an earlier paper (see [7]). There is also a paper that covers UML conversions in general [22]. Of special interest here are the components for the partner definition, which are directly derived from the IHE roles and transactions. Those names are generic and allow, together with a run-time generation of WSDL interfaces (see next section), a dynamic model of BPEL process execution (see Section 4). Finally, XPath [17] expressions are generated using a lookup in the message attribute classifications *complexactivities* section. In our example the value for the PatientClass variable is “HL7_A04_TYPE/PV1-132”, if an A04 type message is sent by the application.

3.5.2 WSDL interface definition

The second part is the WSDL interface definition. Here we have to distinguish between design-time and run-time operations. During design-time the portTypes, which are required by the BPEL process, are defined. As Figure 11 shows, message attributes are used here.

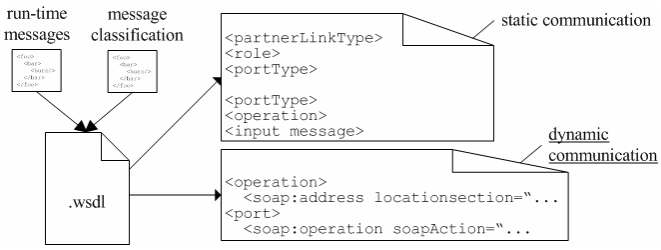

Fig. 11. WSDL process definition as design-time

Appendix C, Table 3 [27] contains the mapping between those elements and the WSDL content. We want to focus on the dynamic binding part in the WSDL definition, which contains the information about the communication endpoint. Selection of a specific endpoint can be performed dynamically during run-time by configuring the WSDL file. This can be done by the Web service or the BPEL engine, if it supports dynamic endpoint configuration. The dynamic parts of the WSDL file are the *binding* and *port* sections. The definition consists of a base URI and an extension that references the specific service.

- WSDL section: operation, element: soap:address, attribute: location
- WSDL section: port, soap:operation, attribute: soapAction

As each business partner always uses the same generic Web service, it is only necessary to store a mapping for one destination URI. The source for the URI mappings can be any value of the HL7 and DICOM message attributes that have been classified in the *bindings* section of the message classification document. During run-time the values are extracted from the messages. Then the mapped URI is calculated and the WSDL file is configured for the required endpoint. In the next step, BPEL processes can perform activities with the dynamically added business partner using the newly configured port.

3.5.3 Policy definitions

The third part is the policy definition. Security and attachment requirements have to be converted to WS-security and proprietary constructs (see Figure 12).

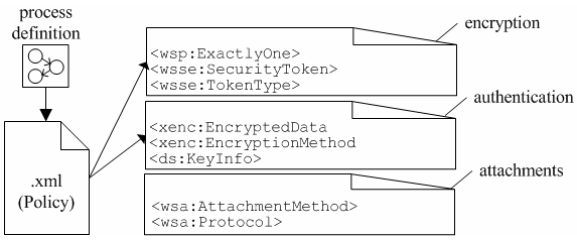

Fig. 12. WS-policy definition

For the WS-policy definition we use the WS-SecurityPolicy [21], WS-Encryption [19] and WS-Signature [20] standards. For attachments we defined a proprietary

policy description. The attachment requirement is constant, all messages contain attachments. Currently, the DIME [15] standard is specified but is supposed to be superseded by MTOM [16] soon.

For the security part the process diagram has to be parsed and for all security boundaries the properties for encryption and authentication have to be applied (Appendix C, Table 4 [27]). The security credentials are provided at run-time (Section 4).

4 BPEL process at run-time

We split the run-time activities into 3 phases. Figure 13 shows phase 1.

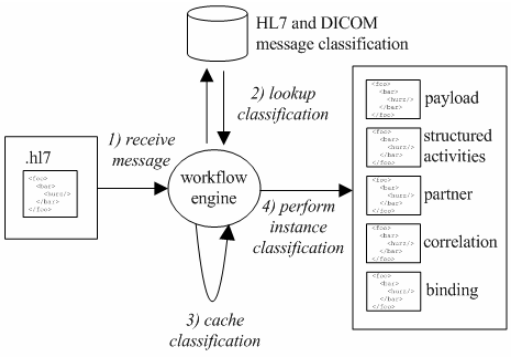

Fig. 13. Run-time phase 1: receiving a message

First, a component (that we call workflow engine) receives a HL7 message from an application using the business protocol. Then it has to lookup and cache the message classification and perform a classification of message attributes on the received message instance. According to the XSL scheme (Appendix A [27]), classification values are extracted and stored. The next phases 2 and 3 are shown in Figure 14.

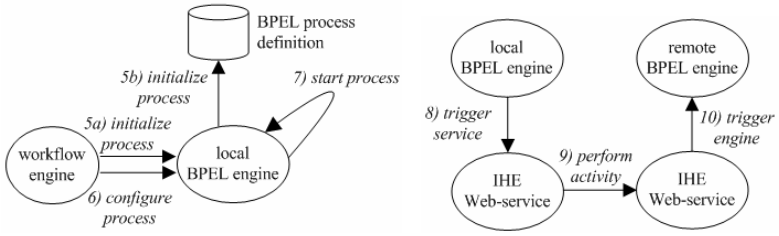

Fig. 14. Run-time phase 2 and 3: start and execute BPEL process

In Phase 2, the BPEL process is initialized (if it is a start message of a BPEL process) using its process definition and configured with the values of the message classification. For each initial and subsequent message the dynamic ports are bound according to the values of the *binding* section in the message classification. Next, depending on the capabilities of the BPEL engine, the security requirements are selected

by the engine (in Phase 2) or the Web service (in Phase 3) and are used to configure secure ports. Biztalk Server 2004 [18] for example supports secure ports with an additional Web service adapter. The security credentials (for example asymmetric keys) depend on the communicating applications. Information from the *partner* section of the message classification is used to lookup credentials in a database (via UDDI [24] for example). The credentials are inserted into the SOAP message before the Web service calls the partner. The same steps occur on the receiving Web service. As the sending port is now identified by the initiating application, a lookup to a database can be performed to decode and validate the SOAP message, before it is passed to the BPEL engine for workflow processing. Additionally, both partners have to insert the business protocol message into the attachment part of the SOAP message.

5 Conclusions and future work

In this paper we presented a model-driven approach to define Web service orchestration in the healthcare domain. We were able to meet our goals, to define a design-time modeling process. Through semi-automatic modeling steps we produced results out of standard documents of the business protocols and applied security and transaction requirements. We created several artifacts: a BPEL file for the process definition, a WSDL file with a dynamic port configuration, a WS-policy file containing security and attachment requirements. We have also shown the run-time behavior of the suggested solution using the artifacts produced during modeling. The benefits stated initially, the reduction of complexity and required effort can be concluded from our work. Further, the modeling steps *public process definition* and *message classification* can be applied to other domains, too. We find it especially noticeable, that it is possible to execute several BPEL processes using one generic Web service. For future work we plan to extend our current prototype implementation to encompass a model-driven toolset. Currently, several steps need more standardization before we can proceed. Furthermore, atomic transaction requirements should be investigated and an executable example transaction should proof our approach.

6 References

1. HL7 Organization: Health Level 7, http://www.hl7.org (2000)
2. NEMA and Global Engineering Group: DICOM 3 Standard, http://www.nema.org (1998)
3. Radiological Society of North America: IHE Technical Framework 1.1, http://www.rsna.org /IHE/index.shtml (2003)
4. Anzböck, R., Dustdar, S.: Interorganizational Workflow in the Medical Imaging Domain. Proceedings of the 5th International Conference on Enterprise Information Systems (ICEIS), Angers, France, Kluwer Academic Publishers (2003)
5. W3C: Web services Description Language 1.1, http://www.w3.org/TR/wsdl.html (2001)
6. BEA Systems, IBM, Microsoft, SAP AG and Siebel Systems: Business Process Execution Language for Web services version 1.1, http://www-128.ibm.com/developerworks/library/ specification/ws-bpel/ (2003)

7. Anzböck, R., Dustdar, S.: Modeling Medical Web services, BPM 2004 - Conference on Business Process Management (2004), Springer LNCS 3080, pp. 49 – 65.
8. Anzböck, R., Dustdar, S.: Modeling and Implementing Medical Web services. Data and Knowledge Engineering, Elsevier, forthcoming (2005)
9. Von Berg, et.al.: Business Process Integration for Distributed Applications in Radiology, Philips Research; Hamburg, Germany (2001)
10. From PACS to integrated EMR: Osman Ratiba, Michael Swiernik, J. Michael McCoy, Computerized Medical Imaging and Graphics 27 Pages 207–215 (2003)
11. HL7 Organization: HL7 XML encoding scheme: http://www.hl7.org/ (2003)
12. BEA, IBM, Microsoft: Web Services Security (WS-Security), www-106.ibm.com/developerworks/webservices/library/ws-secure/ (2002)
13. BEA, IBM, Microsoft: Web Services Transactions (WS-Transactions), http://www.ibm.com/developerworks/library/ws-transpec/ (2002)
14. BEA, IBM, Microsoft, SAP, Sonic, VeriSign: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnglobspec/html/WS-policy.asp (2004)
15. Microsoft: Direct Internet Message Encapsulation (DIME), http://msdn.microsoft.com/library/en-us/dnglobspec/html/draft-nielsen-dime-02.txt (2002)
16. BEA, Canon, IBM, Microsoft: http://www.w3.org/TR/soap12-mtom (2005)
17. W3C: XPath, www.w3.org/TR/xpath (1999)
18. Microsoft Biztalk Server 2004: Microsoft Corporation, www.mirosoft.com (2004)
19. XML-Encryption: W3C Working Draft, "XML Encryption Syntax and Processing,", http://www.w3.org/TR/xmlenc-core/ (2002)
20. XML-Signature: W3C Proposed Recommendation, "XML Signature Syntax and Processing,", http://www.w3.org/TR/2001/PR-xmldsig-core-20010820 (2001)
21. Microsoft, IBM, Verisign, RSA Security: WS-SecurityPolicy, http://msdn.microsoft.com/webservices/default.aspx?pull=/library/en-us/dnglobspec/html/WS-Securitypolicy.asp (2002)
22. IBM: From UML to BPEL, http://www.ibm.com/developerworks/ webservices/library/ws-uml2bpel/ (2003)
23. Schmit, B.A., Dustdar, S. (2005). Model-driven Development of Web service Transactions, XML4BPM 2005 - XML for Business Process Management Workshop, 11th GI Konferenz Business, Technologie, und Web (BTW 2005), 1 March 2005, Karlsruhe, Germany.
24. IBM/Microsoft/SAP, et.al.: UDDI 3.0.2, http://www.oasisopen.org/specs/index.php#uddiv3 (2005)
25. W3C: SOAP Version 1.2, http://www.w3.org/TR/soap12-part1/ (2003)
26. Dogac, A., et.al.: Artemis: Deploying semantically enriched Web services in the healthcare domain, Elsevier, Information Systems, (2005), Article in Press
27. Appendix see http://www.infosys.tuwien.ac.at/Staff/sd/papers/BPM2005Appendix.pdf