Available online at www.sciencedirect.com
SCIENCE DIRECT®
Data & Knowledge Engineering 55 (2005) 203–236
DATA & KNOWLEDGE ENGINEERING
www.elsevier.com/locate/datak

Modeling and implementing medical Web services

Rainer Anzböck [a], Schahram Dustdar [b,*]

[a] *DATA Corporation, Invalidenstrasse 5-7/10, 1030 Wien, Austria*
[b] *Distributed Systems Group, Information Systems Institute, Vienna University of Technology, Argentinierstrasse 8/184-1, 1040 Wien, Austria*

Received 25 January 2005; received in revised form 25 January 2005; accepted 24 March 2005
Available online 26 April 2005

Abstract

On the one hand, Web services are increasingly gaining attention. Standardization efforts have improved their stability and range of applications. Composition and coordination techniques for Web services enable an application integration effort beyond loosely coupled systems. On the other hand, medical Web services are covered by the DICOM and HL7 communication protocols and are profiled by the IHE (Integrating the Healthcare Enterprise) technical framework. Standardization is more extensive, most workflows are well defined, and integration is tighter than in most other domains. Nevertheless, so far standardization focused on conventional workflow systems. In an Internet-based medical environment with high security standards, communication is strongly restricted and conventional systems fail to deliver. This paper proposes a modeling process for medical Web services. The IHE patient administration process flow serves as a well defined example. Furthermore, the paper defines requirements of a Web service based middleware for the execution of medical Web services by investigating currently relevant Web service protocols. The technique should enable building medical applications for Internet-based workflow execution. Finally, Biztalk 2004 is used to evaluate the implementation of the analyzed sample functionality.

Keywords: Web services; Medical Web services; Workflow; Modeling

* Corresponding author. Tel.: +43 1 58801 18414; fax: +43 1 58801 18491.
E-mail addresses: ar@data.at (R. Anzböck), dustdar@infosys.tuwien.ac.at (S. Dustdar).

0169-023X/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.datak.2005.03.009

1. Introduction

With recent work in the field of workflows it is possible to define more flexible business models than in traditional workflows based on the Workflow reference model (WFMC) [1]. With the standardization of coordination, composition, transaction, and security for Web services, a new implementation method for Web service based scenarios is available. Especially the medical services domain is in a permanent evolution. Its workflows are complex and highly structured and a standardization of communication protocols has been covered by HL7 [2], DICOM [3], and the IHE framework [4]. Further standardization processes for health informatics are enforced by the European Union with the CEN/TC 251 work program [5].

One goal of this paper is to outline a modeling process for medical Web services. We start by initially introducing the medical services domain, define the requirements subsequently, and conclude how to model such services based on the IHE *administrative process flow* sample in 5 steps. The modeling process should be refined in further research and result in a guideline or semi-automatic process for defining medical Web services' workflows using Web service based composition.

Another goal of our paper is to show how recent work on protocols of the Web service stack and standardization efforts in the medical services domain (the IHE framework in particular) help to solve application integration. First, we provide an introduction to the medical services domain. Then, we define an implementation scenario for a Web service based medical middleware (medical Web services). To outline requirements of a Web service oriented approach, we use a specific example, the *administrative process flow*. When going into detail, we further focus on two IHE transactions, the *patient registration* and *retrieve image* transactions, as they are representative for HL7 and DICOM communication.

A third goal is the discussion of requirements for modeling medical Web services. Related to the example introduced, we discuss Web service concepts and standards like SOAP [6], WSDL [7], WS-Coordination [8], WS-Transaction [9], WS-Security [10] and many more. From there we focus on the composition of Web services using BPEL [11] and define requirements to model IHE transactions as medical Web services. In a last step, we evaluate a sample implementation of the *patient registration* transaction using Microsoft Biztalk 2004 [70] as the workflow management software. We show possibilities and limitations of the platform as an example for an implementation framework for Web service composition. Finally, we conclude the results and provide topics for future work.

To summarize, our paper (i) suggests a modeling process for the *IHE administrative process flow* example and outlines implications for a general modeling process and a Web service middleware to implement medical Web services, (ii) introduces the medical services domain and the *administrative process flow*, (iii) defines requirements of a modeling process based on current Web service stack standards and (iv) evaluates the process using Biztalk 2004.

The paper is structured as follows. Section 2 introduces medical information systems, communication protocols and the IHE technical framework. Section 3 provides requirements of a modeling process for services of the IHE *administrative process flow*. Section 4 outlines a modeling process of medical Web services. Section 5 guides through an implementation of the sample workflow based on Biztalk 2004. Section 6 concludes the results and outlines further work.

2. Medical Web services

In this section we briefly introduce medical information systems, communication protocols and the IHE framework to better understand the sample transactions and modeling process, where we have to use domain-related terms extensively.

2.1. Medical information systems

Three types of medical information systems, the HIS (Hospital Information System), the RIS (Radiology Information System), and the PACS (Picture Achieving and Communication System) are the backbone of current information systems in the hospital and medical Web services environment. They are comparable to ERP (Enterprise Resource Planning) or SCM (Supply Chain Management) in business organizations. The HIS is an enterprise-wide system used for administrative services such as patient and visit management, operation planning, billing, amongst others. The RIS is a management system for medical imaging facilities (radiologists) and covering applications including patient registration, examination scheduling and control, report generation and transcription, speech recognition. As can be concluded, both systems have overlapping services to fulfill: one on an enterprise the other on a department level. The second main software system category in medical Web services is called PACS and is responsible for all image management services. It transfers patient data to examination facilities (modalities), announces finished procedures, and stores, prints, burns CDs, archives or forwards the generated image data.

These software systems are often integrated as departmental services for a larger hospital environment and sometimes spread to several locations. Because of their special storage, network and processing performance requirements RIS and PACS [55–57,71] systems are very important departmental services. Company related information on these systems can be found in [12–18], more theoretical work in [19–21].

2.2. Medical communication standards

The most relevant protocol standards for these services are HL7 for the RIS and DICOM for the PACS. PACS and RIS both implement a workflow model and cover implementations of the standard. Both systems have to be tightly integrated to perform services efficiently. The DICOM standard covers Client/Server communications used to exchange Patient and Examination information. The standard covers objects like patients, visits, medical procedures, images, films, printers, and examination modalities. Additionally, notifications, data query, and exchange services based on these objects are defined. The HL7 standard is used for data exchange between different healthcare providers and is more suited for non-radiological institutions. Some functionality overlaps with DICOM, for example, the scheduling process and the patient and result management. Other functionality, such as the exchange of image data, is not part of HL7. More detailed information on HL7 can be found in [2] and on DICOM in [3,22,23]. Besides these protocols, additional standards like CEN 251 [5] exist. Ambitions to converge these standards by using a common framework have led to the definition of IHE [4].

2.3. A medical workflow framework

The IHE technical framework has been defined to extend the enterprise application integration to a level of scenario-based interaction. Over the years software products implemented the DICOM and HL7 standards by their own interpretation. This led to a situation of incompatibility and a lot of effort has to be put into integration aspects. The framework defines usage-scenarios with the goal that products which conform to the framework can be integrated seamlessly.

IHE defines (workflow) transactions between applications by profiling DICOM and HL7 operations. Messages (domain activities) are selected and put into sequences to implement real-world scenarios. Additionally, flows (workflow services) are defined that correspond to a set of related transactions performed by different actors (administration application, image archive, etc.). Applications may perform the role of one or more such actors in one or more of these flows. To claim IHE conformity for a role in a workflow, a required set of flows and transactions has to be implemented. Further relevant IHE terms used throughout the framework are subsequently introduced during the process definition in further sections.

IHE conformant applications can be integrated more tightly than applications in other domains. Nevertheless, integration based on this framework is currently reached using traditional workflow models in Intranet-based environments. An Internet-based network infrastructure, as currently common in most environments, restricts interorganizational workflow [24] integration. In a real world scenario integrators have to deal with applications in an Intranet and Internet environment. Workflow items like patient and image data are exchanged within and across organizational boundaries. Fig. 1 shows an example of such an environment.

An Intranet-based environment consists of conventional HL7 and DICOM communication over a secure and reliable transport. Additionally, the IHE framework provides a solid foundation for defining medical workflows in this environment. Current solutions integrate applications based on conventional middleware. For example, gateways, acquisition modalities and patient registration

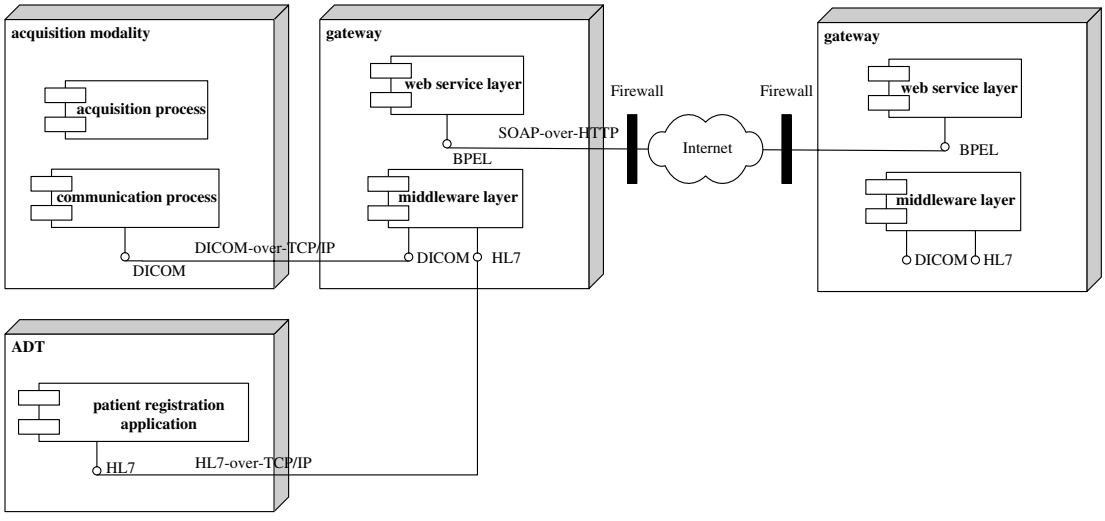

Fig. 1. Mixed Intranet/Internet environment for medical Web services.

applications are directly connected by their middleware layers. In contrast, we have to deal with interorganizational workflows, which are executed between nodes distributed over the Internet.

The gateways mentioned have two different responsibilities. On one hand, they implement IHE conformant Web service based workflow models for medical Web services. On the other hand, they enable internal nodes to participate in IHE conformant workflows, to attach their messages to XML workflow messages and to apply security and transaction support. In this paper we primarily focus on the first functionality.

This scenario is beneficial for many reasons, like exchange of patient information which results in a reduced number of examinations, load balancing work between specialized physicians, etc. Through the standardization process related to the Web service stack [25] it is feasible to suggest a workflow infrastructure based on a separate layer that meets the requirements of an Internet environment on one hand and supports standardization efforts of the medical industry, as outlined above, on the other.

Related to Web services, we have to consider the following aspects. First, we have to provide a transport mechanism, where SOAP-over-HTTP communication is a reasonable option. Next, we have to meet reliability and security requirements with additions like WS-Security [10], WS-ReliableMessaging [26] and others. To model workflows in a service-oriented computing (SOC) environment a composition language like BPEL is required. Furthermore, transactional behavior is beneficial for the quality of the business processes. BPEL prefers the use of WS-Transaction [9], which we will focus on, when defining service modeling requirements. To summarize the aspects that have to be discussed when modeling medical Web services, we find

- a high degree of vertical standardization through DICOM, HL7 and IHE,
- currently implemented systems based on conventional middleware,
- lack of interorganizational workflow support as a common problem,
- no current Web service based approach which tries to fill this gap.

Therefore, we suggest a Web service based workflow model that implements IHE conformant transactions to provide medical Web services functionality in a mixed Intranet/Internet environment.

2.4. Sample transactions

We choose two sample transactions from the IHE framework that cover relevant infrastructure, security, and transaction requirements. As the first sample we choose the IHE *patient registration* transaction. This task is expected to be easily established, but several participating actors performing secure communication and tight transactional behavior require a detailed study for the modeling process and the implementation. Fig. 2 provides an overview of the transaction.

The *ADT* corresponds to an administration application that provides patient data to different subsystems. The *Order Placer* is responsible for examination planning and the *Department System Scheduler* for scheduling of examinations. The *MPI* is responsible for enterprise wide patient identification. Different HL7 ADT messages are exchanged, they also contain transaction compensation messages and acknowledgements.

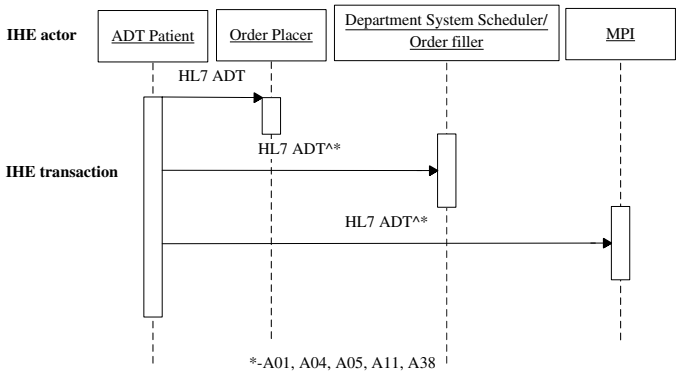

Fig. 2. IHE *patient registration* transaction.

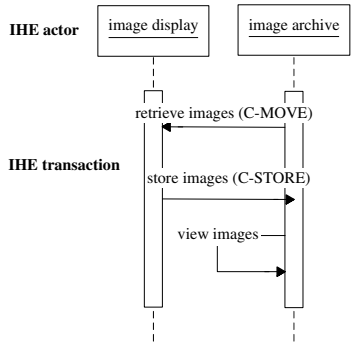

Fig. 3. IHE *retrieve images* transaction.

The second sample is the transfer (retrieval) of image data between two sites, which corresponds to the IHE *retrieve images* transaction as shown in Fig. 3.

The *Image Display* application supports the radiologist in performing the diagnosis on the patient. The *Image Archive* centrally stores image data of several examination modalities. A similar model can be found when integrating two image archives. The operation performed is a DICOM C-Store that transfers images. One transaction contains several 100 MB of data, which requires handling of attachments for Web service modeling and special techniques for image compression for an implementation.

2.5. Related work

Information related to medical Web services can be found in the corresponding standardization documents for HL7, DICOM and IHE. In our previous work, a discussion of an interorganizational workflow in the medical imaging domain can be found in [27]. A first approach of Web service definition and middleware design for the medical imaging domain can be found in [28]. The paper covers the separation of the workflow layer using WSDL [7] and BPEL [11], and the domain layer using DICOM and HL7. Additionally, it performs a mapping between BPEL activities

and DICOM and HL7 messages. An earlier paper on modeling medical Web services can be found in [60].

Besides our work, one paper [58] compares classical workflow models for medical imaging with Biztalk. This work is related to the middleware paradigm in an Intranet-based environment and does not take into account a mixed environment (see Fig. 1). However, it is architecture, infrastructure and modeling related and helps understanding medical services requirements. Another paper on KIS/RIS/PACS integration [71] helps understanding the "large picture" of medical services and the IHE framework but does not focus on modelling services based on Web service technology.

Several other papers focus on compression technologies and one especially on the transfer of compressed images over http [59]. The paper points out an alternative Internet-based communication pattern for medical imaging data, but it does not cover underlying workflows. Finally, [61] provides a different approach for a workflow security infrastructure that is also used for medical services workflows. In [61] the approach focuses on the subset of security requirements discussed in this paper and serves a promising alternative to satisfy those requirements for medical services.

Furthermore, there is work related to the medical industry and Web services standards as referenced throughout this paper. As one can see, a lot of work is being done on the integration of medical services. However, the focus of our paper on mapping IHE transactions on BPEL processes is, to the best of our knowledge, not covered in the literature so far.

3. Requirements for medical Web service modeling

In this section we cover requirements that have to be met when modeling medical Web services. We discuss impact and usefulness of current Web service stack protocols and its alternatives. Furthermore, we outline the relationship of HL7, DICOM and IHE concepts to Web service modeling constructs. We proceed from the lower-level protocols including encoding, transfer, and other, to the higher-level protocols for orchestration, and binding. Choosing BPEL [11] as the modeling language for medical Web services (see Section 3.7) has several implications for the lower levels of the Web service stack (Sections 3.1–3.6).

3.1. HL7 and DICOM encoding

When implementing medical Web services using Web service technology, we have to consider transferring HL7 and DICOM messages using XML and SOAP. One solution is a conversion of messages and binary data into XML. Another, more advantageous approach, is to simply attach original messages to SOAP messages and to only use identifiers and other attributes required for a proper workflow execution within the SOAP message. A third approach is to separate workflow and domain communication, with the disadvantage of an additional communication channel inappropriate for a firewall based Internet environment [28].

In this paper we focus on the first two approaches. For most cases HL7 and for some cases DICOM consist of small amounts of data that can be translated into XML messages. Because both standards define binary data types an encoding (like base64) has to be implemented. However, at least DICOM services, that transfer images, contain up to several 100MB of data.

Additionally, DICOM requires every single image transfer to be acknowledged. For the transfer of these messages we have to use attachments for the original messages and integrate domain identifier into the SOAP message. The image data can be compressed using lossless JPEG 2000 [66] compression. Compression of an average DICOM image lasts less than 1 s and can therefore be neglected. A study [65] performed on sample data shows that the transfer of an image study takes about 4 min using 1 MBit DSL lines.

For attachments, several techniques are available like WS-Attachments [29] based on DIME [30] or SOAP Messages with Attachments (SwA) as described in [31]. The SOAP 1.2 [6] specification supports base64binary encoding [32] of data and is currently evolving as the standard mechanism for transferring binary data as it does not require additional protocol or message parsers. Furthermore, security as in WS-Security can be applied on binary data and attachments too.

Because current composition languages like BPEL lack support for attachment requirements, a policy has to be defined using WS-Policy [63] and a technique similar to WS-SecurityPolicy [64].

3.2. Data and service identification

First, a clear identification of messages and data items is required. A necessary similarity between the HL7 and DICOM protocols is that they contain message identifiers (message ID for HL7 and association ID for DICOM). Furthermore, the data exchanged is identified by system wide identifiers (patient ID, visit ID, image ID). Furthermore, HL7 supports own and foreign IDs because it supports a two-level hierarchy of hospital-wide and departmental systems. Additionally, DICOM objects and HL7 messages use different definitions and identifiers for data items.

Fortunately, IHE chooses the more specific protocol for a given situation. Furthermore, it defines a mapping between identifiers used in HL7 and DICOM and describes usage conventions to provide interoperability of the standards. The standardization effort lets us easily select the message segment IDs (HL7) or object modules UIDs (DICOM) suggested by IHE in each modeled IHE transaction. Related to our example in Section 3, the *patient registration* transaction messages are identified by the PID-3 (Patient identifier list) and the PV1-19 (Visit number) HL7 segment attributes. The DICOM modality worklist service uses patient UIDs, procedure IDs (for examination), study instance UIDs (for images) and an accession number and admission id (for visits) for identification.

For service identification the unique IHE transaction name, e.g., patient registration can be used. This identification is required by coordination and registration protocols as described in the next sections.

3.3. Web service coordination

When using Web services the coordination of business partners is required for distributed activities. Currently, the main purposes of coordination protocols like WS-Coordination [8,33] or other approaches [34–36] are reliable messaging, transactions, and security. For medical Web services, business partners are correlated by IHE transactions. Each of these transactions might be executed between two participants requiring transaction or security services. It has to be stated, that not every IHE actor might be a separate application. Therefore, participants are normally not 1:1 related to an actor. However, the IHE actor's name perfectly expresses the role in an IHE transaction.

To support coordination protocols, unique identifiers are required. These identifiers are used by coordinators to define a coordination-context for the participants. As stated above, IDs for messages and transactions can be derived from the standards. Nevertheless, process instances that register coordination-contexts might use the same messages and transactions during their execution and are improper in this manner. A unique ID generator must be used instead. To coordinate service instances, information about used ports (service endpoint) can be extracted from the WSDL definition. Furthermore, specific roles, like master or slave in a 2PC transaction, might be required by the coordinator. However, BPEL uses a different transaction mechanism based on compensation as outlined in the next section. For security purposes, service participants can define a security context. As for transactions, unique identifiers are required and have to be generated. For reliable messaging services, like WS-ReliableMessaging, there are additional message sequence numbers, which have to be generated by the middleware like context identifiers. Furthermore, medical Web services require delivery semantics of *ExactlyOnce* and *InOrder*, because the IHE framework only mentions messages delivered accordingly. The behavior for messages that are out of sequence is undefined. For example, the Oracle (Collaxa) BPEL Server [37] product contains support for reliable messaging in a delivery service module. Furthermore, it uses WS-Addressing [38] to handle the correlation of asynchronous messages.

3.4. Web service transactions

As can be seen from the IHE framework definition, not all medical Web services require transactional behavior. Often, as frequently found in information systems, read-only operations are performed. On the other hand, we have to face medical Web services that perform transactions which span several business partners performing a lot of processing and using even different communication pattern (HL7 uses message exchange, DICOM uses client/server communication). The previously introduced IHE patient administration transaction serves as an example for an operation with very tight transactional requirements, where data consistency between different actors is crucial.

Transaction protocols are used to increase the quality of a Web service based business process to the standards already provided by conventional middleware. Currently the most important standards are WS-Transaction and more recent but not yet widely used WS-TransactionManagement [39]. In general, there are different transaction models for direct, queued and compensation-based transaction processing [40]. While direct processing is useful for short lived transactions and compensation-based for long lasting, queued processing lays in-between. In compensation-based transaction processing, compensating actions are executed to ''undo'' the effects of actions that have been successfully completed [11]. More information on Web service transactions can be found in [41,42].

DICOM and HL7 basically do not specify any transactional behavior. The application logic takes care that, for example, payments are not booked twice. Furthermore, the IHE framework defines a compensation mechanism for transactions. This supports the decision of using BPEL, which provides compensation-based transactions. With the introduction of an IHE based Web service middleware it is feasible to provide transaction services. Furthermore, as their name suggests, IHE transactions provide a granularity of activities useable for a transaction context. To implement a compensation-based model, compensation actions for IHE transactions have to be

defined. A modeling process should provide a guideline to decide transactional behavior based on the operations executed in the IHE transaction.

As an example for compensation-based transaction processing, the *patient registration* transaction uses a HL7 ADT^A01 or A04 message to register a patient. In case of an error in the sending application, the registration process has to be undone with the A11 cancel message. If a patient is pre-registered (A05) the A38 cancel message is used. We provide the modeling issues of this example in Section 5. As a second example, the *retrieve image* transaction is read-only and therefore has no compensation activity. Models like the Direct Transaction Processing using the 2-phase commit (2PC) protocol and the Queued Transaction Processing used in queue-based middleware systems are currently inappropriate for the modeling of BPEL processes. For example, the Oracle (Collaxa) BPEL Server [37] contains support for WS-Transactions and executes compensation activities defined in the BPEL workflow model.

Finally, it is reasonable to use the same granularity of an IHE transaction as in WS-Transaction.

3.5. Web service security

Several requirements for security e.g., [47] have to be met when modeling medical Web services, because the data transferred is often highly confidential. For Internet-based infrastructures as outlined in Fig. 1 existing standards in the medical industry [2–5,43] require strong encryption with a minimal key length of 128bit and authentication based on asymmetric keys. IHE defines a Secure Node actor that has to be implemented by secure nodes. A collection of secure nodes establish a secure domain where strong authentication is performed before executing any IHE transaction. Each Secure Node needs its own certificate, and must be configurable to enter a list with trusted certificates. An IHE Secure Node may be part of multiple secure domains [4]. The options that we can choose from can be seen in Fig. 4.

The diagram is a simplified view of the OSI reference model [62]. On the transport layer TLS [46] is used to secure communication between two nodes. While the technology is well understood and broadly used, the practicability for medical Web services as for Web services in general is low. It requires an additional infrastructure and its use is limited in an environment with established firewalls. The session layer provides a currently often used encryption and authentication solution for Web services based on the HTTP protocol. Nevertheless, its services are bound to network end-points. There is no support for intermediates where traffic passes through and, even more restrictive, there is no application or user level security possible.

Medical Web services require security credentials to be bound to individuals or service instances acting on behalf of an actor of medical workflows (a physician, a patient, etc.). Therefore, WS-

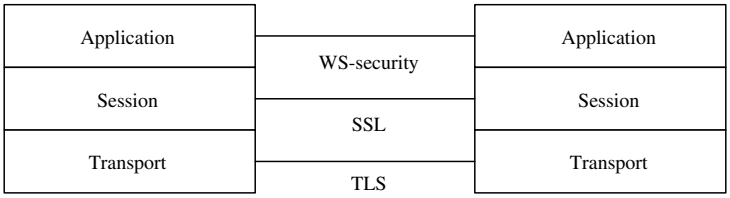

Fig. 4. Security overview.

Security [10] is the security model of our choice. The upcoming standard for securing Web services overcomes all drawbacks earlier mentioned, by

- providing security between individual actors or service instances,
- extending the already existing Web service infrastructure and therefore,
- enabling a common modeling process for security requirements of Web services and,
- reducing implementation efforts.

WS-Security supports username/password security, X.509 certificates [72] and Kerberos [73] authentication or SSL for authentication and encryption purposes. However, it only defines the SOAP encoding of these standards. An established infrastructure for the authentication and encryption process has to be in place. The standard also states which parts of a SOAP message have to be signed and encrypted to avoid message tampering and to ensure the privacy of the communication partners. Furthermore, in WS-SecureConversation [45], Web service providers specify security requirements and requestors provide claims that can be matched prior to security establishment.

On the other hand, by focusing on WS-Security we also recognize hurdles that have to be taken. Medical Web services often transfer large data-sets and a complete encryption of the data requires time. In the case of using HL7 and DICOM as attachments in SOAP messages, WS-Security provides a specification of how this data has to be encrypted additionally. In cases with image data of several 100MB the application-level encryption based on WS-Security is infeasible. Primarily, security should only be applied if definitely necessary. In some cases limited encryption can be applied to the remaining part of the SOAP message.

One additional aspect is models for trust relationships between business partners, where security requirements between trusted partners are more relaxed. WS-Security provides the notion of actors that can be related 1:1 to the concept of IHE actors. For actors with an established trust relationship security tokens, for example, do not have to be evaluated by a certification authority. Furthermore, in Intranet environments encryption and authentication might be omitted completely. If trust relationships as defined in WS-Trust [44] are used, an additional infrastructure for a Security Token Service is required.

For the implementation, the meta-data required to identify communication partners and to select security credentials for a specific workflow instance have to be promoted to be available for the Web service engine to perform security operations. This meta-data resides within the business protocols (HL7, DICOM) and has to be extracted before the data is encoded for transfer.

3.6. Web service registration and binding

The UDDI standard [48] specifies Web services for service registration, subscription, and binding. The binding process can be implemented at design-time or at run-time. For workflows based on the IHE standard run-time binding is required, if a decision for a specific IHE actor differs on a process instance base. This is the case, for example, if a report for an examination is created by a physician based on the patient's diagnosis. The dynamic binding depends on attributes like modality name and requesting physician (DICOM) or referring doctor and assigned patient location (HL7). Attributes like these, which are required for dynamic binding, have to be modeled in BPEL.

For our purpose, the UDDI registry provides yellow pages and green pages services. The former can be used to search for a service that implements specific IHE transactions of an IHE actor. The latter is required to bind to the service at run-time. There is a private and public model to distribute UDDI registries. For our infrastructure, we consider a private model where a registry is maintained by one participant of an IHE transaction.

UDDI stores information about companies, services in general and Web services in particular in an 1:*n* relationship. The IHE framework can be mapped to the registry by creating entries for IHE applications (services) and IHE Web services. Furthermore, a classification scheme is supported and can be used in the IHE context by classifying applications for their support of IHE actor (classes) and IHE Web services for their support of IHE transaction (classes). There is not necessarily a 1:1 relationship between a Web service and an IHE transaction.

UDDI supports a security model for the communication with and the manipulation within the registry. Because the gateway (Fig. 1) already requires a security infrastructure, securing the registration service is reasonable.

3.7. Web service composition

For Web service composition we have to consider the structure and granularity of a Web service to be a suitable part of the executed workflow. Table 1 provides a mapping between IHE concepts and BPEL language constructs that will be discussed further.

An IHE actor is modeled as a BPEL business partner. Applications might perform one or more roles and therefore participate in different BPEL processes. An IHE flow, like the administrative process flow introduced in Section 4, is modeled as a BPEL process. An IHE transaction is mapped on a BPEL service link, where only two business partners are communicating with each other over two BPEL ports. A single HL7 message or DICOM object is embedded in a SOAP message and transferred between the business partners using a BPEL invoke and receive activity. As stated above, BPEL uses WS-Transactions and a compensation mechanism. Compensation activities themselves are implemented as HL7 messages and DICOM objects.

3.7.1. BPEL variables

To specify a BPEL process, it is required to define variables required for the workflow. For medical Web services they consist of the following four categories. First, we require environment attributes for the participating IHE actors and the implemented IHE transactions. This information is stored during composition in the BPEL server itself or for dynamic binding in a UDDI registry. For dynamic binding attributes suggested in Section 4.6 (requesting physician, etc.) have

Table 1
Relationship of IHE concepts and BPEL constructs

IHE concept	BPEL construct
IHE actor	BPEL partner
IHE flow	BPEL process
IHE transaction	BPEL service link, 2 BPEL ports
HL7 message/DICOM service	1 BPEL invoke + receive activity, 1 SOAP message
HL7 message/DICOM service	1 BPEL compensation activity

to be stored additionally. The second category are attributes used to identify the message type (HL7 ADT^A01, etc.) and message content (patient UID, etc.). All message content identifying attributes are used to construct a BPEL correlation set. The third category consists of attributes used in state information and BPEL expressions. For example, the HL7 *PatientClass* is used to control the process flow of the *patient registration* transaction. The last category are the remaining attributes that reside only in the payload and are not part of the BPEL definition.

3.7.2. Basic activities

BPEL uses basic activities to execute the workflow between business partners. In Web services, IHE transactions are executed by performing HL7 and DICOM operations. For each operation between two partners the initiating part executes an *invoke* activity on a defined BPEL port and the receiving partner performs a corresponding *receive* activity on another port. The ports are related in a BPEL service link associating the business partners. The modeling process in Section 4 provides a corresponding example. Details of this relationship for a medical workflow can be found in [28]. Another approach focusing on a supply chain example can be found in [49].

3.7.3. Expressions and structured activities

BPEL uses expressions for conditions and variable assignment using extensions of the XPath [50] standard. Variables of the first three categories can be used in expressions. For example the HL7 *PatientClass* can be used in a boolean expression. BPEL supports, for example, *sequence*, and *while* activities to structure the process. A model of these activities can be partially derived from the sequence diagrams provided in the IHE framework. As shown in the example of Fig. 8 an A01, A04, and A05 message can be sent depending on the HL7 *PatientClass*, therefore a *switch* construct is used within the process. The modeling process in Section 4 provides a corresponding example. For other examples refer to [28,49]. A detailed analysis of BPEL patterns can be found in [51].

3.7.4. Message correlation and correlation sets

The messages sent and received in an IHE transaction have to be correlated by a unique identifier, a BPEL correlation set. This set can be constructed by appending all identifying HL7 and DICOM message attributes and depends on the structure of the underlying messages exchanged. An example for the patient registration has been introduced in Section 2.4. In general, the attributes have to be derived from the standardization documents for each transaction.

3.7.5. Scopes and compensation activities

A scope is a BPEL construct used for error or compensation handling. Compensation handlers can be defined on a scope level to perform compensation activities in case of application level errors. Compensation activities can also be used in error handlers for system level errors. As mentioned earlier, some of the IHE transactions activities require compensation and some do not. This information has to be derived from the respective standard documents. For the patient registration example the HL7 A01 message has to be compensated by an A11 message. The granularity of an IHE transaction is a candidate for defining scopes as its outcomes are defined clearly within the IHE framework. Further modeling examples should prove this assumption. Currently, there is no evidence for the use of nested transaction.

3.8. Infrastructure requirements

Combining the requirements outlined above, we are able to define infrastructure requirements for implementation of medical Web services. This infrastructure is a detailed view of the gateway node (Fig. 1). The node acts as the translator between the business protocols (DICOM, HL7) and the Web service infrastructure. In general, this results in a component model similar to the one shown in Fig. 5.

The infrastructure consists of a workflow core in the middle that uses a database, a workflow engine and a BPEL engine. We do not assume that the overall workflow is executed within the BPEL engine.

For medical Web services we require data adapters for DICOM and HL7 to incorporate business data into the workflow. Additionally, the DICOM data has to be compressed as outlined above. On the back-end (the Intranet side) the protocols have to be provided through the native DICOM and HL7 services. Furthermore, a proprietary adapter might support locally used protocols.

On the Web service part of the infrastructure security, transaction, and attachment operations (WS-Security, WS-Transactions, DIME, etc.) are performed. Those operations require a policy that is established from IHE requirements implemented in the BPEL engine and enforced on workflow activities. The policy component has to intercept the SOAP engine, evaluate policies, attach security headers to messages and generate attachments. Depending on the target platform, Web service standards are provided within or outside the BPEL or workflow engine.

For BPEL workflow execution of medical Web services all operations are meant to pass back and forth between the native services and the Web service. Additionally, the native services might provide functionality independently. Nevertheless, with workflow instances executed in the workflow core, the IHE conformity can be guaranteed.

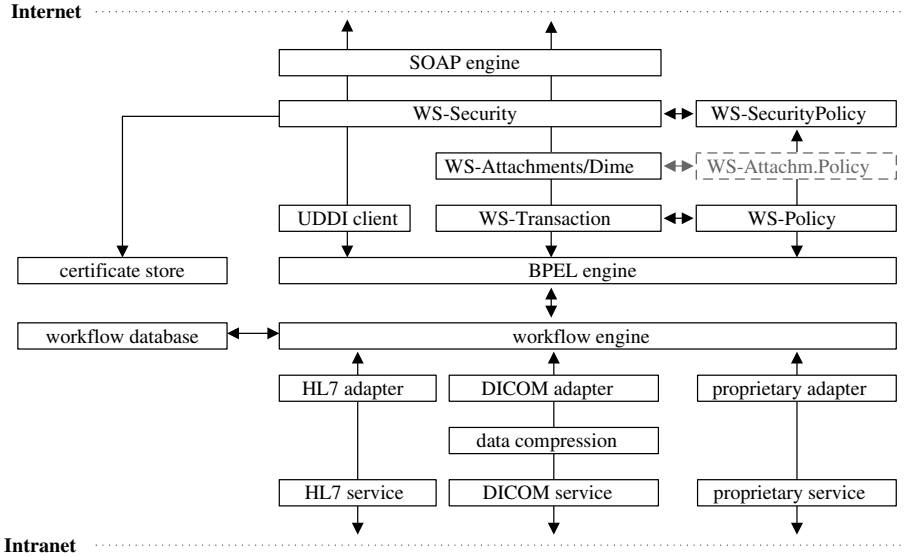

Fig. 5. Medical Web service gateway infrastructure.

4. Modeling medical Web services

The modeling process is separated into four steps. First, we provide the four layer model to structure the content of the IHE framework. In the second step the process flow is defined and normalized. In the third step a similar approach is performed for the IHE transactions. Finally, based on the normalized descriptions, BPEL and WSDL definitions are derived.

4.1. Definition of a four-level Use-Case model

The first step for modeling medical Web services is the definition of 4-level UML [52,53] Use-Case model. This model has been introduced in [20] and is shown in Fig. 6.

The layers used correspond to the definitions for profiles, flows, transactions, and messages used in the IHE framework. On the top layer the IHE integration profiles are shown, a coarse grained overview of what an application performs. The *IHE Scheduled Workflow profile* we focus on is shown in the gray shaded area. These profiles are split into several flows. Each flow must be supported by an application that implements the profile (in our example the *administrative process*

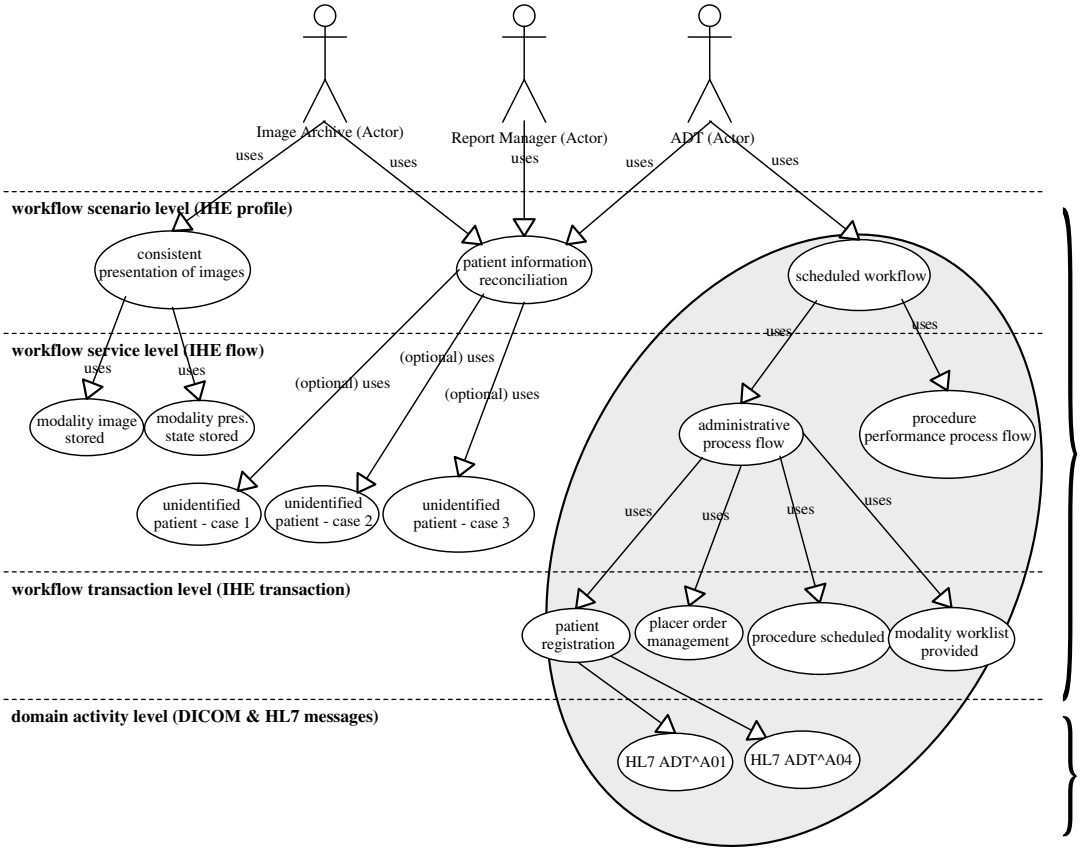

Fig. 6. Modeling process—four-level Use-Case model.

flow). IHE flows are defined as sequence diagrams in the IHE framework. Each IHE flow is further defined using several IHE transactions. These transactions are sequentially ordered and not all transactions of a flow have to be implemented by every participating actor. Finally, a transaction consists of one or several HL7 and DICOM messages that have to be sent or received. The upper three levels correspond to the workflow layer of the middleware, while the fourth resides in the domain layer. While conventional workflow systems focus on the third and forth layer our approach takes the structure of the whole IHE specification into account. For readability different Use-Case models should be created to focus on the implemented IHE actors of a specific application. The IHE transactions that have to be modeled in the next step can be taken from layer 3. For designing medical Web services, we further focus on the *administrative process flow*. The Use-Case model has been introduced for medical workflows in [20].

4.2. Selection, definition, and normalization of process flow

In a second step, we proceed to focus on the *administrative process flow* and provide an activity diagram (Fig. 7) that corresponds to the public workflow for the *Department System Scheduler* IHE actor and is the derived from the corresponding sequence diagram defined in the IHE framework [4].

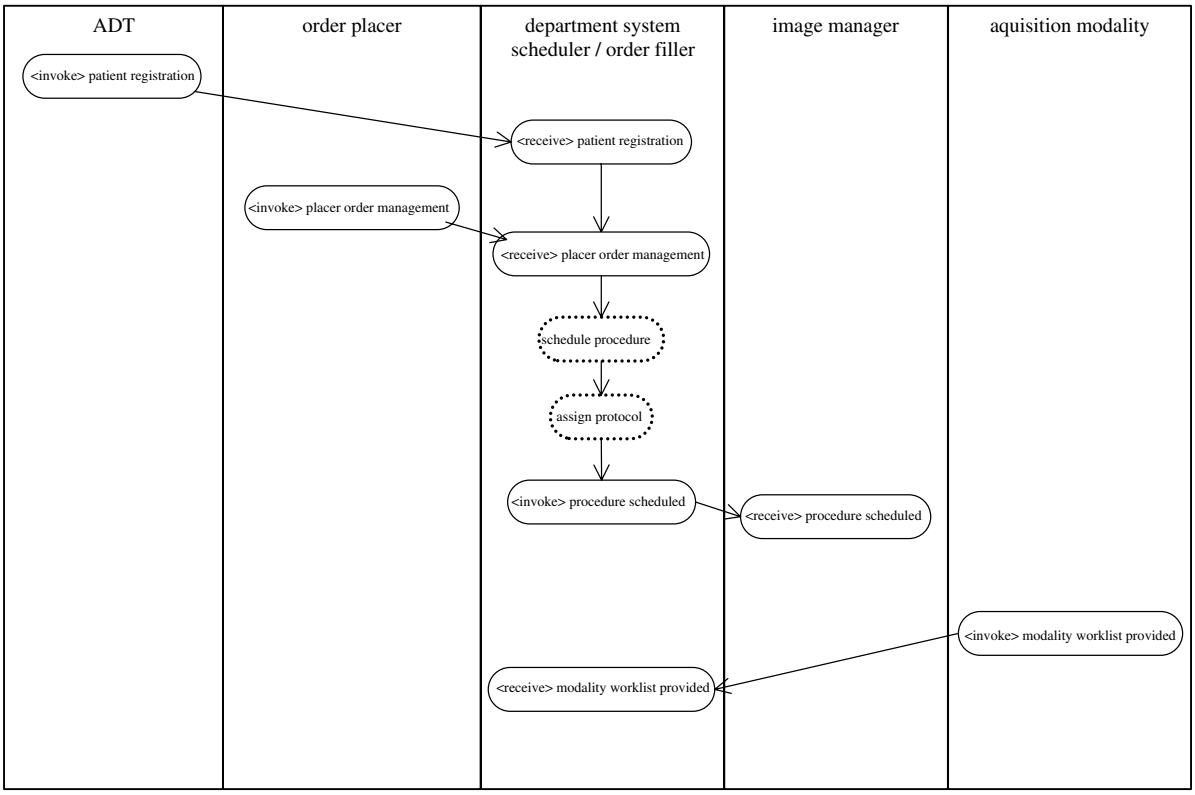

Fig. 7. *Administrative process flow*—public process of *Department System Scheduler*.

UML activity diagrams are widely used as a representation language for workflows as discussed in [54]. The public process contains all activities (IHE transactions) performed by the IHE actor, internal operations are shown for readability. The diagram can be derived from the sequence diagram by performing several normalization operations.

First, an IHE actor to define the public process for is selected and actor independent and internal operations are deleted. Next, IHE transactions are translated into BPEL *invoke* and *receive* activities. Caution has to be taken, because IHE defines some of the transactions in the wrong direction. For example, the DICOM service in the IHE *modality worklist provided* transaction is shown as being executed by the *Department System Scheduler* on the *acquisition modality*. However, it is the client (*acquisition modality*) that queries the server during this operation, therefore the *invoke* activity is performed by the *acquisition modality*.

Furthermore, the conversion results in two independent processes, therefore an IHE flow not necessarily corresponds 1:1 to a BPEL process. As another fact, an application might implement several roles in the IHE flow, therefore converting external transactions to internal which are not modeled in a BPEL process. To join two actors *invoke* and *receive* activities between them are converted to internal operations and omitted. The two sets of other activities are joined.

The diagram outlines requirements of the process to implement. However, a BPEL process cannot be directly derived because details of the underlying domain layer are omitted. These details are provided in the next step.

4.3. Selection, definition and normalization of transactions

In a third step we focus on the activities performed in an IHE transaction. The HL7 and DICOM messages exchanged between two systems in a *patient registration* transaction are outlined in Fig. 8.

The activity diagram corresponds to the sequence diagram of the *patient registration* transaction defined in [4]. The diagram is a more detailed view of the IHE flow above. The simplified *invoke* and *receive* activities of Fig. 7 might now be split into one or more BPEL activities. The *invoke* operation is annotated in the flow at the initial sender of the transaction (the *ADT* actor in our example).

Several implications on an implementation have to be considered from the standard documents to normalize the activity diagram. For example, the patient registration distinguishes in-patient, out-patient, and pre-registration. These cases depend on the *PatientClass* attribute of the PV1 segment of HL7 ADT messages. In the BPEL process this results in a *switch* structured activity and in several initiating *receive* activities for the process of the *Department System Scheduler*. BPEL supports multiple start activities by setting the *createInstance* attribute of these activities to "yes". Furthermore, HL7 requires acknowledge messages to be sent back to the initiator. These are modeled using an additional pair of *invoke* and *receive* activities.

As another example, Fig. 9 shows the activities performed in a *retrieve images* transaction. The transaction is split in two pairs of *invoke* and *receive* activities. No additional steps are necessary. The requirement for attachments, because of the large amount of data being exchanged, is simply annotated.

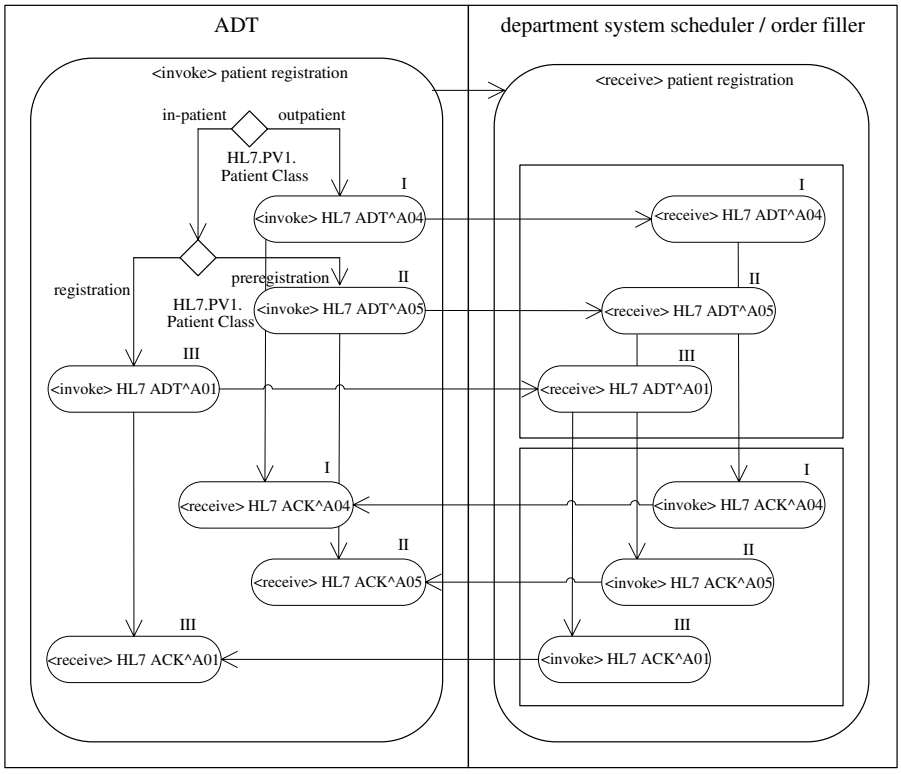

Fig. 8. *Patient registration* transaction—public process of *Department System Scheduler*.

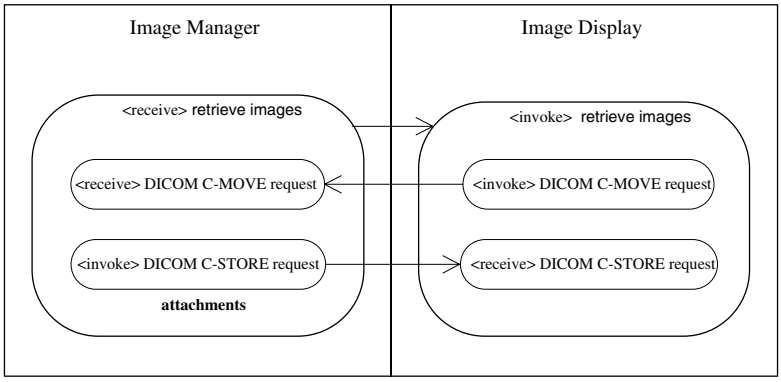

Fig. 9. *Retrieve images* transaction—public process of Image Display.

4.4. Integrating security requirements

Our modeling process has to be enriched with security attributes on different levels. On the level of an IHE flow (BPEL process) we introduce the notion of security domains and their security boundaries. A security domain contains one or more IHE actors (BPEL partner) and can be either

authentication-free, if the actors trust each other, or encryption-free if the actors reside in the same safe physical location. Security boundaries on the other hand distinguish between actors inside and outside such a domain and therefore require encryption and/or authentication. To illustrate this concept, Fig. 10 shows the public process of *Department System Scheduler* where the patient administrative data is exchanged between the central *ADT* actor and a branch location where the *Department System Scheduler* and the other actors reside. We use annotations within the activity diagrams to show whether encryption, authentication or both is required.

Domain A covers the ADT, while Domain B covers all other actors. Both sites trust each other. Therefore, no authentication between the actors is required.

On the next level of detail in the modeling process, we have to provide information about the security attributes of the performed IHE transactions. The medical domain requires a minimum of 128bit key-length and asymmetric encryption. However, the exact algorithms that are used within the implementation are not part of the modeling process. Fig. 11 shows the security attributes for the IHE *patient administration* transaction.

This model differs from the definition in IHE. It enforces security requirements only if necessary with respect to performance and simplicity.

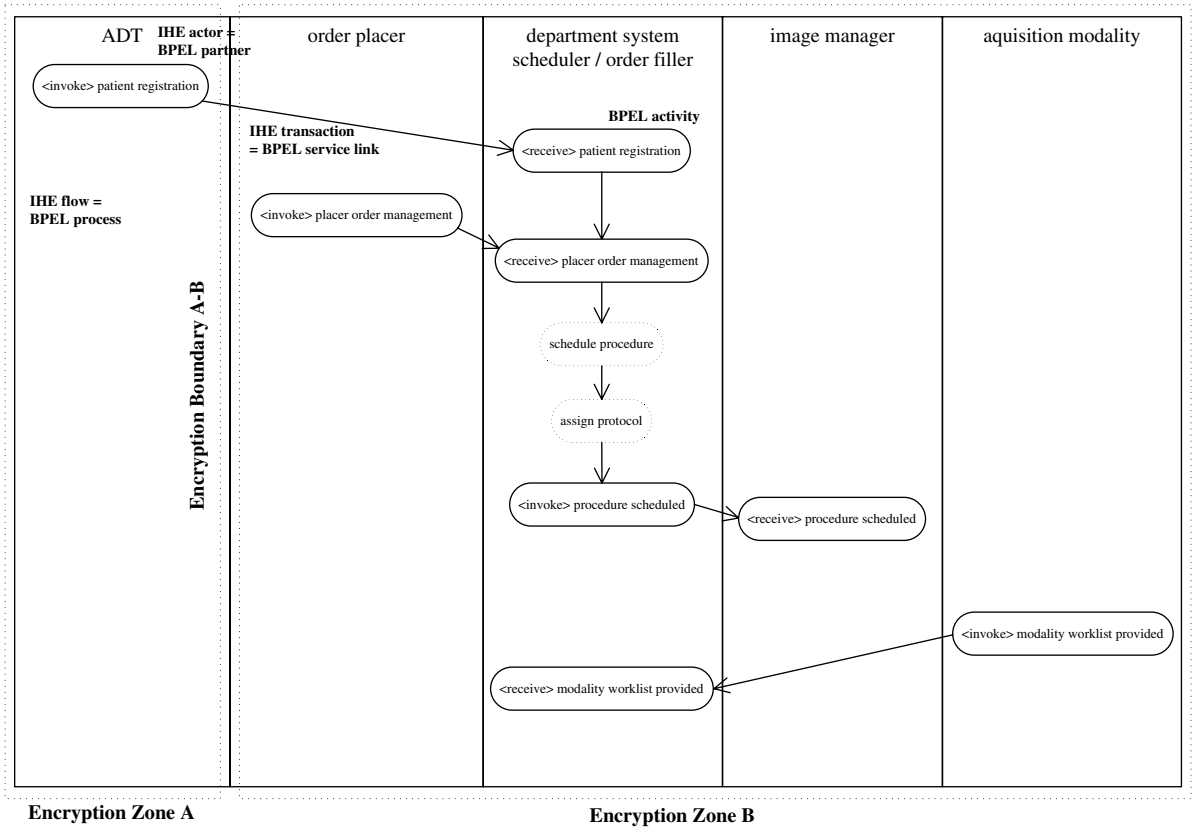

Fig. 10. Security domains in the public process of *Department System Scheduler*.

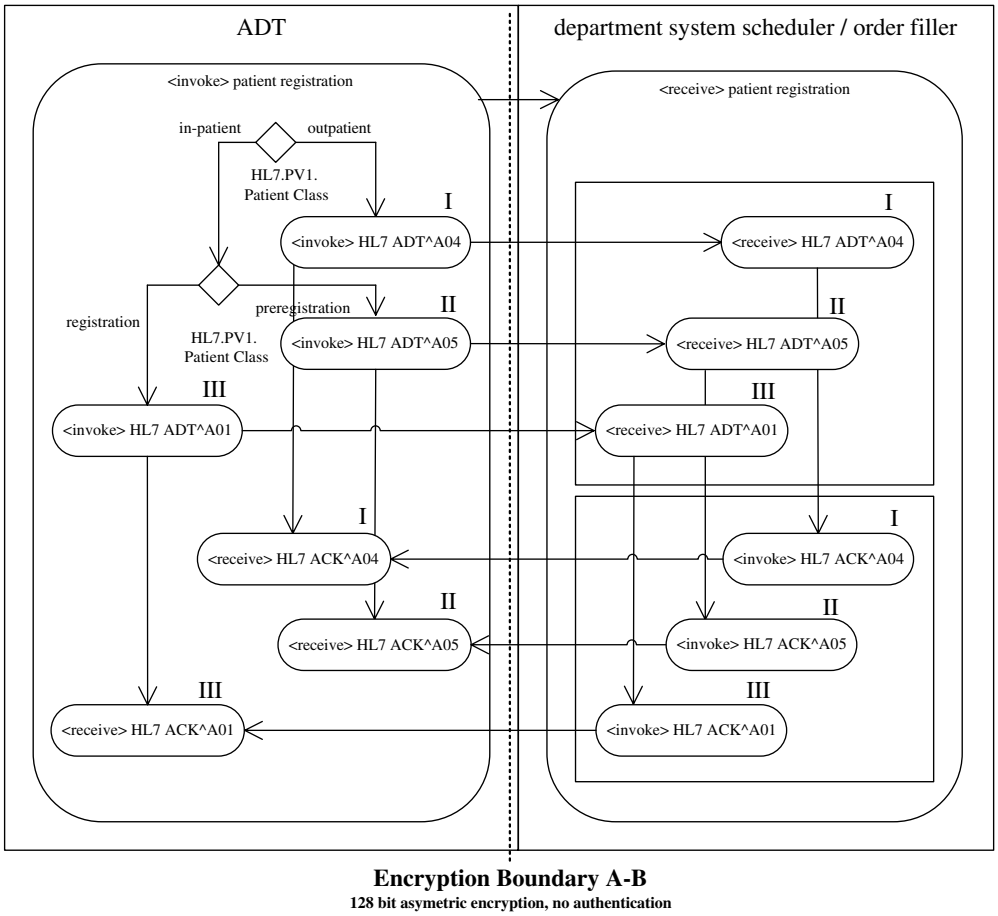

Fig. 11. Security attributes for IHE *patient administration* transaction.

As we will see in the next section, the transactional semantics for workflow languages like BPEL are further developed than security-related ones. That might be a consequence of security operations being more orthogonal to the workflow activities themselves, while transactions are more a part of modeling workflow activities. These differences become visible when defining infrastructure requirements and finally when applying our examples to the implementation environment.

4.5. Integrating transaction requirements

For the modeling process we have to define scopes within the activities and provide corresponding compensation activities as specified in WS-Transaction. Again, we use the sample of the *patient administration* transaction as shown in Fig. 12.

The transactions are shown in rectangles and scopes are shown with curly braces. The compensation activities are drawn below the scope within the transaction rectangle. In our case, the ADT^A04 message is compensated in case of system or application level errors using an

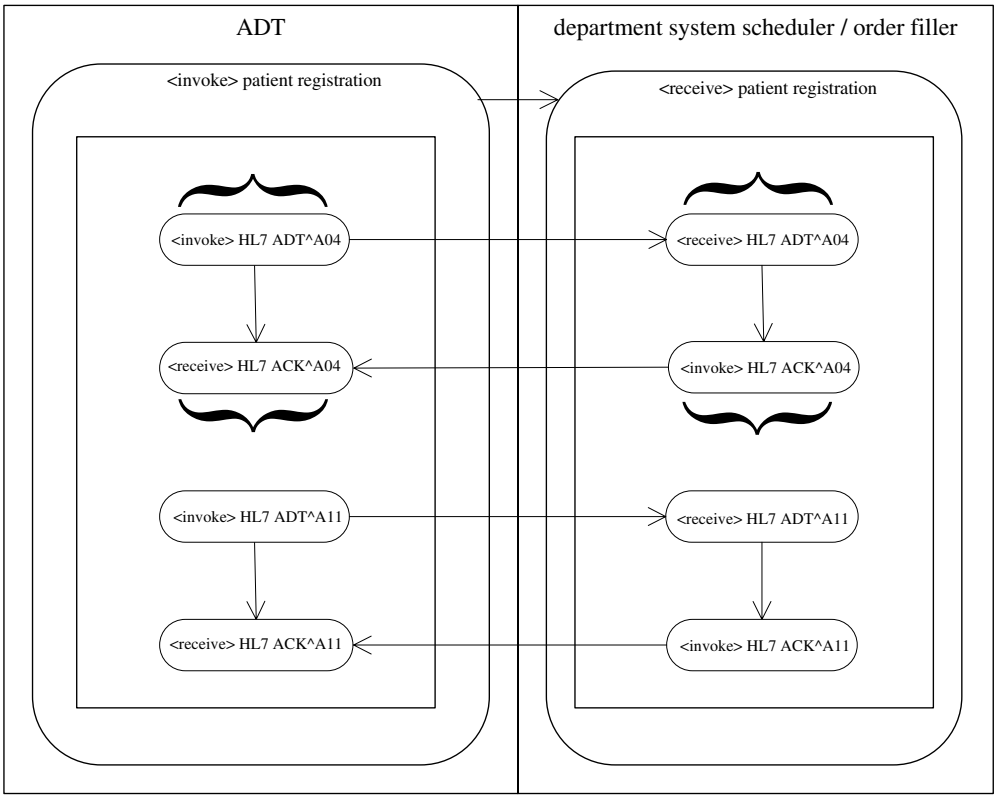

Fig. 12. Compensation-based transaction for the IHE *patient registration* transaction.

ADT^A11 message that undoes the effects of a patient registration (e.g., by canceling the admission or deleting records).

In general, an IHE transaction corresponds to a WS-Transaction in granularity. Which compensation activities have to be performed in a certain situation has to be taken from the standard documents. Of course, several transactions within IHE lack an explicit compensation activity. For example, the *retrieve image* transaction can only be compensated by deleting the images, database entries, and other application specific operations. This behavior is out of the scope of the modeling process.

4.6. Definition of a BPEL process

In the next step, we are able to derive a BPEL process specification from the provided activity diagrams for the *patient registration*. In short, the following tasks are necessary. The BPEL specification contains definitions of *types*, *variables*, *messages*, and *correlationSets* that can be derived from DICOM and HL7. Furthermore, business *partners* and a *process* using basic and structured activities are defined. The WSDL file contains a *portType* and a *serviceLinkType* Section to define the Web services. Finally, compensation activities are provided using scopes and security issues are outlined.

4.6.1. Definition of a BPEL specification

The workflow process is specified as a BPEL file with a predefined structure. We provide the main sections of the specification. First, the *partners* section covers the implementing application, the supported *serviceLinkType* and the role that is performed in the process.

```
<partners>
  <partner name="app X"
    serviceLinkType="IHETransPatientRegistration"
  myRole="IHEActorADT"/>
</partners>
```

Next, two sections contain message property type definitions used throughout the communication. For example, the A01 message is outlined.

```
<definitions name="properties"
  <bpws:property name="patientId" type="xsd:string"/>
  <bpws:property name="registrationProcessId" type="xsd:string"/>
</definitions>
<types>
  <xsd:schema>
    <xsd:complexType name="HL7_A01_TYPE">
      <xsd:element name="patientId" type="xsd:string"/>
      <xsd:element name="registrationProcessId" type="xsd:string"/>
      <xsd:element name="payload" type="xsd:string"/>
...
```

Furthermore, two sections contain the variable and message definitions used throughout the process. For example, the A01 messages and acknowledgement are outlined below.

```
<variables>
  <variable name="HL7_A01_VAR" messageType="HL7_A01_MSG">
  <variable name="HL7_A01_ACK_VAR" messageType="HL7_A01_ACK_MSG">
...
  <message name="HL7_A01_MSG">
    <part name="HL7_A01_PART" type="HL7_A01_TYPE"/>
...
```

An additional definition for the correlation sets is required. The correlation is defined by the unique patient identifier and a registration process id.

```
<correlationSets>
  <correlationSet name="HL7_A01_CS" properties="patientId registra-
   tionProcessId"/>
...
```

Finally, the flow section contains the process definition itself. The example shows the *switch* statement corresponding to the sequence diagram, the *invoke* and *receive* activities for the A01 messages and the used correlation set.

```
<flow>                                                    <!– patient registration flow –>
  <switch>                                               <!– switching PatientClass –>
  <case HL7_A01.PV1.PatientClass=in-patient>             <!– in-patient –>
    <switch>                                              <!– switching PatientClass –>
    <case HL7_A01.PV1.PatientClass=registration>         <!– registration –>
      <invoke partner="app Y" portType="IHETransPatientRegistrationCallbackPort"
      operation="HL7_A01" inputvariable="HL7_A01_MSG">
        <correlations>
            <correlation set="HL7_A01_CS">
        </correlations>
...
```

4.6.2. Definition of a WSDL specification

Besides the process definition, it is also necessary to define the Web service communication ports using a WSDL description. The main parts, the *portType* and the *serviceLinkType* sections are outlined here. A patient registration port and callback port are specified together with the required messages exchanged.

```
<portType name="IHETransPatientRegistrationPort">
  <operation name="HL7_A01">
    <input message="HL7_A01_MSG">
  </operation>
<serviceLinkType name="IHETransPatientRegistration">
  <role name="IHEActorADT">
      <portType name="IHETransPatientRegistrationPort">
  </role>
  <role name="IHEActorDepartmentSystemScheduler">
      <portType name="IHETransPatientRegistrationCallbackPort">
  </role>
</serviceLinkType>
```

4.6.3. Applying transactions

To complete the modeling process, transactions, security, and attachments have to be applied. As stated in Section 4, a transaction and security context has to be generated for each IHE transaction. Compensation-based transactions are supported in BPEL using the *scope* section of the process. In our example, an A01 message is compensated by an A11 message as outlined below.

```
<scope>
  <compensationHandlers>
    <invoke partner="app Y" portType="IHETransPatientRegistrationCallback-
     Port"
     operation="HL7_A11" inputvariable="HL7_A11_MSG">
```

```
      <correlations>
        <correlation set="HL7_All_CS">
      </correlations>
    </invoke>
    <receive partner="app Y" portType="IHETransPatientRegistration-
     Port"
     operation="HL7_All_ACK" inputvariable="HL7_All_ACK">
     <correlations>
       <correlation set="HL7_All_CS">
     </correlations>
    </receive>
  </compensationHandlers>

  <invoke partner="app Y" portType="IHETransPatientRegistrationCallback-
   Port"
   operation="HL7 AO1 admit visit notification" inputvariable="HL7_
   AO1">
   <correlations>
     <correlation set="HL7_AO1_CS">
   </correlations>
  </invoke>
  <receive partner="app Y" portType="IHETransPatientRegistration-
   Port"
   operation=" HL7 AO1 acknoledge" inputvariable="HL7_AO1_ACK">
     <correlations>
       <correlation set="HL7_AO1_CS">
     </correlations>
    </receive>
  </scope>
```

4.6.4. Applying security and attachment requirements

Security and attachment semantics are not part of the BPEL specification. They have to be implemented in a separate component using WS-Security, WS-Policy, and WS-Security Policy (see Fig. 5). Security requirements, appropriate for our sample *patient registration* transaction, can be modeled as outlined below. Of course, security headers consist of much more than what is presented here. We focus on topics for medical Web services.

```
<wsp:Policy xmlns:wsse="..." xmlns:wsp="...">
  <wsp:ExactlyOne>
    <wsse:SecurityToken wsp:Usage="wsp:Required" wsp:Preference="100">
      <wsse:TokenType>wsse:Kerberosv5TGT</wsse:TokenType>
    </wsse:SecurityToken>
```

```
    <wsse:SecurityToken wsp:Usage="wsp:Required" wsp:Preference="1">
      <wsse:TokenType>wsse:X509v3</wsse:TokenType>
    </wsse:SecurityToken>
  </wsp:ExactlyOne>
</wsp:Policy>
```

The policy used in our example requires the use of a Kerberos TGT (ticket granting ticket) or an X509 certificate (not both). Furthermore, Kerberos is the preferred method, which might differ between implementations. If a security infrastructure is already in place, the security tokens do not have to be attached to the messages; otherwise they are and can be encrypted optionally.

The signing and encryption properties are implemented with the *Signature* element (see XML-Signature [68]) and the *EncryptedData* element (see XML-Encryption [67]). 128bit key-length is required when specifying the algorithms. Of special interest is the encryption of attachments. They can simply be put into the *ReferenceList* of encrypted data items. Additionally, it is necessary to change the MIME type from *application/Dicom* to *Application/octet-stream*.

```
<xenc:EncryptedData MimeType="Application/Dicom">
  <xenc:EncryptionMethod Algorithm="http://www.w3.org/2000/09/xmld-
   sig#rsa-sha1"/>
  <ds:KeyInfo>
    <ds:KeyName>CN=ADT_TEST, C=AT</ds:KeyName>
  </ds:KeyInfo>
  <xenc:CipherData>
    <xenc:CipherReference URI="dicom:image"/>
  </xenc:CipherData>
</xenc:EncryptedData>
```

SOAP headers for signing and encrypting messages and attachments are inserted depending on the security requirements of the corresponding IHE transaction. Attachment requirements are not specified in WS-Policy, a fictitious notation might look like outlined below, where *Wsa* is a namespace for Web service attachments and the *AttachmentMethod* and *Protocol* elements specify the method of implementing attachments.

```
<wsp:Policy>
  <wsa:AttachmentMethod wsp:Usage="wsp:Required">
    <wsa:Protocol Type="wsa:DIME">
  </wsa:AttachmentMethod>
</wsp:Policy>
```

The sample states that the DIME protocol is required for attaching data to SOAP messages. The called Web service might refuse service if it does not support DIME.

5. Implementing medical Web services

In this section we provide an implementation of the IHE *patient registration* sample transaction using Biztalk Server 2004. Where necessary, we added functionality for additional Web service features.

5.1. Introducing Biztalk

Microsoft Biztalk Server was released before the emergence of Web services. It provides functionality for EAI (Enterprise Application Integration) and B2B (Business to Business) scenarios and can be compared to IBM WebSphere [7], BEA Weblogic [69] or Oracle (Collaxa) BPEL Server [37]. Medical Web services infrastructure (see Fig. 1) consists of EAI tasks performed on the middleware layer and B2B tasks additionally performed on the Web service layer.

With the recent specification of BPEL, it also implements Web service orchestration and, therefore, served as a candidate for a medical Web service implementation evaluation. The product contains several tools for the. NET development environment, like an orchestration designer, an XML mapper, and a server administration console.

5.1.1. Security features

Biztalk uses Internet Information Server (IIS) for Web service enactment. IIS supports different application pools to run simultaneously. Web services that are published from Biztalk server can be applied to a specific application pool and therefore run under a different user account, which enhances the security of the gateway node itself. Additionally, Biztalk executes workflows on Biztalk server hosts. One host may contain items that receive, send, and process messages. Security boundaries can be considered by creating different hosts. For example, it is recommended that different hosts for internal processing and external *invoke receive* activities are used. Also trusted and non-trusted items should be separated. Furthermore, Biztalk ports can be assigned access rights (public, restricted, etc.). This allows a role based access control and reduces the proneness to denial of service attacks.

5.2. Design of the implementation

The first task was to check the target platform for the necessary product support. Related to Fig. 5, we are able to outline which parts of the solution can be satisfied within Biztalk Server and others which have to be provided additionally (see Fig. 13).

The grey shaded areas are implemented within Biztalk Server 2004. Primarily, the workflow core is implemented, which can be expected of any implementation environment. HL7 functionality and its TCP based lower layer protocol MLLP are available as an add-on adapter. However, to provide a service implementation, business logic has to be implemented within the workflow engine. Biztalk additionally implements WS-Transaction and provides a transaction compensation mechanism. All the other functionality has to be implemented on top of Biztalk. A DICOM adapter has to be implemented or can be simulated by using the file-drop mechanism of Biztalk. DIME is not part of Biztalk, therefore, images cannot be transferred using an attachment mechanism. Furthermore, WS-Security and WS-Policy standards are not available in Biztalk. Biztalk

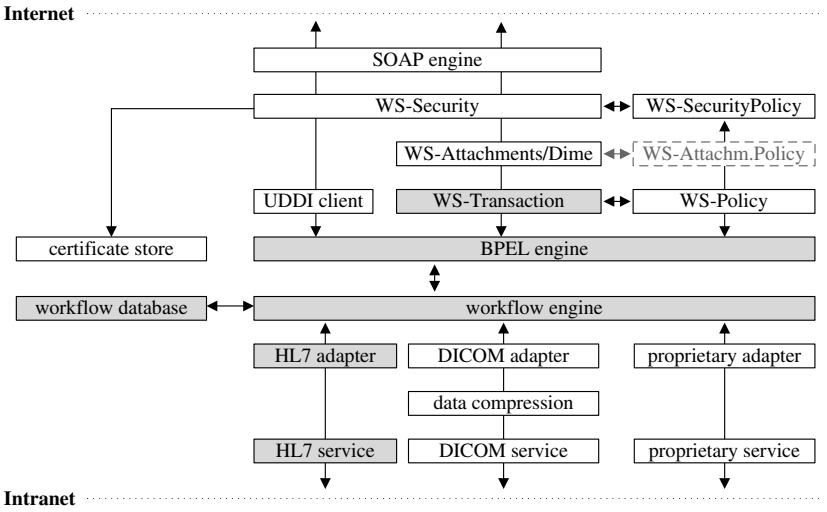

Fig. 13. Medical Web service infrastructure using Biztalk 2004.

supports authentication and encryption functionality for S-MIME with its MIME adapter. Finally, UDDI functionality is not built into Biztalk and has to be provided additionally.

5.2.1. Options for missing functionality

Because Biztalk is only available for the Microsoft Windows operating system this restriction also applies to all additional functionality that has to be provided. The Microsoft Web service Enhancements 2.0 for. NET [38] extends the Web service stack with WS-Security and DIME support. While this is useful for test purposes, the functionality cannot be easily integrated with Biztalk. However, the Windows operating system contains a certificate store that can be leveraged. The UDDI client can be implemented using the UDDI. NET SDK [39].

5.3. Biztalk sample functionality

The IHE *patient registration* sample introduced in Section 2 can be implemented as a BPEL workflow within Biztalk server. Furthermore, the corresponding HL7 messages XSD (XML schema definition) can be imported with the HL7 adapter. Fig. 14 shows the *ADT* and *Department System Scheduler* (*DSS*) roles' part of the workflow that corresponds to Fig. 8 using the Biztalk orchestration designer.

The left part corresponds to the *ADT* role and shows a transaction scope for the sending of an ADT^A01 message and receiving of an acknowledge message. In case of a system or application error the compensation message A11 is sent to undo the effect of the patient registration. Again, an acknowledge message is received. On the right-hand side of Fig. 14, the DSS role receives an A01 message and instantiates a workflow. Again, a scope is used and a negative acknowledge is returned to the sender in case of an error. Fig. 14 does not include the ports that are configured for communications. The sending ports of the *ADT* and the receiving ports of the *DSS* are Web service ports, while all other ports are HL7 native ports that close the gap to the local infrastructure.

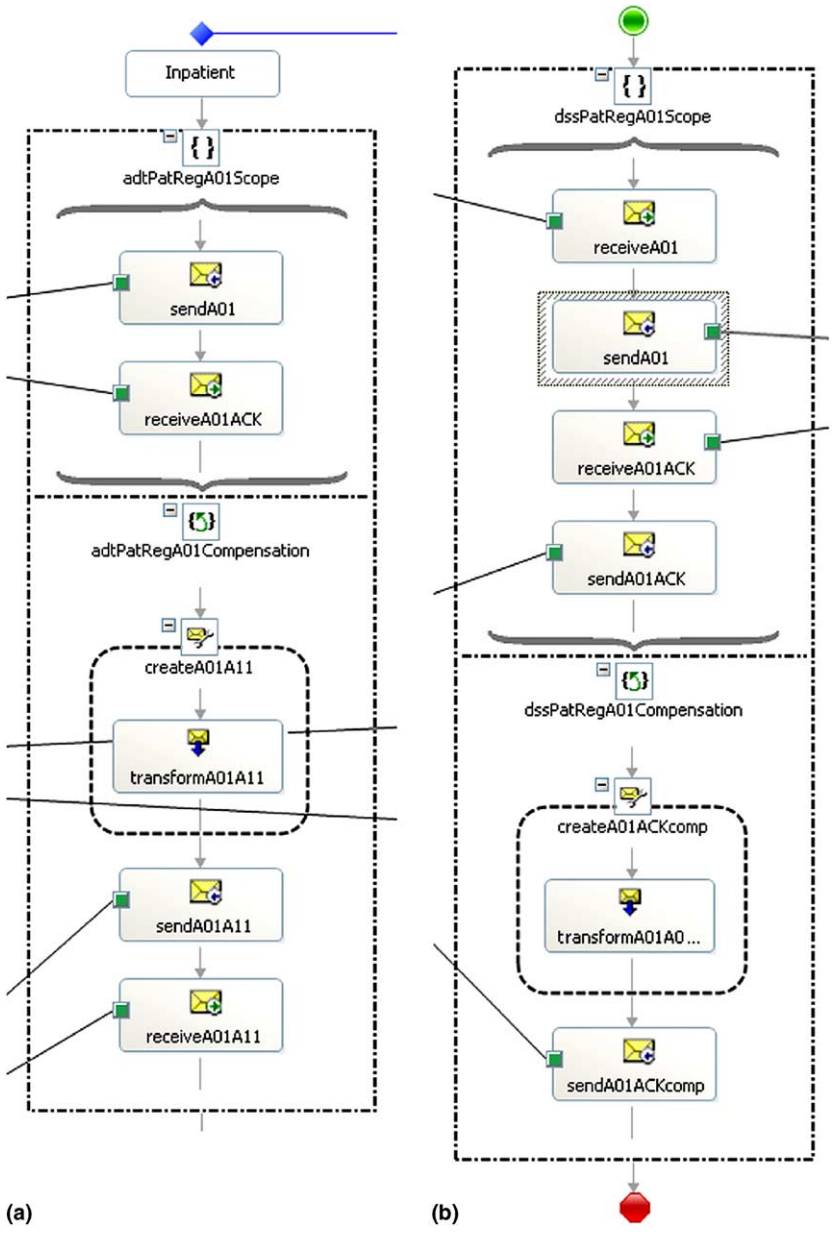

Fig. 14. *Patient registration* transaction in Biztalk 2004 orchestration designer.

Of additional interest is the transformation step that is used when creating A11 or acknowledge messages. Fig. 15 shows the mapping editor that ships with Biztalk.

Fig. 15 shows the A01 message on the left and the A11 message on the right side. Between parts of the messages string, numeric and other operators can be inserted to manipulate the data transition between messages (in our case a bulk-copy of message elements). This mapping feature allows for the creation of HL7 messages on a process instance base automatically. Unfortunately,

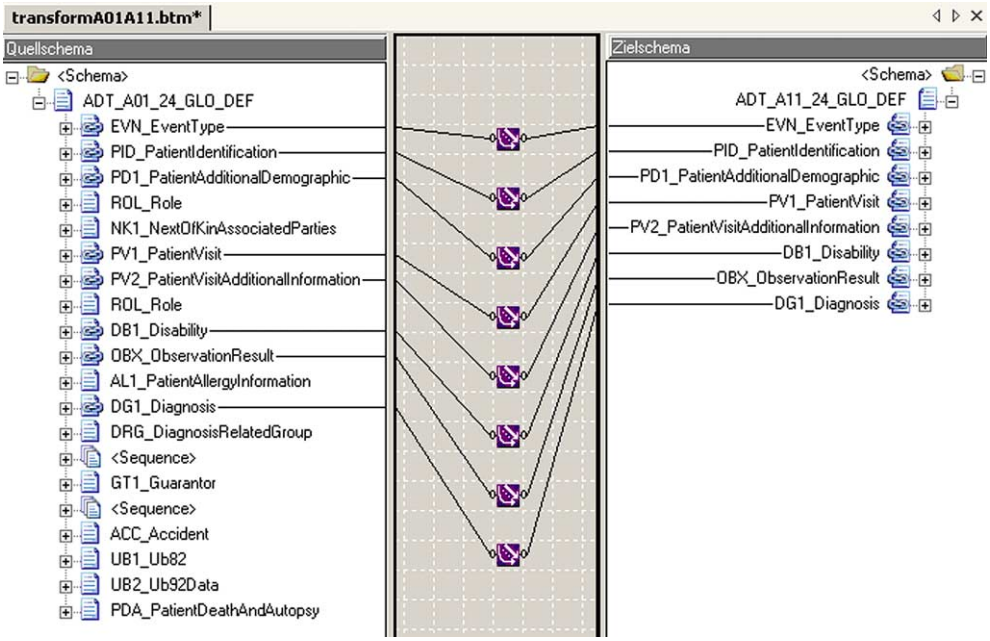

Fig. 15. A01–A11 HL7 message mapping with Biztalk mapping editor.

this functionality is not part of the BPEL specification and all workflow definitions containing such constructs are not allowed to be exported.

5.4. Implementation of WS-Security

The WSE support for WS-Security allows a sample implementation for the encryption and authentication in a simple Web service client/server scenario. Fig. 16 shows an excerpt of the server, Fig. 17 of the client source code using VB.NET.

The WSE easily supports the generation, use, and validation of certificates for encryption and authentication. A correctly set up certificate store is required to perform secure Web service operations. Again, this functionality is not yet part of Biztalk server.

5.5. Implementation of attachments using DIME

The WSE supports an implementation of DIME [41], a standard for attaching custom content to an XML message using base-64 encoding. As stated above the compression and transfer of images cause a small delay in the communication. Therefore, of special interest are additional latencies caused by the base-64 encoding. Measures performed on the WSE show that the encoding of images also lasts less than 1 s.

Fig. 18 shows a VB.NET sample code using WSE and DIME. However, the functionality is not yet integrated into Biztalk.

```
Dim x509Token As X509SecurityToken
Dim serverCert As X509SecurityToken

' check whether encryption was used
x509Token = AppBase.GetEncryptingToken(RequestSoapContext.Current)

If x509Token Is Nothing Then
  Throw New SecurityFault(...
End If

serverCert = AppBase.GetServerToken(False, False)

' check authenticity of client
If Not serverCert.Equals(x509Token) Then
  Throw New SecurityFault(...
End If

' message content goes here
```

Fig. 16. Sample of server code using WS-Security for encryption and authentication.

```
' Create an instance of the Web service proxy and configure the proxy
Dim serviceProxy As New StockServiceWse()

ConfigureProxy(serviceProxy)

' Generate asymmetric key
Dim token As X509SecurityToken = AppBase.GetServerToken(False, True)

If token Is Nothing Then
  Throw New ApplicationException(...
End If

' Add an EncryptedData element to the security collection to encrypt the request.
serviceProxy.RequestSoapContext.Security.Elements.Add(New EncryptedData(token))

' Web service call goes here
```

Fig. 17. Sample of client code using WS-Security for encryption and authentication.

The Web service SOAP context is extended with an attachment collection where files can be inserted. The encoding and transfer of data is performed automatically. DICOM supports the Mime Type ''Application/Dicom'' as specified in [40]. The WSE also directly performs base-64 encoding of XML parameters. Therefore, it is also possible to implement a DICOM data transfer without the DIME protocol.

5.6. Summary

The sample implementation of an IHE transaction with Biztalk 2004 shows the possibilities and limitations of the platform. Transactions based on HL7 messages can be implemented without major difficulties. WS-Transaction on compensation-based transactions can be implemented seamlessly.

```
' The DICOMCStoreWithDime Web service returns a DICOM image in a DIME attachment._
Public Sub DICOMCStoreWithDime(strFileName as string)
  Dim att(1) As DimeAttachment
  Dim ctx As SoapContext = ResponseSoapContext.Current

  att(0) = New DimeAttachment("Application/dicom", TypeFormat.MediaType, strFileName)

  ctx.Attachments.Add(att(0))

  ' Web service call goes here
```

Fig. 18. Sample of client code using DIME to attach DICOM image data.

However, important security features like WS-Security support are still missing. For DICOM additional problems for the handling of binary data arise. Without DIME support it is necessary to use base64 encoded parameters, with negative effects on the performance of an implementation. Additionally, we have to deal with limited platform support.

Beside these drawbacks, Biztalk 2004 contributes to the establishment of a medical Web services infrastructure. A comparison with other products should lead to results about the quality of the solution based on this product.

6. Conclusions and future work

In this paper we have introduced the medical services domain, defined requirements for all aspects of designing medical Web services, outlined a Web service modeling process for IHE framework transactions, and performed an evaluation of an IHE transaction using Biztalk 2004.

However, several points remain unsolved in this context. First, there are problems normalizing activity diagrams caused by ambiguities in the medical industry standards. Furthermore, some of the standard specifications of the Web service stack are not yet widely implemented or, especially for coordinating services, competing standards exist. This area is currently subject to change and the implications on an infrastructure for Web services have to be revised subsequently.

Furthermore, there are several directions to proceed in future work. First, the modeling process, especially the mapping between IHE, HL7, and DICOM standard definitions on one hand and a formal definition of UML diagrams and BPEL constructs on the other hand. Next, more transactions should be modeled to adapt and extend the modeling process. The result should be a semi-automatic process for defining medical Web services' workflows using Web service based composition.

An additional Web services infrastructure should be used to evaluate implementation specific issues and to compare the results with the experiences from Biztalk 2004. Finally, the requirements on the middleware pointed out should result in an architecture for the execution of medical Web services.

Acknowledgements

We thank the anonymous reviewers for their valuable comments which helped to improve this paper.

References

[1] Workflow Management Coalition: WMFC Reference Model, http://www.wfmc.org, 1995.
[2] HL7 Organization: Health Level 7, http://www.hl7.org, 2000.
[3] NEMA and Global Engineering Group: DICOM 3 Standard, http://www.nema.org, 1998.
[4] Radiological Society of North America: IHE Technical Framework 1.1, http://www.rsna.org/IHE/index.shtml, 2003.
[5] CEN/TC251 Health informatics—Medical data interchange: HIS/RIS-PACS and HIS/RIS Modality Interface—ENV 13939, http://www.centc251.org/, 2001.
[6] W3C: SOAP Version 1.2, http://www.w3.org/TR/soap12-part1/, 2003.
[7] W3C: Web services Description Language (WSDL) 1.1, http://www.w3.org/TR/wsdl.html, 2001.
[8] BEA, IBM, Microsoft: Web services Coordination (WS-Coordination), http://www.ibm.com/developerworks/library/ws-coor/, 2002.
[9] BEA, IBM, Microsoft: Web services Transactions (WS-Transactions), http://www.ibm.com/developerworks/library/ws-transpec/, 2002.
[10] BEA, IBM, Microsoft: Web services Security (WS-Security), www-106.ibm.com/developerworks/webservices/library/ws-secure/, 2002.
[11] BEA Systems, IBM, Microsoft, SAP AG and Siebel Systems: Business Process Execution Language for Web services version 1.1, http://www-106.ibm.com/developerworks/library/ws-bpel/, 2003.
[12] Siemens Medical Web services: http://www.medical.siemens.com.
[13] Philips: http://www.medical.philips.com.
[14] GE Medical Systems: http://www.gemedicalsystems.com.
[15] Agfa Healthcare: http://www.agfa.com/healthcare/.
[16] Kodak Medical: http://www.kodak.com/global/en/health/.
[17] R. Anzböck, XR OPEN RIS Architektur, DATA Corporation, http://www.data.at, 2001.
[18] R. Anzböck, XR PACS Architektur, DATA Corporation, http://www.data.at, 2001.
[19] N.A. Kreider, B.J. Haselton, The Systems Challenge: Getting the Clinical Information Support You Need to Improve Patient Care, Wiley, John & Sons, Incorporated, 1997.
[20] E.L. Siegel, R.M. Kolodner, Filmless Radiology, Springer, 1998.
[21] H.K. Huang, PACS: Basic Principles and Applications, Wiley-Liss, 1998.
[22] B. Revet, DICOM Cookbook, ftp://ftp-wjq.philips.com/medical/interoperability/out/DICOM_Information/, 1997.
[23] H. Oosterwijk, DICOM Basics, OTech Inc/Cap Gemini Ernst and Young.
[24] Aalst: Interorganizational Workflows: An approach based on Message Sequence Charts and Petri Nets, citeseer.nj.nec.com/vanderaalst99interorganizational.html, 1999.
[25] ebpml.org: The Web service stack, http://www.ebpml.org/webservices.htm, 2003.
[26] BEA, IBM, Microsoft, TIBCO: WS-ReliableMessaging, http://www-106.ibm.com/developerworks/webservices/library/ws-rm/, 2003.
[27] R. Anzböck, S. Dustdar, Interorganizational workflow in the medical imaging domain, in: Proceedings of the 5th International Conference on Enterprise Information Systems (ICEIS), Kluwer Academic Publishers, Angers, France, 2003.
[28] R. Anzböck, S. Dustdar, Medical Web services workflows with BPEL4WS, http://http://www.infosys.tuwien.ac.at/Staff/sd/papers/MedicalServicesWorkflowsWithBPEL4WS.pdf, 2003.
[29] WS-Attachments: http://msdn.microsoft.com/library/en-us/dnglobspec/html/draft-nielsen-DIME-soap-01.txt, 2002.
[30] Microsoft: Direct Internet Message Encapsulation (DIME), http://msdn.microsoft.com/library/en-us/dnglobspec/html/draft-nielsen-DIME-02.txt, 2002.
[31] W3C, SOAP Messages with Attachments: http://www.w3.org/TR/2000/NOTE-SOAP-attachments-20001211, 2000.
[32] Base64 Content-Transfer-Encoding, RFC 2045, Section 6.8, IETF Draft Standard, 1996.
[33] G. Alonso, F. Casati, H. Kuno, V. Machiraju, Web services Concepts, Springer, 2004.

[34] D. Bunting, M. Chapman, O. Hurley, M. Little, J. Mischkinsky, E. Newcomer, J. Webber, K. Swenson, Web Services Composite Application Framework (WS-CAF), http://developers.sun.com/techtopics/webservices/wscaf/primer.pdf, 2003.
[35] D. Bunting, M. Chapman, O. Hurley, M. Little, J. Mischkinsky, E. Newcomer, J. Webber, K. Swenson, Web Service Context (WS-CTX), http://developers.sun.com/techtopics/webservices/wscaf/wsctx.pdf, 2003.
[36] D. Bunting, M. Chapman, O. Hurley, M. Little, J. Mischkinsky, E. Newcomer, J. Webber, K. Swenson, Web Service Coordination Framework (WS-CF), http://developers.sun.com/techtopics/webservices/wscaf/wscf.pdf, 2003.
[37] Collaxa Inc.: Collaxa BPEL Server 2.0: Reviewer's Guide, http://www.collaxa.com/pdf/cx-bpel-review-20.pdf, 2003.
[38] BEA, IBM, Microsoft: Web services Addressing (WS-Addressing), http://www-106.ibm.com/developerworks/webservices/library/ws-add/, 2003.
[39] D. Bunting, M. Chapman, O. Hurley, M. Little, J. Mischkinsky, E. Newcomer, J. Webber, K. Swenson, Web Service Transaction Management (WS-TXM), http://developers.sun.com/techtopics/webservices/wscaf/wstxm.pdf.
[40] S. Tai, T. Mikalsen, E. Wohlstadter, N. Desai, I. Rouvellou, Transaction policies for service-oriented computing, Elsevier Science Publishers, Data & Knowledge Engineering 51 (1) (2004) 59–79.
[41] S. Frolund, K. Govindarajan, Transactional conversations, W3C workshop on Web services: Position papers, http://www.w3.org/2001/03/WSWS-popa/paper50, San Jose, CA, USA, 2001.
[42] T. Mikalsen, S. Tai, I. Rouvellou, Transactional attitudes: Reliable composition of autonomous Web services, Workshop on Dependable Middleware-based Systems (WDMS 2002), part of the International Conference on Dependable Systems and Networks (DSN 2002), Washington D.C., USA, 2002.
[43] STRING Kommission, Magda-Lena 2 Richtlinie: http://www.akh-wien.ac.at/STRING/ (2000).
[44] WS-Trust, IBM, Microsoft, Verisign, RSA Security: www-106.ibm.com/developerworks/library/ws-trust/, 2003.
[45] WS-SecureConversation, IBM, Microsoft, Verisign, RSA Security: www-106.ibm.com/developerworks/library/ws-secon/ (2003).
[46] IETF, The TLS Protocol Version 1.1: http://www.ietf.org/internet-drafts/draft-ietf-tls-rfc2246-bis-05.txt, 2003.
[47] IETF, The IP Security Protocol: http://www.ietf.org/html.charters/ipsec-charter.html, 1995.
[48] IBM/Microsoft/SAP, et al., UDDI 3.0.1, http://uddi.org/pubs/uddi-v3.0.1-20031014.pdf, 2003.
[49] Mantell: From UML to BPEL, Model Driven Architecture in a Web services world, www-106.ibm.com/developerworks/webservices/library/ws-uml2bpel/, 2003.
[50] W3C, XPath: http://www.w3.org/TR/1999/REC-xpath-19991116, 1999.
[51] P. Wohed, W.M.P. van der Aalst, M. Dumas, A.H.M. ter Hofstede, Pattern Based Analysis of BPEL4WS, Department of Computer and Systems Sciences, Stockholm University/The Royal Institute of Technology, Sweden, 2003.
[52] Object Management Group: UML 2.0 Standard specification, http://www.omg.org, 2003.
[53] M. Fowler, K. Scott, UML distilled: applying the standard object modeling language / Martin Fowler with Kendall Scott Addison Wesley Longman (The Addison-Wesley object technology series), 1998.
[54] M. Dumas, A.H.M. ter Hofstede, UML Activity Diagrams as a Workflow Specification Language, in: Proceedings of the International Conference on the Unified Modeling Language (UML'2001), Toronto, Canada, 2001.
[55] O. Ratib, M. Swiernik, J.M. McCoy, From PACS to integrated EMR, http://www.elsevier.com/locate/compmedimag, Department of Radiology, UCLA School of Medicine, 2002.
[56] N. Lundberg, Impacts of PACS on Radiological Work, Department of Informatics, University of Gothenburg, 2000.
[57] J. Pereira, et al., Design and Implementation of a DICOM PACS With Secure Access Via Internet, A Coruna University, Spain, 2001.
[58] J. von Berg, J. Schmidt, T. Wendler, Business Process Integration for Distributed Applications in Radiology, Philips Research, in: Third International Symposium on Distributed Objects and Applications (DOA'01), Rome, Italy, 2001.
[59] S. Hludov, Chr. Meinel, G. Noelle, F. Warda, PACS for Teleradiology, medicineonline.de, 2000.
[60] R. Anzböck, S. Dustdar, Modeling Medical Web services, BPM 2004, Conference on Business Process Management, Springer LNCS 3080, 2004, pp. 49–65.

[61] R. Breu, M. Hafner, B. Weber, M. Alam, M. Breu, Towards Model Driven Security of Inter-Organizational Workflows, Institut for Informatics, University of Innsbruck, 2004.
[62] ISO: ISO Basic Reference Model for Open Systems Interconnection (OSI), 1995.
[63] Mirosoft, IBM, BEA, SAP: WS-Policy, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnglobspec/html/WS-Policy.asp, 2003.
[64] Microsoft, IBM, Verisign, RSA Security: WS-SecurityPolicy, http://msdn.microsoft.com/webservices/default.aspx?pull=/library/en-us/dnglobspec/html/WS-Securitypolicy.asp, 2002.
[65] DATA Corporation: DICOM compression techniques and broadband Internet access, 2004.
[66] ISO: ISO/IEC 15444-1:2000, JPEG 2000 image coding system, http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail? CSNUMBER = 27687 & ICS1 = 35 & ICS2 = 40 & ICS3 = (2002).
[67] XML-Encryption: W3C Working Draft, XML Encryption Syntax and Processing,, http://www.w3.org/TR/xmlenc-core/, 2002.
[68] XML Signature: W3C Proposed Recommendation, XML Signature Syntax and Processing,, http://www.w3.org/TR/2001/PR-xmldsig-core-20010820, 2001.
[69] BEA Systems: Bea Web Logic Server, www.bea.com/products/weblogic/server/index.shtml, 2004.
[70] Microsoft Biztalk Server 2004: Microsoft Corporation, www.mirosoft.com, 2004.
[71] O. Ratib, M. Swiernik, J.M. McCoy, From PACS to integrated EMR, Computerized Medical Imaging and 1020 Graphics (2003) 207–215.
[72] X.509—pkix charter: IEFT, http://www.ietf.org/html.charters/pkix-charter.html, 2005.
[73] Kerberos—krb-wg charter: IETF, http://www.ietf.org/html.charters/krb-wg-charter.html, 2004.

Rainer Anzböck received a Bachelor degree in Software Engineering and a Master degree in Technical Informatics from the Vienna University of Technology. Currently, he is a Ph.D. student working with Schahram Dustdar. Since 9 years he has been working at DATA Corporation, a software company for medical information systems. His research interests include distributed workflows, Web services collaboration and composition, as well as medical services modeling.

Schahram Dustdar is an Associate Professor at the Distributed Systems Group, Vienna University of Technology. His previous work experience includes several years as the founding head of the *Center for Informatics* (ZID) at the University of Art and Industrial Design in Linz (1991–1999) and as the director of Coordination Technologies at the Design Transfer Center in Linz (1999–2000). While on sabbatical leave he was a post-doctoral research scholar at the *London School of Economics* (Information Systems Department) (1993 and 1994), and a visiting research scientist at *NTT Multimedia Communications Labs* in Palo Alto, USA during 1998. Since 1999 he works as the co-founder and chief scientist of Caramba Labs Software AG (*CarambaLabs.com*) in Vienna, a venture capital co-funded software company focused on software for collaborative processes in teams. More information can be found on: *http://www.infosys.tuwien.ac.at/Staff/sd*.