Requirements for Interoperability in Healthcare Information Systems

Rita Noumeir
Department of Electrical Engineering École de Technologie Supérieure, 1100 Notre-Dame West, Montreal, Quebec, Canada, H3C 1K3
E-mail: rita.noumeir@etsmtl.ca

Submitted October 2011. Accepted for publication March 2012.

ABSTRACT
Interoperability is a requirement for the successful deployment of Electronic Health Records (EHR). EHR improves the quality of healthcare by enabling access to all relevant information at the diagnostic decision moment, regardless of location. It is a system that results from the cooperation of several heterogeneous distributed subsystems that need to successfully exchange information relative to a specific healthcare process. This paper analyzes interoperability impediments in healthcare by first defining them and providing concrete healthcare examples, followed by discussion of how specifications can be defined and how verification can be conducted to eliminate those impediments and ensure interoperability in healthcare. This paper also analyzes how Integrating the Healthcare Enterprise (IHE) has been successful in enabling interoperability, and identifies some neglected aspects that need attention.

Keywords: electronic health record, interoperability, integrating healthcare enterprise, IHE, software specification, requirements, testing, software verification

1. INTRODUCTION
The Electronic Health Record (EHR) improves the quality of healthcare by enabling access to all relevant information at the diagnostic decision moment, regardless of location, whether it is the point of access location or the location where the information was initially gathered and created. EHR also improves the efficacy and efficiency of healthcare processes by enhancing productivity with timely access to information and by reducing the duplication of tests.

EHR is a distributed system that results from the cooperation of several heterogeneous information systems. It is made of components, which are themselves systems or subsystems with regards to the EHR system. Information systems support healthcare professionals in their tasks and communicate with peers to exchange information pertinent to the healthcare process being executed. The main problem comes from the fact that the subsystems are usually standalone systems which have

their own behaviour to serve their own functions. These subsystems need to act jointly to serve the functions of the overall system. Therefore, heterogeneous healthcare information systems need to cooperate to implement the use cases of a specific healthcare process. Interoperability is essential. According to the Merriam-Webster dictionary, interoperability is "the ability of a system… to work with or use the parts or equipment of another system." The Healthcare Information and Management Systems Society (HIMSS) Integration and Interoperability Steering Committee [1] defines interoperability as "the ability of health information systems to work together within and across organizational boundaries in order to advance the effective delivery of healthcare for individuals and communities." This latter definition specifies that systems cooperate to "advance the effective delivery of healthcare" which we mean by "implementing a healthcare process." HIMSS definition also specifies that "systems work together within and across organizational boundaries," which is consistent with a "distributed system." In fact, regardless of whether the archiving of clinical information be centralized or distributed, the EHR is distributed because at least the source of the clinical information is inherently not a single system.

To achieve this joint action, a system-engineering process needs to be followed to gather the requirements of the overall system, to identify the functional requirements, to define non-functional constraints, to model the solution and to execute the evaluation process. Interoperability challenges in healthcare are important: healthcare systems have to deal with such extremely diverse clinical information as diagnostic images and laboratory or cardiology results; they also need to communicate utilizing various healthcare-specific standards [2], such as Digital Imaging and Communications in Medicine (DICOM) [3] and Health Level 7 (HL7). DICOM is a standard that deals with encoding and communicating medical images and imaging information. HL7 is a standard that deals with encoding and communicating such healthcare information as patient demographics, examination ordering and delivery of results. There are two versions of HL7: v2 and v3. Healthcare functional areas are covered by both versions of the HL7 standard; v3 is a model-based standard utilizing XML syntax, web technologies and better methodologies, and includes use cases and interaction models between subsystems.

Healthcare standards are necessary. Without them interoperability is not possible. However, they alone are not sufficient. In this paper, we analyze the obstacles to interoperability and show what is needed in addition to the standards to ensure it. We also analyze the requirements of the Integrating the Healthcare Enterprise (IHE), which aims to achieve interoperability [3]. IHE, a process started in 1998, was initially jointly sponsored by the Radiological Society of North America (RSNA) and the Healthcare Information and Management Systems Society (HIMSS). Currently, many other organizations sponsor IHE development activities.

IHE follows an incremental approach where care providers and end users identify annually the key business problems they face, in terms of use cases. Healthcare manufacturers and information technology experts agree upon a solution described as an "integration profile" and documented in Technical Frameworks. An IHE integration profile is described, with actors cooperating through standards-based transactions in

order to achieve clinical information exchange and workflow. An actor is specified by a set of application roles and responsibilities. IHE actors usually represent real-world subsystems.

IHE has also been a pioneer in healthcare testing by putting in place a testing event – the IHE connect-a-thon. To ensure the success of this live-testing event, participants test their implementation beforehand with interoperability testing software [5]. This testing tool consists of documents and software that simulate communication partners, in addition to providing test data and test plans. IHE has contributed to advanced interoperability in healthcare. In this paper, we analyze interoperability and discuss the methods and reasons that help IHE succeed or fail to solve integration problems.

Interoperability is not specific to healthcare. Research to better model and validate systems has been driven by problems from the manufacturing industry [6, 7] and electronic business [8, 9, 10]. More recently, the same kind of problems have been encountered while integrating governmental services [11], automating precast fabrication and construction [12, 13], and integrating electricity distribution [14]. The method followed in this paper was inspired by the analysis of integration problems encountered in the automotive industry [7].

In healthcare, besides the IHE process, such medical standards as DICOM and HL7 conduct processes mainly to develop standard information models and transactions. There are also several efforts to develop testing software and testing infrastructures. Recently, the National Institute of Standards and Technology (NIST) became responsible for leading the development of the conformance and interoperability testing infrastructure within the healthcare domain in the United States [15].

Interoperability is impeded by various difficulties. In this paper, we analyze interoperability and discuss these impediments. One of our main contributions is to identify what is needed to ensure interoperability in healthcare. We see the various standards as important blocks to be used. We do not analyze them; however, we provide examples to explain the impediments to interoperability and derive guidance to ensure interoperability. Moreover, we analyze how IHE has succeeded or failed to eliminate or reduce these impediments. We believe our work is fundamental not only to improve interoperability requirement definition, but also to achieve better interoperability and conformance testing [16].

2. METHODS

The manufacturing industry has undergone deep transformations during past decades. Companies have increased their efficiency by focusing on their specialized capabilities and by shifting their non-core functions to supplier organizations. Outsourcing ranges from the design and production of many parts to the after-sale maintenance of products. Integration between systems became crucial to support this business process. Lack of integration has been reported to be very costly for the U.S. automotive supply chain [7]. In order to investigate how to lower this cost, researchers at NIST have undertaken a project to investigate whether integration can be automated. They sorted a set of common integration problems into five categories [7]. These categories are based on the Reference Model of Open Distributed Processing (RM-ODP) [17], which defines

essential concepts necessary to specify open distributed processing systems, including the following categories:

1) Technical: Where integration aspects relate to the underlying communications, message structures and content, as well as control flow.
2) Semantic: Where integration aspects concern the consistent interpretation of the exchanged information, which requires an agreement on common concepts and terms used to refer to those concepts.
3) Functional: Where integration aspects concern behaviours of systems consistent with their roles in the overall process; these concerns include the objects to be acted on and the actions to be performed.
4) Quality: Where integration aspects concern the ability to support the business process in an acceptable way. These aspects include security, reliability, availability, accuracy and timeliness.
5) Logistical: Where integration aspects relate to the impact of the design on the overall system. These aspects include cost, flexibility and openness.

We describe, in the next section, the technical, semantic, functional, quality and logistical integration aspects. We believe these categories of integration concerns pertain to the healthcare domains. Analyzing these concerns helps define the requirements for interoperability. For each category, a list of impediments to interoperability is described. For each impediment, we give examples derived from the healthcare domain and provide guidance to mitigate it by detailing requirements at the specification and testing levels. In other words, we try to state what needs to be done to ensure interoperability.

Quality and logistical concerns are not commonly considered as impediments to interoperability, but based on our experience, we believe these concerns are important and will discuss them in the next section. Quality concerns are associated with how well the system performs its functions and whether this is acceptable to support the healthcare process. Quality concerns can be stated in the non-functional requirements of the system and specific testing tools need to be developed and used throughout the lifetime of the system to identify integration issues that can be observed at any time. Logistical concerns may influence the system design, but are not directly related to the functions of the system or to how well it performs them. Logistical concerns include flexibility and openness, which will also be discussed in the next section.

The analysis and recommendations we present with regards to quality and logistical concerns are derived from our involvement with Ontario's largest diagnostic imaging integration project across multiple hospital sites in the greater Toronto area and central Ontario [18]. The project involves five local health information networks and 21 hospitals representing approximately 39 sites. Diagnostic imaging exam volumes are expected to exceed an annual volume of three million exams in five years. The solution deployed involves information systems from seven different providers and requires integrating information generated by a very large number of existing imaging equipment and information systems.

3. RESULTS

In this section, we analyze the technical, semantic, functional, quality and logistical integration aspects. For each category, interoperability impediments are described and discussed; we give examples derived from the healthcare domain, and provide guidance to mitigate the impediment at the specification and testing levels.

3.1. Technical Concerns

3.1.1. Connection Impediment

This impediment occurs when there is a disagreement between components at the communication level, including the lower layer protocols (LLP). An obvious example is when peers are not using the same version of the standard, such as when one is using HL7 version 2.3 and the other is expecting HL7 version 2.5. Although newer versions of HL7 v2 are backward compatible, interfaces between systems may be necessary to remove new fields not supported by older versions. Likewise, using the DICOM standard, peers may not agree on the service-object pair (SOP) Class, i.e., the service (or action) to be executed on the information object. Another example occurs when one peer is using secure socket and the other does not support secure communication.

In order to eliminate this impediment, requirements for the communication protocol, its version, and the underlying LLP must be specified. Moreover, when the communication protocol allows for multiple possibilities, one specific possibility must be specified. For example, when security is required, one needs to specify what kind of certificate is to use, whether encryption is required and what type of encryption algorithms is supported. The testing software issues transactions using the specified protocol and the specified LLP to verify whether the connection could be established with the system under test and information could be exchanged.

IHE has succeeded in eliminating this impediment by specifying, for each transaction, the communication standard to be used, such as DICOM or HL7. Moreover, it specifies the version of the standard when multiple versions exist. For example, the Modality worklist query is achieved using the DICOM standard (Radiology [19]); the Patient Identification Query is achieved using HL7 v2.5 (IT Infrastructure [19]); the Patient Demographics Query v3 is achieved using HL7 v3 (IT Infrastructure [19]); the Provide and Register transaction is achieved using ebXML v 3 (IT Infrastructure [19]). IHE also specifies constraints on the lower layers, when various possibilities are permitted by the standard. It specifies, for example, that the Minimal Lower Layer Protocol (MLLP) is to be used with HL7 v2, and Web Services as the transport mechanism for HL7 v3 messages. As for Web Services, IHE specifies the version as well as the Simple Object Access Protocol (SOAP) version. Moreover, IHE specifies LLP information or behaviour that otherwise would have been ambiguous because multiple possibilities exist. Examples include specifying the use of Web Services Addressing at the SOAP level.

Being able to communicate with peers using specified communication standards is a prerequisite to implementing IHE profiles, and to execute interoperability testing. In fact, IHE testing assumes conformance to standards, without necessarily requiring conformance testing; in other words, IHE testing assumes that communication can take

place between peers. When using DICOM, for example, peers are expected to negotiate and agree on DICOM transfer syntaxes as well as on SOP classes; if image compression is negotiated and used, it is expected that compression and decompression are correctly performed by peers. Interoperability testing software needs to verify that communication successfully takes place and additional communication requirements are implemented as required. Such additional communication requirements may include, for example, the presence of web addressing at the SOAP level.

3.1.2. Syntactic Impediment

This impediment occurs when different data structures or representations are used between peers. Examples include the usage of different XML schemas or when one peer expects a specific field in an HL7 v2 message that the sending peer is not including. To eliminate this impediment, requirements about the encoding and structure of the exchanged information need to be specified. Exchanged information can be objects, such as images and documents, as well as messages. Although communication protocols specify encoding, additional requirements can be added to require the presence or the absence of optional information, or to require that specific information be encoded in specific fields when multiple possibilities are permitted by the standard. By specifying the structure of the exchanged information, it becomes possible to test for interoperability by verifying the structure of a message, the presence of all required data, and the correct structure of exchanged documents.

IHE has succeeded in eliminating this impediment by identifying the information needed to perform a specific task and by specifying accordingly the structure of the exchanged information. For example, it defines HL7 v2 message structures by usually requiring the presence of fields that are otherwise optional, or by specifying constraints on their cardinality. Obviously, fields that are required by the standard are also required by IHE. Thus IHE imposes additional presence constraints on message structures. IHE also specifies HL7 v3 schemas. As for exchanged documents, IHE specifies their encoding structures, such as specifying that the document should be a Clinical Document Architecture (CDA) encoded according to a specified schema, or a DICOM manifest. Furthermore, IHE specifies the encoding of specific types of information, such as specifying when binary data is to be encoded in a base64 format.

3.1.3. Control Impediment

This impediment occurs when peers do not agree on their roles or do not agree on the flow of control in a communication interaction. Examples of control impediments include the case where the initiating HL7 communication expects the response on the same communication channel and when this is not the case a new communication socket is opened by the peer to send the response. Another case we have encountered in practice is where one HL7 peer expected a channel to stay open all the time while the other peer closed the channel after the exchange was completed and reopened a new channel when a new exchange was required.

The DICOM standard usually specifies detailed communication control by requesting peers to deal with this issue, such as negotiating and agreeing on roles

(provider or user, for instance), independently of which peer initiated the communication. Whether two peers could agree on roles can be verified by examining their DICOM conformance statements, checking that they expect to play complementary roles. On the other hand, the HL7 standard does not specify control requirements because it concentrates on level seven of the Open Systems Interconnection (OSI) communication model that does not encompass control. Therefore, details about control need to be specified in the requirements. This has been achieved by IHE.

When multiple control possibilities or ambiguities exist, IHE specifies what is needed to avoid control impediments. Many examples of such specifications exist in the IHE technical framework. For example, when using HL7 communication, IHE requires immediate acknowledgment; it also requires that this acknowledgment occurs on the same network connection. Another example relates to HL7 query responses: IHE requires that the query response be part of the acknowledgement message of an immediate response. The usage of the continuation pointer is another example of control specification that IHE requires for the Patient Demographics Query transaction (IT Infrastructure [19]).

The interoperability testing software is expected to establish communication in accordance with the control requirements when they are clearly specified; moreover, it can verify that the communication peer also implements these control requirements. When special cases require special control, such as the usage of the continuation pointer, specific test scenarios can be dedicated to verify the correct implementation of such requirements.

3.1.4. Quality-of-Service Impediment

This impediment occurs when the behaviour of a communication peer does not satisfy technical requirements derived from "quality" concerns, such as a timely response to a communication request. In other words, the communication channel could not be established because of quality concerns. An example of this impediment occurs when a peer requests a query from another peer that is taking a long time to respond, so the initiating peer closes the connection because of a timeout. Another example is when communication cannot be established because the security certificate is expired or the system is down.

Quality-of-service impediments do not normally occur. However, a well-designed system usually takes these cases into account and implements exceptional software flow. To ensure these impediments are adequately considered, the system project manager should ask each component manufacturer to provide documentation about error handling. Moreover, the system testers need to simulate quality impediments to verify the behaviour of the peers. Furthermore, behaviour specifications are needed for handling errors that are considered critical to the overall system behaviour and state or data consistency. Examples include methods for dealing with events not delivered, queries not timely answered, conditions and maximum number of automatic retries, conditions for immediate failure without retries, and methods for alerting operating managers of situations which require human intervention.

This impediment has not received much attention so far. In most cases, assuming conformance to communication standards has been abusively considered sufficient. Unfortunately, IHE does not define requirements to deal with this impediment, leaving its solution to the overall system requirements for definition and testing. These requirements are not specific to any system in particular and could be defined along with the integration profile.

3.1.5. Data Consistency Impediment

This impediment occurs when peers do not consistently use information that is not directly communicated in the interaction (e.g., configuration data). Information that is not directly communicated can be of different kinds: the usage of specific code sets as allowed vocabulary in specific data fields; mapping of information, such as mapping a DICOM Application Entity (AE) to a specific TCP endpoint (IP address and a port number) or HTTP endpoint (Universal Resource Locator [URL]); mapping an instance of a concept to multiple other related concepts, such as mapping a radiology procedure to a set of modality acquisition steps.

In order to eliminate this impediment, a project manager should coordinate the definition of the information that needs to be shared by all peers and ensure that it is distributed and effectively shared. Shared information includes the list of all codes used by the various components. Such codes include the type of user, the type of document, the identification of institutions, the identification of exams, the identification of ID issuers, etc. Shared information also includes the addresses of peers along with their aliases. Usually, such information should be configurable within each component and documented in the component's design document. The project manager then needs to verify what is configurable by each component and ensure that the data is consistently shared.

The common data used by the testing software needs to be documented as part of the testing software to enable the system under test to use it. Therefore, the testing software is required to provide information about peers' addresses, procedure codes, document types and other codes that would be shared for the tests to succeed. The testing software would require and verify that the system under test uses this shared configuration information appropriately.

IHE does not usually specify allowed vocabulary content or relationships. However, IHE specifies when and how the consistency of non-communicated information is to be ensured, enabling testing software to provide and impose common non-communicated information on systems under test, and to test for the consistent usage of such information. For example, the IHE testing software documents the codes that are used.

3.2. Semantic Concerns

3.2.1. Conceptualization Impediment

This impediment occurs when communicating applications have incompatible representations of the same concept. Examples include how to describe an address, a person and a document. Other examples of concepts to be shared by all peers include the users' roles and the permissions associated with each role.

To eliminate this impediment, all concepts pertinent to the system should be described in a single document shared between the component's providers and the project management team. This is more than sharing common terminologies or ontologies. Each information concept needs to be described and mapped to the real word concept. Identification of the information concept needs to be detailed. Moreover, relationships between concepts need to be specified with the cardinalities. Sometimes unusual cases need to be specified and described, mainly when a zero instance of a concept is possible; an example of such case is when a radiology report exists without images related to it. As concepts, identifications and relationships are specified; they can be verified and validated by the interoperability testing software. This software may implement use cases where different concept relationships are verified. For example, the testing software can verify specific system behaviour where one report is related to two studies, two reports are related to a single study, one report has no study or one study has no report.

IHE explains, describes and specifies concepts along with their relationships. For example, IHE defines a radiology procedure and how it relates to a radiology request. Another example is the clarification of the radiology accession number. Most importantly, IHE specifies how concepts are uniquely identified and how related instances can be linked using identifiers. Examples include where to put identifiers in the message structure and how to encode them, such as where to include a patient identifier (ID) and how to specify the ID issuer. Concepts related to sharing images include the definition of how a manifest document is expected to reference images, as well as what identifications to use and where to encode them. Another important example about how IHE solves the conceptualization impediment is related to query keys: IHE specifies what keys are required for a specific query transaction, such as a modality worklist query (Radiology [19]), or a patient demographic query (IT Infrastructure [19]). HL7 v3 attempts to solve this impediment by introducing the Common Message Element Types (CMET) to express a common concept in a reusable message type fragment.

3.2.2. Conceptual Scope Impediment

This impediment occurs when an important concept is not communicated by one of the peers. One example is when a component, in order to enforce security constraints, needs to know specific information that is not available because it is not communicated by the other peer. For instance, a viewing application needs to decide whether it should hide the existence of results for a specific patient. If it enforces this constraint based on a VIP flag, the VIP information needs to be communicated with the patient demographics; otherwise the viewing application would be unable to enforce its confidentiality constraint.

To eliminate this impediment, technical representatives from component providers should meet and discuss important concept exchange. Then component interface documents need to be distributed to all peers. Each interface document needs to be discussed by the interface consumers to identify any missing concept needed within a specific information exchange. When required fields are identified and documented, their presence in the exchanged messages can be verified by the interoperability testing software.

By analyzing the information needed to conduct a specific task, IHE identifies the required information and adds presence requirements on otherwise optional data fields. Almost every IHE transaction has additional presence requirements. One example is the information returned in a query response where IHE details the fields required to be returned by the peer fulfilling the query.

3.2.3. Interpretation Impediment

This impediment occurs when the message has a different meaning for the receiver than for the sender, i.e., when the technical communication is completely successful, but the intent is not fulfilled. Examples of this impediment occur when the receiver of information is not capable of using it, such as when a component receives a medical report that includes a reference to an image without being able to display the image. In the imaging domain, a visualization workstation can receive an image without being able to display it, or without being able to apply some encoded transformation before displaying it. To eliminate this impediment, the specifications need to describe explicitly the actions expected from peers. Testing software can thus verify whether the expected actions have been carried out.

Usually, peers are required to trigger the exchange of specific information or change their internal states. Interoperability testing software can validate whether expected actions have been accomplished in mainly two ways: (1) if the system under test is required to trigger a communication, the testing software awaits and validates the communication content; and (2) if the system under test is required to change its internal state, the testing software triggers a transaction for the system under test and validates the response content. Change in internal state can be verified sometimes, such as in workflow managing systems where tasks are added to or removed from worklists that can be queried. It can be difficult to automatically test a change in internal status, as when a system is expected to mark a specific piece of information as persistent while there is no message that can be exchanged to request the deletion of that information, so effective persistency cannot be automatically checked.

IHE specifies the expected actions for every IHE transaction and for each peer involved in the transaction. Exceptions to the usual use case are also specified and detailed in the technical framework. Although HL7 v3 includes descriptions of expected actions, most standards do not explicitly describe expected behaviours. By explicitly describing expected behaviours when such descriptions are missing in the standard specifications, IHE specifications help eliminate this interoperability impediment.

3.2.4. Reference Impediment

This impediment occurs when the communicating applications use different systems of reference for identical concepts. Reference impediments are very common when the same concept is exchanged by multiple standards – these multiple standards need to reference the same instances of that concept. Examples include referencing of orders using DICOM and HL7, referencing instances of images in security logs, and referencing the same patient using different standards (e.g., HL7 v2, HL7 v3, DICOM and ebXML). One very common example is how to reference an imaging procedure,

whether this is done with the accession number or the procedure ID. Another very common example is how to reference the evidence (i.e., images) that was examined to generate the results.

To eliminate this impediment, all peers must agree on how to reference a specific concept and where to encode that reference. In a common architecture design document, all concepts should be described and references to every concept detailed. Interoperability testing software can validate that instances are consistently referenced using different transactions and different standards. For example, the testing software can validate whether the referenced image manifest inside an ebXML transaction is in fact the document that is published, or that the images referenced inside the manifest correspond to the instances intended to be published.

IHE describes concepts that are referenced within transactions. Moreover, IHE specifies the standard fields to encode the reference, such as using the HL7 v2 Filler Order Number to communicate the order accession number. Furthermore, IHE specifies how to map references from one standard to another, such as mapping HL7 v2 data fields into DICOM header fields. Requiring IHE integration profiles relieves the system architect from defining concepts and their references. Unfortunately, some projects build their system using IHE profiles but do not fully adopt the IHE concept definition and reference model. It is in everyone's interest to define the concepts and their references once and for all, and to not reinvent the wheel every time.

3.3. Functional Concerns

3.3.1. Functional Model Impediment

This impediment occurs when two applications have incompatible factorings of the process activity space: there may be a task that each expects the other to do (nobody's job), or a task that both expect to do themselves (overlapping roles). An example of a functional model impediment is when a list of patients is queried and the returned list is not supposed to include VIP patients. Which component is supposed to enforce this constraint? Is the query server supposed to filter out the VIP patients from the list returned in the query response? Or is the query client supposed to filter out the VIP patients before displaying the list to the user? If both components expect the other to filter out VIP patients, a confidentiality breach is possible due to a functional model impediment.

To eliminate this impediment, the responsibilities of each component need to be specified in a single architecture design document distributed to all component providers. When all the responsibilities of components are identified and described, the testing software can verify that every responsibility is fulfilled by the right component.

IHE helps eliminate the functional model impediment because, for every integration profile, IHE describes roles and responsibilities for each actor involved in the profile. Interoperability testing software simulates all peers needed to test a specific actor and tests all responsibilities associated with that actor, as required by the specific profile.

Moreover, IHE allows actor grouping. When grouping is permitted and when a system cannot delegate part of its behaviour to a peer, it is expected to fulfil the role of that peer completely. Furthermore, in cases when more than one possibility is permitted,

explicit requirements are carefully expressed to eliminate this impediment. In the Cross Enterprise Document Sharing for Imaging (XDS-I) [20] profile, for example, an Imaging Document Consumer can query the Imaging Document Source using either DICOM C-Retrieve or Web Access to DICOM Persistent Objects (WADO) transactions, but IHE requires the system receiving the query be able to accept both types of transactions.

3.3.2. Functional Scope Impediment

This impediment occurs when one party's behavioural model for a function contains more activities than the other party's model: (1) when the requester's model is larger than the performer's model, the performing application executes a subset of the expected behaviour, leaving some expected tasks not executed; (2) when the requester's model is smaller than the performer's model, the performing application executes unexpected activities as well as those requested.

The difference between the functional model impediment and the functional scope impediment is that the former relates to a function, needed in the overall system, that two components either both implement or do not implement at all, while the latter, the functional scope impediment, relates to a function of a single component that is either implemented but not wanted or the opposite, wanted but not implemented. An example of a functional scope impediment is when the receiver of a report is expected to automatically print the result but this function is not implemented, or the opposite, when it automatically prints the received result while this is not desired.

This impediment, like all functional impediments, arises because of implicit assumptions about components' functions. To eliminate this impediment, the components' required functions need to be specified. Moreover, every component needs to be tested for acceptance to identify functions that are present but are not desired.

IHE has invested great effort in recent years into specifying requirements to reduce functional scope impediments. Examples of such specifications relate to such profiles as Mammography Image and Nuclear Medicine Image display. Neither of these IHE profiles describes a message exchange between two actors; they describe the functions of one specific actor in detail. IHE has not succeeded in eliminating the functional scope impediment. For example, when IHE failure relates to the Modality Performed Procedure Step (MPPS) transaction where the expected behaviour of the receiving system is not specified (Radiology [19]), this transaction is usually successfully received but does not trigger actions or state change as implicitly expected.

When a communication succeeds, i.e., when there is no technical impediment and the exchanged message or information object is parsed correctly and used, and there is no semantic impediment, a functional scope impediment is possible if the behaviour of the receiving component is not completely specified. The difficulty arises when the behaviour is described but not sufficiently. Examples of such impediments can be found when one report creator actor creates a radiology report containing a reference to a key image, encodes the report as a DICOM-structured report and sends it to a receiving actor that receives the report, displays it, but does not display the referenced image. Yet another example is encountered when one component creates an image and encodes

multiple window-width and window-centre pairs to display various structures in the image, the receiving system succeeds in receiving the images and, displays them, but can display only one pair of window-width and window-centre coordinates.

3.3.3. Embedding Impediment

This impediment occurs when the behaviour of an application is affected by the attempted integration with other peers. This impediment may occur when a component bases a decision on data received from another component. Unless there is a guarantee the data will always be received consistently, there is a high risk that an embedding impediment occur. One example is when a component is relying on the study description field to make a decision, such as choosing a layout to display the study; the integrated system behaves correctly most of the time, but sometimes, when the study description is empty for instance, there may be an embedding impediment. This impediment may also happen when a component is expecting that other peers do not have quality impediments. For example, when a component is expecting to receive a patient registration message before receiving a new order for that patient, when both messages are sent by different peers and when the registration message is not sent in a timely fashion so the order is received before the patient is registered, there could be problems.

This impediment cannot be completely avoided with better specifications. System testing would help reduce the risk of such an impediment, but cannot eliminate it completely. The testing software cannot detect an embedding impediment per se, but performing the testing with the help of interoperability testing software simulates the integrated environment in which the application is supposed to operate in a real situation; therefore, an embedding impediment would eventually be fixed before deployment. System testing cannot be effective in eliminating this impediment unless test scenarios are explicitly designed to verify it. Identifying potential embedding impediments is difficult. Analysis of a component's behaviours that depend on external parameters (e.g., external data, external events and external sequencings) allows the identification of embedding impediment risks and the development of specific testing.

3.3.4. Intention Impediment

This impediment occurs when the application is being used in a way its design did not anticipate, resulting in unexpected behaviours. This relates to differences in the details of the component specification versus the specification as needed for the role of that component in the larger system.

This kind of impediment is hard to grasp. For example, such an impediment is encountered in the way x-ray images are organized into series: some acquisition equipment may group multiple x-ray images into one series, while another may put each image in a different series. Although both have the right to do so, the receiving system may not be able to function with one or the other type of image grouping. This impediment may also occur when a visualization workstation is not designed to handle different modality images as part of the same study, such as a scanned image of the radiology order along with a series of computed tomography (CT) images. Another

example is when a visualization workstation receives axial CT images and assumes that they are numbered incrementally, from the head down to the toes. When a modality sends the images ordered in the opposite direction, this assumption no longer holds. When a visualization workstation assumes all images in a series have the same size, an impediment exists if the modality equipment sends a scout image as part of a CT series, because the scout image is usually bigger.

An intention impediment occurs when the specifications did not anticipate the case; therefore, the specifications are insufficiently detailed and implicit design assumptions are made without documentation because they seem obvious. Although hard to grasp, real world examples of non-interoperability due to this impediment are numerous; they are also very hard to fix because they can occur any time during the lifecycle of the overall system.

To help mitigate intention impediments, specifications need to be very detailed and cover all requirements that are otherwise implicit. IHE recently started to devote time to creating integration profiles that specify behaviour without transaction exchanges. Examples include specifying behaviours related to the display of nuclear medicine or mammography images (Radiology [19]). By specifying how to present the images to the user (presentation intention), these profiles help reduce intention impediments.

3.4. Quality Concerns

3.4.1. Security Impediment

Security concerns relate to various aspects, including information integrity, information protection from destruction, and protection from unauthorized access. Although the archiving of information may be centralized, information is distributed over several systems. For example, a system generates clinical information and sends it to a central archive; another system accesses the information for display. In this simple scenario, information is transferred between systems and needs to be transferred in a secure way to protect its integrity and to protect it from non-authorized access.

Requirements may differ depending on the underlying network architecture. Data encryption is needed if the communication takes place over unsecure open infrastructure, while data encryption may not be needed if the network is protected and closed. Because encryption comes with a complex overhead that affects performance, it affects maintenance cost, requiring the management of certificate distribution. Protection from unauthorized access requires authentication of users at every access point, and the validation of the access right of the authenticated user. It also requires the deployment of access logs to account for non-repudiation. User authentication and access control is difficult in healthcare, mainly because of the large number of subsystems and access points and because the denial of access may hinder the safety of the patient. Although several mature technologies can provide mechanisms to deploy authorization and control access, specific roles and exceptions need to be designed by specialized security engineers early in the deployment process. Moreover, logs need to be implemented and, even more importantly, the archiving of logs needs to be centralized and monitored to detect abnormalities and attempted security breaches. Monitoring is almost impossible if logs are not

centralized. Furthermore, monitoring is impossible without automation. Tools that automatically look for specific patterns in logs are currently available. However, effort needs to be deployed to implement such tools, operate them, decide what patterns to look for, and to configure the tools accordingly. As for the safe archiving of data, special attention is required when deciding the architecture infrastructure. Of course, long-term archiving needs to be robust with regards to hardware failures. Short-term archiving also needs to be robust when the information is not yet under the responsibility of the long-term archive. Moreover, the information should not be unadventurously destructed before archiving responsibility is transferred to the long-term archiving subsystem.

IHE provides specification and testing tools that help deploy subsystems which cooperate in a secure manner. The Audit Trail and Node Authentication (ATNA) integration profile requires subsystems to generate specific standardized log messages and to use specific encryption schemes with the underlying communication protocols (DICOM, HL7 and HTTP). The Cross-Enterprise User Assertion integration profile (XUA) specifies how the Security Assertion Markup Language (SAML) can be used to provide single sign-on. Special data fields are also provided as part of the Cross Enterprise Document Sharing (XDS) [21] data model to specify confidentiality constraints. DICOM and HL7 also provide special data fields that can be used to exchange information about confidentiality constraints with respect to specific data objects. Although the technology permits the implementation and testing of security mechanisms, specific security design is required. This should encompass security policies, role definition and permissions, and, more importantly, mutual vocabularies and security enforcement rules between subsystems. One problem we have encountered is that logging messages were not transferred over reliable connections. In the project we were involved with, additional requirements have been added to send logs with the reliable Syslog Internet standard and to require that the sending subsystem be responsible for the safe persistency of such messages, as far as the reception of those messages was not ensured.

3.4.2. Correctness Impediment

This describes concerns about the quality of data in the system: How close to the true state of the business is the information? To illustrate this concern, consider a case where a radiologist is looking at the images of a specific study: How can this radiologist be sure the displayed images are complete and there is no missing image? Another example is derived from a day-to-day physician worklist: How can one be sure all workitems are part of the worklist and all information related to a specific workitem is examined?

In healthcare, correctness is critical. Correctness mainly means completeness, ensuring no information is missing; it also means the information is associated with the correct patient. The healthcare system is supposed to be 100% correct; i.e., the information is totally correct and complete. There is no way of automating the testing for this concern. Usually, attention is required to display the number of exams, the number of images, etc., so that completeness can be checked by humans. It also requires

the display of patient's name and ID with any displayed information. Testers need to be aware of this concern and human observation is required to verify it.

3.4.3. Timeliness Impediment

In this context, timeliness is when the system is capable of executing its functions fast enough to support the healthcare process. Each function of every subsystem should execute fast enough. However, the timeliness of the overall integrated system depends not only on the timeliness of each subsystem, but on the network speed and the overall load as well. The expected load, when estimated, can help decide the architecture and the number of parallel components to deploy in order to split the load over multiple parallel nodes with the help of load balancers, as far as the components can run independently on such architecture. Special attention is needed when planning the project: (1) to estimate the load, and (2) to choose components that can run simultaneously on multiple nodes.

To ensure timeliness, one should be able to measure it when the system is up and running. Several actions can be considered to ensure timeliness: maximum acceptable response time can be specified for each operation; for every operation and subsystem involved in an operation, performance messages can be logged where each message contains start and stop times along with information about the operation parameters; messages can be generated and saved in specific performance logs to allow monitoring software to perform measurements or calculations about the overall timeliness and the time distribution among subcomponents. Moreover, pre-cached information can be utilized whenever possible; for example, recently accessed information can be kept in short-term storage for fast retrieval. We have also found very practical the use of a software tool specifically designed to simulate a large number of simultaneous accesses. This software can be used to verify performance and function when a large number of users are accessing the system simultaneously. It can also be used to measure timeliness of the system in production. Because specific performance logging affects performance and because the timeliness measurement is not necessarily continuous, performance logging needs to be toggled on and off.

3.4.4. Reliability Impediment

The reliability concern is expressed in terms of availability, timeliness, continuity and maintenance of state. Availability is ensured by putting in place an architecture where there is no single point of failure, such as deploying redundant components on redundant hardware, and by ensuring that information is redundantly stored. The basic idea is that if a component, hardware, or a communication path fails, there is an alternate path that would be able to perform the required healthcare request. Although timeliness has been discussed, it is related to availability because when the load on the available components and paths is balanced, timeliness is improved. Availability comes with a cost as hardware and components hardware need to be redundant. This is project-specific and must be addressed at the very beginning of the system design in order to choose components able to run on different nodes and able to transparently fulfill requests, regardless of the underlying network topology.

Furthermore, it is necessary to restart the component or node that failed. This is ensured first by knowing where the failure is. Monitoring and restarting can be automated. Usually a component can be monitored by sending it a simple "heart beat" request to which it responds. One example of this is sending a C-Echo request to a DICOM peer. Another is sending a simple HTTP request to a web application. This can be achieved with the help of a script that is executed periodically by a monitoring application. If a failure is observed, another script could be run to restart the component. Many such monitoring applications exist, whether as freeware or commercially licensed. However, the scripts for monitoring and restarting are usually delivered by the component manufacturer.

Continuity is the ability of the system to finish a started task before it fails, or to resume it after a failure. An example is when a component that has recorded the physician's interpretation fails while sending the result to a peer. What happens to the result? Will the result be received by the peer? Will the peer receive multiple copies of the result? How the receiving peer will react in case multiple copies are received? Continuity can be tested by identifying the critical tasks and by choosing components able to provide continuity for those tasks. This is achieved by studying the design of the components executing the task and simulating failure to test and evaluate continuity. Critical tasks include result delivery, information update notification (e.g., registering or updating patient's demographics), change to shared worklists (e.g., adding a new workitem), security monitoring (e.g., logging a security breach message). Ideally, messages that need to be exchanged between two different components can be put in a queue that is persistent: if the exchange fails, the item in the queue is resent. Moreover, the system receiving the information should be able to ignore information received more than once. The number of retries needs to be limited; when this limit is exceeded, the item should be tagged as "failed" so a human can be alerted to fix the problem.

Maintenance of state is the ability of the system to retain its information when a component or network path fails. This implies that each component be able to recover the state it had before it failed, and that the state of the whole system stay consistent. Important states need to be persistent within each component. One example is a result that is in a "pending for approval" state. Another is whether a result or information object has in fact been delivered to a specific receiver. Critical states need to be persistent within components. When the state is changed after a communication with a peer, the success or failure of that communication needs to be considered before the state is changed.

The IHE technical framework contains very few examples where maintenance of state is addressed. Examples include the "storage commitment" transaction that enables a component to transfer the storage responsibility of a specific image to another component and to wait for the notification of success. Another example is encountered with the "provide and register" transaction issued by a document source actor to publish a document to a document repository and document registry pair. For the latter transaction, IHE specifies atomicity, such as when the transaction between the repository and the registry fails, the whole transaction is supposed to fail, making the source component aware of the failure.

3.4.5. Version Impediment

Version impediments arise when a change or a revision to one component causes incompatibilities for its usage with the rest of the system. This impediment is hard to prevent when different components rely for their functioning on single third-party software. A common example is relying on Java. If two components are installed on the same node where one component is updated and updates the Java virtual machine, the other component may fail due to the Java update. Another is when two different components are deployed in the same web server: both components rely on the web server version. To overcome this impediment, each component needs to manage and rigorously maintain all the third-party libraries and versions it relies on to function. Moreover, the list of the infrastructure components with their versions must be identified and maintained.

With the wide use of the web, version impediments are more likely to happen because it is not possible to control the platform (i.e., the operating system and browser) that will be used by the user to access information. Of course, various platforms need to be tested for compatibility and a list of tested platforms needs to be made available to the user. New platforms will inevitably require that the system adapt and change. We are far from the old era when a healthcare system hardly ever changed during its lifetime.

3.5. Logistical Concerns

3.5.1. Flexibility Impediment

Flexibility is the ability of the system to support minor changes in the data or in the process. One example is the ability to change access control privileges or to implement additional roles; another is the ability to change the messages displayed to the user; and yet another example is to change rules affecting the process, such as what exam type is considered relevant, prior to an exam that needs to be interpreted.

Testing flexibility consists of evaluating the cost in time and effort required to implement projected changes. This can be achieved by examining the configuration documentation that component providers make available, and by identifying possible future changes and adding specific flexibility requirements. An example of the latter is requiring that a component provide an Application Programming Interface (API) for later integration with other applications.

3.5.2. Autonomous Change Impediment

Autonomous change is the ability of the system to accept minor changes in its components. Stated differently, how are components allowed to change without affecting the system? This is directly related to component openness.

When integrated into the system, peers rely on the component's published specifications. Although complete interface specifications do not necessarily eliminate integration impediments, depending on undocumented behaviour increases risks from future unexpected changes. To increase the level of confidence that the system will not incur major changes from changes in its components, interface and design documents for each component must be maintained and shared by all peers.

4. DISCUSSION

Interoperability cannot be achieved without ensuring that all impediments are addressed and resolved. Therefore, we have provided, for each concern, recommendations regarding specifications or management best practices to mitigate them. Impediments and recommendations are summarized in Table 1.

We have shown that many of these concerns have been successfully resolved by most of the IHE integration profiles. We have discussed how this was achieved. By describing use cases, by defining actors, their responsibilities and behaviours with respect to a specific cooperation, by defining concepts, and by imposing constraints on underlying communication standards, IHE has provided specifications that succeed in eliminating most of the technical, semantic and functional integration impediments. However, IHE did not address the technical quality-of-service impediment, the functional scope, the embedding and intention impediments. Besides security impediments, IHE did not address quality or logistical concerns at all.

Quality-of-service impediments could be addressed by IHE integration profiles by specifying actor behaviours for handling errors critical to the behaviour, state or data consistency of the system, such as how to deal with messages that are not timely delivered or not delivered due to connection failure.

Functional scope impediments could also be addressed by IHE integration profiles. Functional scope impediments are due to implicit assumptions about components' functions and relate to functions implemented but not wanted or wanted but not implemented. Eliminating functional scope impediments require more detailed functional specifications that can be provided as part of IHE profiles. On the other hand, verifying this impediment is particularly difficult, in the sense that it not only consists of testing required specifications but also identifying the presence of undesired functions.

Embedding impediments cannot be systematically addressed. Moreover, they can occur at any time, even after the system is verified and is in operation. Therefore, they are expensive to solve as they usually require modifying the components. Component designers and implementers need to identify when decisions depend on external data or external events, to document these cases and to provide test scenarios tailored to verify them. IHE specifications can help reduce this impediment by adding requirements related to the sequencing of events when necessary.

Intention impediments are hard to prevent; they can occur at any time, even after the system is tested and in operation, because the component may face cases that were underspecified, resulting in unexpected behaviours. To prevent this impediment, specifications are needed to detail requirements that are otherwise implicit.

Besides security for which IHE has provided several integration profiles, quality and logistical concerns are to be addressed on a per project basis. Quality requirements, such as timeliness and availability, need to be specified for the system. The system architecture must then be defined and described in an architecture design document in which the communication infrastructure is described. Monitoring requirements need to be identified. If the system is expected to run without

Table 1. Interoperability impediments with recommendations for mitigation

Impediments	Recommendations
	Technical
Connection	Specify the communication protocol, its version and the underlying LLP, security, certificate type, encryption algorithms and addressing.
Syntactic	Specify the encoding and structure, presence or absence of optional information, specific information to be encoded in specific fields when there are multiple possibilities, XML schemas, and whether binary data should be encoded with base64.
Control	Specify roles, acknowledgment models, whether acknowledgments occur on the same network connection, whether query response is part of the acknowledgement message and the utilization of the continuation pointer.
Quality-of-service	System project manager should ask each component's manufacturer to provide documentation about error handling, such as dealing with events that are not delivered, queries not timely answered, conditions and maximum number of automatic retries, conditions for immediate failure without retries, and methods for alerting operating manager of situations that require human intervention.
Data consistency	System project manager should coordinate the definition of the information that needs to be shared by all peers and should ensure it is distributed and effectively shared: list of all codes, type of users, type of documents, identification of institutions, identification of exams, identification of ID issuers, addresses of peers. Project manager also needs to verify what is configurable by each component and ensure the data is consistently shared.
	Semantic
Conceptualization	Describe all concepts in a single document that is shared among component providers and project management team; concepts need to be mapped to real word concepts. Identification of the information concept must be detailed; relationships between concepts are specified with cardinalities; unusual cases are specified and described mainly when a zero instance of a concept is possible.
Conceptual scope	Technical representatives from component providers should meet and discuss important concept exchange. Component interface documents must be distributed to all peers. Each interface document needs to be discussed by the interface consumers to identify any missing concept needed within a specific information exchange.
Interpretation	Specifications must explicitly describe expected actions from peers.
Reference	Peers must agree on how to reference a specific concept and where to encode that reference. In a common architecture design document, all concepts should be described and references to every concept detailed.

(*Continued*)

Table 1. Interoperability impediments with recommendations for mitigation (*Continued*)

Impediments	Recommendations
	Functional
Functional model	Specify responsibilities of each component in a single architecture design document distributed to all component providers.
Functional scope	Specify required functions for components. Each component needs to be tested for acceptance, to identify any functions present but not desired.
Embedding	This impediment cannot be avoided with better specifications. Analyze a component's behaviours that depend on external parameters (i.e., external data, external events, external sequencings) to identify risks and develop specific test scenarios explicitly designed to verify this impediment.
Intention	Because this impediment occurs when specifications did not anticipate the case, specifications need to be very detailed and cover all requirements that are otherwise implicit.
	Quality
Security	Specific security design is required, including policies, role definition and permissions and, more importantly, mutual vocabularies and security enforcement rules between subsystems.
Correctness	Display the number of exams, the number of images, etc., so that completeness can be checked by humans. Display patient's name and ID along with any clinical information.
Timeliness	Specify maximum acceptable response time for each operation; for every operation and subsystem involved in an operation, log performance messages containing start and stop times along with information about the operation parameters. Monitor software to quantify the overall timeliness and time distribution among sub-components.
Reliability	Require that components are able to run on different nodes and transparently fulfil requests regardless of the underlying network topology.
Version	Document and manage all third-party libraries and versions, and infrastructure components with their versions. For web applications, test operating system and browsers for compatibility and mail a list of tested platforms available to the user.
	Logistical
Flexibility	Require and examine the configuration documentation that component providers make available, identify possible future changes and specific flexibility requirements.
Autonomous change	Maintain and share with all peers the interface and design documents for each component.

interruption, redundancy must be planned and monitoring automated by using special monitoring software. The component design documents need to be studied to identify whether they can operate within the planned architecture, on the proposed infrastructure. The documents also must be examined to ensure that logistical concerns are mitigated.

Although IHE provides testing software and runs testing events where peers can test their components according to the IHE integration profile, system testing is needed. Evidently, when the system's components are tested as part of an IHE testing event, it is expected that system testing would be easier, because the component engineers are familiar with interoperability testing and because a large number of the impediments identified here are addressed by IHE testing software and testing events. However, the system needs to be tested to verify that all impediments are solved, including the ones covered by IHE testing and those not covered. Testing of quality concerns is also important. As it is not addressed by IHE, quality concerns require building software tools specifically for the project. Quality concerns must be evaluated before the system comes alive and then periodically after it starts operating. Specific software needs to be maintained and test data and test cases need to be identified and maintained as well.

Due to the importance of testing for interoperability, it is attracting attention and effort [22, 25]. However, testing so far has lacked detailed specifications with regards to exactly what is tested. Additional documentation needs to accompany the test scenarios, test data and test software. The additional documentation should detail what impediments are verified, along with a description of how this is achieved, such as what data fields, what quality-of-service issues and what action flows are verified. Moreover, because the life of the system is expected to span several years, testing tools must be maintained in order to run the regression tests required throughout the system's life.

5. CONCLUSION

To derive what is needed to achieve interoperability in healthcare information systems, we have described the interoperability impediments that are grouped into five categories:

1) Technical impediments that relate to problems in communication and process flow. These problems include connection, syntactic, control, quality of service and data consistency.
2) Semantic impediments that relate to understanding exchanged information. These problems include conceptualization, conceptual scope, interpretation and reference.
3) Functional impediments that relate to the difference between the expected and the actual behaviours of a component. These problems include functional model, functional scope, embedding and intention.
4) Quality impediments that relate to how well a component performs its function. These problems include security, correctness, timeliness, reliability and version.
5) Logistical impediments that relate to the impact of the design on the system. These problems include flexibility and autonomous change.

To help understand these impediments, we have provided concrete examples from the healthcare domain. Interoperability cannot be achieved without ensuring that all these impediments are addressed and resolved. We have also described testing requirements for each integration impediment and have discussed functional scope and correctness impediments where testing may not be automated but would require human observation. We have emphasized how tests can be designed and more importantly linked to impediments.

In conclusion, when defining the system requirements, designing or testing the system, impediments described in this paper need to be addressed, one by one, to ensure that the resulting system will effectively and successfully support the healthcare processes.

ACKNOWLEDGMENTS

This work was supported by the Natural Sciences and Engineering Research Council of Canada.

CONFLICT OF INTEREST

The author indicated no potential conflicts of interest.

REFERENCES

[1] The HIMSS Integration and Interoperability Steering Committee (I&I), Interoperability Definition and Background. 2005. http://www.himss.org/content/files/interoperability_definition_background_060905.pdf. Accessed October 18, 2011.

[2] Muñoz P, Trigo JD, Martínez I, Muñoz A, Escayola J, García J,The ISO/EN 13606 Standard for the Interoperable Exchange of Electronic Health Records, *Journal of Healthcare Engineering* 2011, 2(1):1–24.

[3] The Digital Imaging and Communications in Medicine (DICOM) standard. DICOM Homepage. http://medical.nema.org. Accessed October 18, 2011.

[4] Noumeir R. Integrating the Healthcare Enterprise Process. *Int. J. Healthcare Technology & Management*. 2008. 9:167–180.

[5] MESA software, IHE Test Tools. http://ihedoc.wustl.edu/mesasoftware/index.htm. Accessed October 18, 2011.

[6] NIST Manufacturing Business to Business (B2B) Interoperability Testbed. http://www.mel.nist.gov/msid/b2btestbed. Accessed October 12, 2010.

[7] Barkmeyer EJ, Barnard Feeney A, Denno PO, Flater DW, Libes DE, Steves MP, Wallace EK. Concepts for Automating Systems Integration. NIST Interagency/Internal Report (NISTIR). 2003. 6928.

[8] Junho S, Jaewook K, Ivezic N. Application information mapping test: an efficient content-level semantic equivalence test procedure for B2B integration. *International Journal of Computer Integrated Manufacturing*. 2009, 22(10):976–986.

[9] OASIS ebXML Implementation, Interoperability and Conformance Technical Committee. ebXML Test Framework Committee Specification, version 1.1. 2004.

[10] Dongsoo K, Jung-Hee Y. Development of an ebXML Conformance Test System for e-Business Solutions. *Lecture Notes in Computer Science*. 2003, 2738:145–154.

[11] Bednar P, Furdik K, Kleimann M, Klischewski R, Skokan M, Ukena S. Semantic integration of eGovernment services in Schleswig-Holstein. *Lecture Notes in Computer Science. 7th International Conference Electronic Government – EGOV 2008*. 5184:315–327.

[12] Jeong YS, Eastman CM, Sacks R, Kaner I. Benchmark tests for BIM data exchanges of precast concrete. *Automation in Construction*. 2009, 18(4):469–484.

[13] Eastman CM, Jeong YS, Sacks R, Kaner I. Exchange model and exchange object concepts for implementation of national BIM standards. *Journal of Computing in Civil Engineering*. 2010, 24(1):25–34.

[14] Lambert E. Return of experience regarding use of IEC CIM standard in distribution. IEEE Power & Energy Society General Meeting. 2008, 5.

[15] National Institute of Standards and Technology. NIST Health IT Standards and Testing web site. http://healthcare.nist.gov/index.html. Accessed October 12, 2011.

[16] Saboor S, Hörbst A, Ammenwerth E. Quality of Electronic Health Records - Coverage of Potential Information Weaknesses by Major EHR Quality Seals, *Journal of Healthcare Engineering* 2011; 2(2):365–388.

[17] Open Distributed Processing: Reference Model – Part 1: Overview. International Organization for Standardization, Geneva, Switzerland, 1998. ISO/IEC IS 10746 ITU-T X.900.

[18] Work begins on Ontario's fourth and final diagnostic imaging repository. https://www.infoway-inforoute.ca/lang-en/about-infoway/news/news-releases/745-work-begins-on-ontarios-fourth-and-final-diagnostic-imaging-repository. Accessed October 12, 2011.

[19] IHE Technical Framework and supplements. http://www.ihe.net/Technical_Framework/index.cfm. Accessed October 18, 2011.

[20] Noumeir R, Pambrun JF. Images within the Electronic Health Record. IEEE International Conference on Image Processing. 2009, 1761–1764.

[21] Noumeir R. Sharing Medical Documents and Images: Architecture and Communication Infrastructure. *IEEE IT-Professional*. 2011, 13(4):46–52.

[22] Namli T, Aluc G, Dogac A. An interoperability test framework for HL7-based systems. *IEEE Transactions on Information Technology in Biomedicine*. 2009, 13(3):389–399.

[23] Vega DE, Schieferdecker I, Din G. Design of a test framework for automated interoperability testing of healthcare information systems. *2nd International Conference on eHealth, Telemedicine, and Social Medicine*. 2010, 134–140.

[24] Noumeir R, Bérubé R. IHE cross-enterprise document sharing for imaging: interoperability testing software. *Source Code for Biology and Medicine*. 2010, 5(9):9–15.

[25] Gebase L, Snelick R, Skall M. Conformance testing and interoperability: a case study in healthcare data exchange. *Proceedings of the 2008 International Conference on Software Engineering Research & Practice*. 2008, 143–149.