An Interoperability Architecture for the Health Information Exchange in Rwanda

Ryan Crichton[12], Deshendran Moodley[1], Anban Pillay[1], and Christopher J. Seebregts[123]

[1] Health Architecture Laboratory, Centre for Artificial Intelligence Research, University of KwaZulu-Natal and Council for Scientific and Industrial Research, Durban, South Africa
[2] Jembi Health Systems, Cape Town and Durban, South Africa,
[3] Medical Research Council, Cape Town, South Africa

Abstract. Rwanda, one of the smallest and most densely populated countries in Africa, has made rapid and substantial progress towards designing and deploying a national health information system. One of the challenging aspects of the system is the design of an architecture to support: interoperability between existing health information systems already in use in the country; incremental extension into a fully integrated national health information system without substantial re-engineering; and scaling, from a single district in the initial phase, to national level without requiring a fundamental change in technology or design paradigm. This paper describes the key requirements and the design of the current architecture using ISO/IEC/IEEE 42010 standard architecture descriptions. The architecture is based on the Enterprise Service Bus architectural model. We also describe a partial implementation of the architecture, and give a preliminary analysis based on our experiences.

Keywords: interoperability; national health information system architecture; enterprise service bus

1 Introduction

The current landscape of health information systems, especially in the developing world, is mostly characterized by fragmented, piecemeal applications deployed by multiple organizations or contractors[1,4]. Applications are mostly custom built around specific outcomes, using different architectures and technologies, with interoperability low on the list of priorities. While these systems may be useful in a specific domain, their integration into a coherent national health information system (NHIS) poses serious challenges, limiting the usefulness and sustainability of these applications. One potential solution to this problem is to implement a national Health Information Exchange (HIE) and interoperability architecture to facilitate the integration of such applications into the NHIS.

In our previous work[16] we identified general challenges and requirements for designing and developing NHIS architectures in developing African countries.

In this paper we identify specific interoperability challenges and requirements for the Rwandan NHIS and describe the design and implementation of an interoperability architecture that has been adopted in Rwanda for use in its NHIS. In section 2 we describe the background to the Rwandan NHIS. Section 3 provides the key requirements and challenges for interoperability. The HIE interoperability architecture is presented in section 4 and section 5 gives an analysis of this architecture. We draw our conclusions in section 6.

2 Background: A national health information system for Rwanda

The Rwanda Ministry of Health (MoH) has already made significant progress in developing their NHIS, including community health systems, health management information systems and the national rollout of an electronic medical record application[15]. The Rwanda Health Enterprise Architecture (RHEA) project, led by the Rwanda MoH and supported by a consortium of partners and donors has developed an architecture to contribute to the development of the NHIS. One of the main deliverables is the architecture for a HIE to support the Rwandan NHIS in order to facilitate interoperability between individual health information systems and applications.

Implementation of the Rwandan HIE will be achieved in several phases. The first phase will implement foundational services and improve interoperability between two point of care information systems supporting maternal health in the Rwamagana district in Rwanda, including 13 health centers. The two supporting systems are OpenMRS[14,2,21], an Electronic Medial Records (EMR) system which is being implemented and maintained by the Rwandan MoH and RapidSMS, an SMS based data collection tool that is currently being used by community health workers in Rwanda. RapidSMS allows community health workers (CHWs) in Rwanda to submit maternal and child health information to a central server using SMS based messages from mobile phones. There are many CHWs within Rwanda and this information plays an important role in monitoring the progress of pregnant women and the health of children where frequent visits to clinics are not possible.

An important feature of the HIE interoperability architecture is that it will serve as the foundation for the HIE in Rwanda and enable other systems currently implemented in the country to connect and inter-operate more easily. The HIE plans to promote data re-use between the connected systems and to facilitate information sharing. It also aims to provide patients with a continuity of care record[9] to enable access to a patient's clinical information from different health facilities, improve the tracking of patients and reduce patients lost-to-follow-up.

The first phase involves deploying the HIE interoperability architecture combined with a set of foundational infrastructure services, providing services to point of care applications, initially, OpenMRS and RapidSMS. The HIE will allow them to share clinical information and ensure that shared information

uniquely identifies the patient, provider and facility within the information exchange (Figure 1).

The foundational infrastructure services are:

- Shared Health Record
  - This system persists and responds to queries for an appropriate subset of the patient's longitudinal, patient-centric medical record.
- Client Registry
  - This system persists and responds to queries for a patient's demographic and identifying information used to uniquely identify patients.
- Facility Registry
  - This system persists and responds to queries for data of the facilities participating in the information exchange. This is primarily used to maintain current and valid facility codes required in transactions.
- Professional Registry
  - This system persists and responds to queries for information about health care professionals who work at participating health care facilities in the information exchange. This is primarily used to uniquely identify health care professionals within the HIE.
- Terminology Service
  - This system stores all the clinical code systems (eg. LOINC, ICD10 and country specific code systems) that will be used within the HIE and facilitates verification and mapping between codes. It exposes endpoints that allow codes to be verified against the stored code systems.

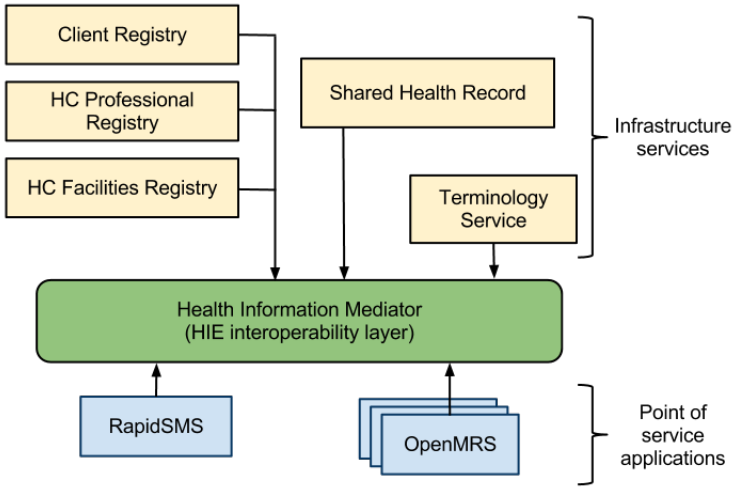

Fig. 1. The architecture of the Rwandan Health Information Exchange

3 Interoperability: challenges and requirements

The HIE interoperability layer, shown in figure 1, is the cornerstone of the Rwandan HIE architecture and its design has significant impact on the effectiveness, scalability, sustainability and adaptability of the system.

Its design was informed by the following requirements and challenges:

Facilitate interoperability between disparate and heterogeneous systems, both existing and future.

In the context of the Rwandan NHIS, the HIE initially allows the OpenMRS and RapidSMS systems to inter-operate with the infrastructure services (client registry, provider registry, facility registry and the shared health record) in order to share information. Each system embodies a different technology and architecture and the interoperability layer enables these systems to interact effectively.

The Interoperability Layer must provide mechanisms to allow existing disparate and heterogeneous systems to be incorporated into a HIE with minimal changes to the systems and still allow for local autonomy. The systems need to be able to grow and develop independently of the overall HIE and the other systems participating in the HIE. The architecture must be technology agnostic, with minimal restrictions on the technologies used within participating systems. Challenges include syntactic, semantic and process or pragmatic heterogeneity [17,11].

Adapt and scale within a changing environment

The focus of the current project is to enable the sharing of maternal health information between point of service applications in a single district. However, this architecture will need to adapt to new requirements and grow as the project progresses. It has to be designed to expand such that the services may be readily expanded to other districts in Rwanda, to incorporate additional domains of health care (for example, HIV/TB programmes) and allow other systems to be incorporated as part of the growth of the HIE.

The architecture must support incremental development and evolution of the HIE and also must be able to grow as the country's needs expand over time. This is especially true in low-resource environments where many organizations implement disparate information systems for a variety of purposes[3]. An essential feature of a HIE is its ability to cope with change. The architecture must be flexible enough to deal with changing and evolving NHIS requirements.

The system must also be able to scale, in terms of transaction volume, geographical locations and increased functionality.

Local changes should not propagate through the system

In Rwanda, development teams in different organizations design and maintain participating systems such as OpenMRS, RapidSMS and the infrastructure services. There are currently 14 partners working on the Rwandan HIE with 7

different development teams working on the various systems. Participating systems must be able to develop independently without affecting other systems. Participating systems will need to balance local requirements and NHIS requirements, but from a practical perspective development teams will often prioritise local requirements. Changes to participating systems should have minimal effect on other systems and systems must also be protected as much as possible from changes to infrastructure services. All systems must still maintain a large degree of local autonomy, especially since these systems are implemented and maintained by a variety of disparate organizations.

Provide a low barrier to entry to connect new and legacy systems

Implementing partners have development teams distributed around the world with varying degrees of expertise and technical skills. Inter-operating with the infrastructure services must be simple and require minimal effort both for current as well as new technical teams. A number of existing health information systems including the OpenMRS implementations and the RapidSMS implementation existed before the HIE was conceived.

The HIE should reduce the burden of connecting new and legacy systems participating in the HIE. The approach toward integration of legacy systems should be to 'embrace and extend' and not to 'rip and replace'. The architecture must provide a minimal barrier to entry to incorporate a system into the HIE and reduce the overhead required to modify a particular system to participate in the HIE. This feature will maximize the existing investment in legacy applications and help prevent useful and functioning legacy applications from being abandoned unnecessarily.

4 The architecture of the Health Information Mediator

This section describes the design of the Health Information Mediator shown in figure 1. The descriptions provided are based on the ISO 42010 architecture descriptions [13,8].

ISO/IEC FDIS 42010 provides a formal language and a metamodel for creating, analysing and sustaining architecture descriptions. An architecture can be described by a number of architectural viewpoints with each viewpoint framing a number of concerns (including requirements) of different groups of stakeholders with an interest in the system. Together, these views make up the architecture description. Based on the requirements identified in section 3, three major viewpoints and their associated concerns are described below.

4.1 Logical Viewpoint

This viewpoint describes the overall functionality of the system. The model kinds include custom diagrams showing how transactions flow through the architecture.

This view frames the following concerns:

- The architecture must facilitate interoperability between heterogeneous systems
- The architecture must provide a low barrier to entry to connect both new and legacy systems
- Changes should be kept local and not propagate through the system

Based on the above requirements, we have designed a middleware system, the Health Information Mediator (HIM) to enable interoperability between participating systems and infrastructure services. The HIM is based on the Enterprise Service Bus architectural model.

An Enterprise Service Bus (ESB)[5,20] is a middleware system that facilitates interoperability by providing a central bus that manages all communications between systems. Since the components within an ESB are loosely coupled and can run completely independently of each other, each component can still function independently when other components fail.

ESB is a well established architectural model for meeting the requirements associated with interoperability between distributed and disparate systems that has previously been applied to the problem of interoperability between disparate health information systems[19,12].

All participating systems in the HIE are represented as services. Systems that provide services to other systems are termed service providers, while systems that make requests of other systems are termed service requesters. All service requests are made via the HIM. The HIM thus provides mediation and orchestration functions within the system.

Our approach contains three major components described by the following 3-tuple:

$$\text{HIM} = \{\text{I, P, M}\}$$

where HIM is the Health Information Mediator, I is the Interface component, P is the Persistence component and M is the Mediation component.

Figure 2 shows the order in which transactions flow through each of the components.

Each of these components are described below:

I - Interface Component All interactions are carried out via the HIM. The interface component exposes an API that allows systems or applications to make service requests through the HIE. It is responsible for defining and handling all incoming service requests. Service requests are received using a standard protocol (e.g. HTTP) and translated into a common internal format that is accessible by the other components in the layer (e.g. Java Object). The request is then passed to the persistence component for further processing.

This component not only provides a single and consistent entry point for all service requests, but also enforces security and access policies for the HIE.

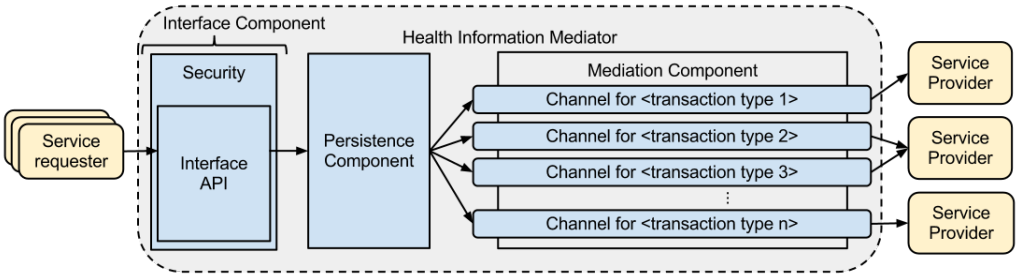

Fig. 2. Overview of components in the HIM architecture

A single point of access simplifies interactions with the HIE as the systems can make service requests without needing to know the location or security requirements of the service providers.

The API currently uses web services which affords the HIM greater flexibility when connecting systems using varying platforms and technologies. The functions provided by the API are defined according to the requirements of the HIE implementation. In the Rwandan use case this includes functions to save and query a patient's clinical record within the shared health record and to query and update records in the client, provider and facility registries.

This component also provides a central place for defining and applying advanced security policies. In this component, access to the API and access to specific functions of the API should be strictly controlled. The component also allows data-level security policies to be applied, if needed. In this paper, we have not addressed the complexities of defining how these security policies could be applied.

P - Persistence Component This component receives authorised service requests from the interface component and starts and monitors a transaction required to fulfill the request to completion.

It stores a copy of each transaction received by the HIM and maintains a persistent data store for the request data, the response data and metadata for each transaction. This data is stored for logging and audit purposes and can also be used to identify and handle exception conditions. This allows the administrators of the system to identify and solve recurring problems or failures. In this paper, we acknowledge that audit trails and exception handling are important issues to consider within a HIE however we do not explore these issues further, at this stage.

The metadata stored around transactions allow administrators of the system to monitor transactions and gauge the health of the system. This is useful for discovering bottlenecks and performance problems.

M - Mediation Component The mediation component executes transactions. Its main functions are orchestration and message translation.

The mediation component is made up of a number of transaction channels. A channel is provided for each transaction type, e.g. a transaction type to save a patient's encounter. It contains the necessary logic to normalise, orchestrate and de-normalise that transaction. Each function exposed by the API in the interface component maps to a transaction type and therefore to a transaction channel.

Below we describe the process that occurs within a single transaction channel contained within the mediation component.

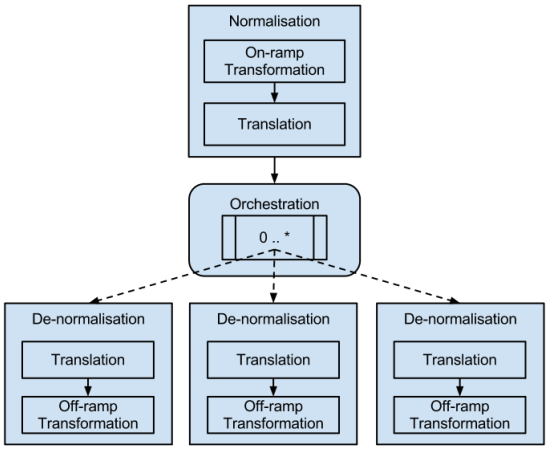

Fig. 3. The workflow of a transaction channel within the transaction mediation component

Figure 3 shows the inner workings of the transaction mediation component described earlier. Each transaction type has its own transaction channel. The diagram represents the workflow within a single transaction channel.

A transaction channel always begins with a normalisation sub-component. This sub-component transforms the request message contained within a transaction to a normalised state. After this process the transaction data must be in a consistent and predictable format to allow components following this to process it in a predictable fashion, no matter what format it arrived in. This process consists of 2 operations. Firstly, an on-ramp transformation is applied. This ensures syntactic interoperability for the transaction. For example, if the transaction arrives from a legacy application that only supported exporting data in a custom XML format, this process would ensure that the XML is transformed into a form that the rest of the exchange can understand, e.g. an HL7v2[4] message.

[4] HL7 is a standard messaging format for data within the health domain.

Secondly, a translation operation is invoked. This operation is responsible for ensuring the codes and code systems used within the transaction are translated to a standard set of vocabulary or clinical terms that have a common interpretation by other components of the HIM. This involves a call to the terminology service to translate and verify that the codes used within the transaction are in or are translated to an internal standard vocabulary. The terminology server is responsible for maintaining a standard vocabulary and mappings to other vocabularies used by participating systems. In this way semantic interoperability between service requesters and providers is achieved.

Following this, the transaction is sent to the orchestration sub-component. This sub-component is responsible for performing implementation-specific orchestration for the current transaction. The process of orchestration is described in Peltz et al[18]. The aim of the orchestration component is to execute the received transaction and perform any consequent action(s) required for this transaction. This could include 0 or more calls to external services. This component also compiles the response for the executed transaction and returns this to the persistence component which forwards the response to the service requester via the interface component.

A de-normalisation sub-component is provided for each external service. This sub-component is responsible for transforming (or constructing) a service request into a format that is understandable to the service provider. This operates in a similar way to the normalisation component except the operations occur in reverse order. This approach serves to decouple service providers from the orchestration component, which allows for service providers to be easily modified or replaced with minimal impact on the mediation component.

4.2 Scalability view

This view serves to show how the architecture can scale.

This viewpoint frames the following concern:

- The architecture must scale in terms of the number and volume of transactions

Figure 4 show the scalability of the architecture. In the architecture there are 3 major components; the interface API, the persistence component and the mediation component. Each of these components are loosely coupled to allow them to be deployed accross different servers. This is shown in 'Configuration 2' in figure 4. The 3 components are responsible for separate units of work. This loose coupling allows the components to be spread over different hardware as long as they can communicate over a network. The ESB architectural model used for this architecture ensures that the components are loosely coupled and can be deployed distributedly.

It is also feasible to further separate the persistence component and the transaction mediation component through clustering. The persistence component performs the static function of persisting any transaction that passes through it. As

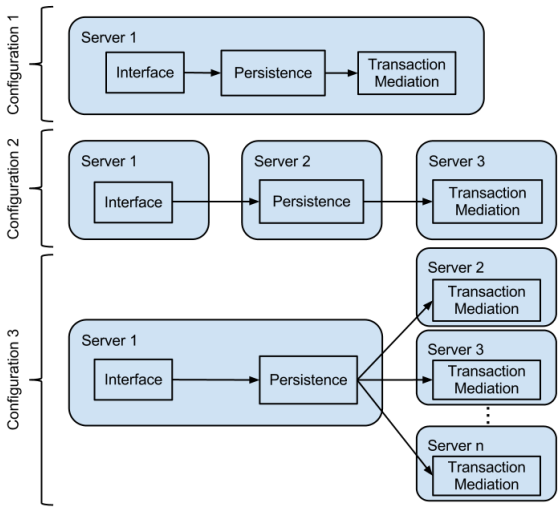

Fig. 4. Scalability configurations of the HIM architecture

this function is not dynamic it could easily be replicated over multiple servers with the provision that the data store is kept in sync. This component could also be invoked in an asynchronous fashion as the mediation component subsequent to it does not require this process to complete in order to continue.

The transaction mediation component can be scaled horizontally. The transaction mediation component holds a set of channels, one for each transaction type that is supported by the implementation. Each of these channels encapsulates information about how each transaction should be transformed and orchestrated. Each transaction channel runs independently which allows for deployment of the channels across different servers. This is shown in configuration 3 in figure 4.

4.3 Adaptability view

This view shows the architecture's ability to grow with a country's NHIS and how new services can be easily added or changed within the architecture.

This view frames the following concern:

- The architecture must be adaptable in a changing environment

Adaptability is an important consideration for this architecture. Figure 5 shows how additional services could be added to the architecture. As can be seen, to add additional services the interface component's API needs to be extended to add new API endpoints for each new function that needs to be supported. The persistence component is generic enough that it does not require any change to process new types of service requests. The transaction mediation component is where most of the changes are required. This component is designed to encapsulate transaction mediation logic for each transaction type. A new transaction

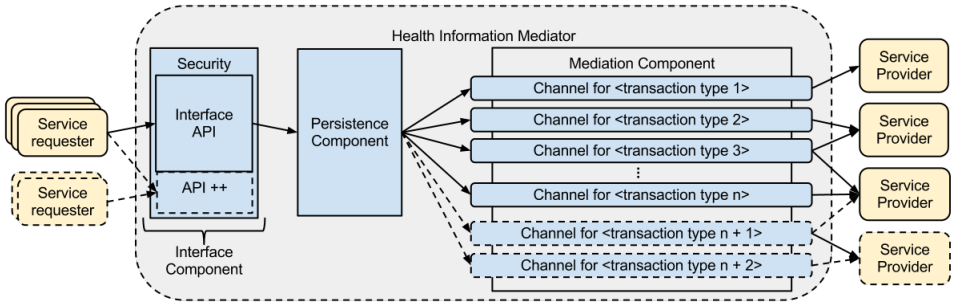

Fig. 5. Adapability of the HIM architecture

channel can easily be added to support a new type of service request. The channel will encapsulate all the logic for normalising the transaction, executing the necessary orchestration steps and to de-normalise the transaction when an external service orchestration call is made. This encapsulation simplifies the addition of new service request types as functionality increases and the HIE expands.

5 Analysis

In this section the HIM architecture is analysed against the requirements set out in section 3. This HIM architecture is currently being used to drive the development of the Rwandan HIE. This implementation and deployment of the first phase of the HIE is currently underway and the architecture is already showing benefit during this process. The discussion below is based on our experiences of implementing this architecture.

One of the core requirements of the HIE is to allow disparate systems to connect to each other easily. These could be legacy or new systems built by various international organizations. The architecture accomplishes this by enforcing a single interface API to connect to the HIE. This API hides the complexity of the HIE as well as the underlying system(s) that are invoked to fulfill service requests. This architecture also protects the application requesting services from changes that will inevitably occur to service providers, their API's or migration to a different location. This enables and supports local autonomy of the participating systems.

As new services are being developed and deployed for the Rwandan NHIS the Rwandan HIM implementation is being used to quickly and easily switch between mock service providers and the actual service provider implementations. This demonstrates one of the most critical features of the architecture; the ability to adapt. We are able to easily swap-out systems providing services as the environment changes. This will inevitably be a very important feature when the system goes live within Rwanda due to the ever changing nature of HISs in low resource environments.

The proposed architecture proves to be highly adaptive. This can be seen in the adaptability view of the architecture. Adding additional transaction types to the HIM is simplified by minimising the points at which changes are needed and by encapsulating transaction type specific logic into channels dedicated to specific transactions. This allows the architecture to adapt effectively as the HIE environment and functionality grows.

One of the major benefits of this architecture is that is does not prescribe the use of a particular data exchange format. There are many messaging standards available in the health domain for syntactic interoperability, each with different structures for representing data. Standards exist for various types of messaging needs. For example, sending clinical information (HL7 v2, HL7 v3, OpenEHR Archetypes[6,10,7]) or aggregate health information for reporting (SDMX-HD[3]). A defacto standard for health care messaging has yet to emerge[7]. New standards will emerge over time and current standards will fall away. Given these facts we can see that no single standard will ever be sufficient for all messaging needs. Therefore, the architecture must support current and future standards for syntactic interoperability. In the proposed architecture any data can be exchanged as long as we have normalisation and de-normalisation transforms defined to allow the data format to be transformed into and out of a form that the mediation component can understand and orchestrate. This affords the architecture greater flexibility in the types of data that can flow through it and allows the architecture to cater for multiple domains of health care even if the standard data exchange formats used within those domains are very different. This approach also future proofs the architecture against the inevitable change and evolution that will occur in the syntactic interoperability domain in health care.

A criticism of the architecture presented here is that it does not draw a clear line between parts of the system that are implementation specific and parts that can be part of a more general interoperability framework. Within the interface component and the mediation component there are parts that need to be defined depending on the API and business processes that are being implemented. These parts are implementation specific. The interface component defines an API that will be heavily driven by implementation needs and the mediation component defines orchestrations that are defined by the implementation as well as on-ramp steps and off-ramp steps that would depend on the data representations used within that implementation. It would be beneficial to identify the implementation specific aspects of this architecture so that a general interoperability framework can be extracted and implementation specific configuration can be plugged-in as needed. The current architecture does not account for this. This can be explored in future work.

The security architecture is also not expanded upon greatly in this architecture. It is identified that having a common entry point into the interoperability layer is beneficial in this regard as there is only a single endpoint to secure, however there are much greater considerations that need to be identified. Two main examples are: restricting transactions that specific applications can execute

within the interoperability layer and providing data level security on the clinical information that passes through the system.

Overall, the architecture fulfills the key requirements needed to implement a HIE interoperability architecture for a NHIS in Rwanda. This has been proven to work in a lab environment as the implementation for the Rwandan HIE is being developed. Many of these requirements are not specific to Rwanda and can be applied to other low-resource settings where a HIE is needed. Therefore, the authors believe this architecture is highly applicable for use in other countries.

6 Conclusion

In this paper we have identified the need for an interoperability architecture to solve the problem of interoperability between many disparate Health Information Systems. The Rwandan HIE use case was used to drive the identification of the requirements for this middleware layer, however, these requirements are largely applicable to other contexts. We introduce the HIM architecture that attempts to solve all the problems identified by the requirements. ISO 42010 is utilised to describe this architecture so that we can ensure all of the concerns are satisfied by utilising 3 different views of the architecture.

The HIM architecture description presents a proposed solution for interoperability architectures for use in low-resource countries like Rwanda and attempts to formalise the description of such an architecture so that it can be reused in other settings. The architecture is analysed using experience in implementing the architecture for use in the Rwandan HIE. It is identified that the architecture solves the problems identified by the requirements, however, it fails to provide a clear separation between the implementation specific configuration and the framework for a more general architecture. Overall, the architecture provides a solution to the major problems faced when attempting to facilitate interoperability between many disparate health information systems and it has proven in practice to be an appropriate, adaptable and scalable solution.

Acknowledgements

The authors wish to acknowledge the support of the Rwanda Ministry of Health and, in particular, the National eHealth coordinator, Dr Richard Gakuba, and advisers, Elizabeth Peloso and Randy Wilson. Other inputs were received from the Rwanda Health Enterprise Architecture (RHEA) project team, including Mead Walker, Beatriz de Faria Leao, Paul Biondich, Wayne Naidoo and Eduardo Jezierski. Additional support was obtained from eZ-Vida in Brazil and, in particular, Dr Lincoln Moura. The RHEA project is funded by grants from the IDRC (Open Architectures, Standards and Information Systems (OASIS II) – Developing Capacity, Sharing Knowledge and Good Principles Across eHealth in Africa. Grant Number: 105708), the Rockefeller Foundation (Open eHealth Enterprise Architecture Framework and Strategy Development for the Global

South; Grant Number: 2009 THS 328) and the Health Informatics Public Private Partnership Project funded by the United States President's Emergency Plan for AIDS Relief (PEPFAR). The HEAL project is funded by grants from the Rockefeller Foundation (Establishing a Health Enterprise Architecture Lab, a research laboratory focused on the application of enterprise architecture and health informatics to low-resource settings, Grant Number: 2010 THS 347) and the IDRC (Health Enterprise Architecture Laboratory (HEAL), Grant Number: 106452-001).

References

1. Carla AbouZahr and Ties Boerma. Health information systems: the foundations of public health. *Bulletin of the World Health Organization*, 83(8):578–583, August 2005.
2. Christian Allen, Darius Jazayeri, Justin Miranda, Paul G. Biondich, Burke W. Mamlin, Ben A. Wolfe, Chris Seebregts, Neal Lesh, William M. Tierney, and Hamish S. Fraser. Experience in implementing the OpenMRS medical record system to support HIV treatment in Rwanda. *Studies in health technology and informatics*, 129(Pt 1):382–386, 2007.
3. Jørn Braa, Andrew S. Kanter, Neal Lesh, Ryan Crichton, Bob Jolliffe, Johan Sæbø, Edem Kossi, and Christopher J. Seebregts. Comprehensive yet scalable health information systems for low resource settings: a collaborative effort in Sierra Leone. *AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium*, 2010:372–376, 2010.
4. Jorn Braa and Humberto Muquinge. Building collaborative networks in Africa on health information systems and open source software development - Experience from the HISP/BEANISH network. 2007.
5. David Chappell. *Enterprise Service Bus: Theory in Practice*. O'Reilly Media, July 2004.
6. Rong Chen. *Towards interoperable and knowledge-based electronic health records using archetype methodology*. PhD thesis, 2009.
7. Marco Eichelberg, Thomas Aden, Jörg Riesmeier, Asuman Dogac, and Gokce B. Laleci. A survey and analysis of Electronic Healthcare Record standards. *ACM Comput. Surv.*, 37(4):277–315, December 2005.
8. David Emery and Rich Hilliard. Updating IEEE 1471: Architecture Frameworks and Other Topics. In *Seventh Working IEEE/IFIP Conference on Software Architecture (WICSA 2008)*, pages 303–306, Washington, DC, USA, February 2008. IEEE.
9. J. M. Ferranti, R. C. Musser, K. Kawamoto, and W. E. Hammond. The Clinical Document Architecture and the Continuity of Care Record: A Critical Analysis. *Journal of the American Medical Informatics Association*, 13(3):245–252, May 2006.
10. Sebastian Garde, Rong Chen, Heather Leslie, Thomas Beale, Ian McNicoll, and Sam Heard. *Archetype-Based Knowledge Management for Semantic Interoperability of Electronic Health Records*, pages 1007–1011. IOS Press, 2009.
11. Patricia Gibbons, Noam Arzt, Susie Burke-Beebe, Chris Chute, Gary Dickinson, Tim Flewelling, Thomas Jepsen, Don Kamens, Joanne Larson, John Ritter, Michael Rozen, Sherry Selover, and Jean Stanford. Coming to Terms: Scoping Interoperability for Health Care. Technical report, February 2007.

12. Ibm. IBM Enterprise Service Bus for Healthcare. Technical report, 2010.
13. Iso. ISO/IEC FDIS 42010 IEEE P42010/D9. Systems and software engineering - Architecture description. Technical report, March 2011.
14. Burke W. Mamlin, Paul G. Biondich, Ben A. Wolfe, Hamish Fraser, Darius Jazayeri, Christian Allen, Justin Miranda, and William M. Tierney. Cooking up an open source EMR for developing countries: OpenMRS - a recipe for successful collaboration. *AMIA Symposium*, pages 529–533, 2006.
15. Moh. Health Sector Strategic Plan. July 2009 - June 2012, July 2009.
16. Deshendran Moodley, Anban Pillay, and Christopher J. Seebregts. Position Paper: Researching and Developing Open Architectures for National Health Information Systems in Developing African Countries. In *International Symposium on Foundations of Health Information Engineering and Systems*, volume 1234, August 2011.
17. A. M. Ouksel and A. Sheth. Semantic interoperability in global information systems. *SIGMOD Rec.*, 28(1):5–12, March 1999.
18. C. Peltz. Web services orchestration and choreography. *Computer*, 36(10):46–52, October 2003.
19. Amanda Ryan and Peter Eklund. The Health Service Bus: an architecture and case study in achieving interoperability in healthcare. *Studies in health technology and informatics*, 160(Pt 2):922–926, 2010.
20. M. T. Schmidt, B. Hutchison, P. Lambros, and R. Phippen. The Enterprise Service Bus: Making service-oriented architecture real. *IBM Systems Journal*, 44(4):781–797, 2005.
21. Christopher J. Seebregts, Burke W. Mamlin, Paul G. Biondich, Hamish S. F. Fraser, Benjamin A. Wolfe, Darius Jazayeri, Christian Allen, Justin Miranda, Elaine Baker, Nicholas Musinguzi, Daniel Kayiwa, Carl Fourie, Neal Lesh, Andrew Kanter, Constantin T. Yiannoutsos, and Christopher Bailey. The OpenMRS Implementers Network. *International Journal of Medical Informatics*, 78(11):711–720, November 2009.