An Architecture and Reference Implementation of an Open Health Information Mediator: Enabling Interoperability in the Rwandan Health Information Exchange

Ryan Crichton[1,2], Deshendran Moodley[1], Anban Pillay[1], Richard Gakuba[3], and Christopher J. Seebregts[1,2,4]

[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] eHealth Coordination Unit, Ministry of Health, Rwanda
[4] 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 more 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 the ISO/IEC/IEEE 42010 standard architecture descriptions. The architecture takes an Enterprise Service Bus approach. A partial implementation and preliminary analysis of the architecture is given. Since these challenges are experienced by other developing African countries, the next steps involves creating a generic architecture that can be reused for health information exchange in other developing African countries.

Keywords: interoperability, national health information system architecture, enterprise service bus, health information exchange.

1 Introduction

The current landscape of health information systems, especially in the developing world, is mostly characterised by fragmented, piecemeal applications deployed by multiple organizations [1,4]. Applications are usually custom built to satisfy very specific needs, 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) is challenging. One potential solution to enable interoperability is to implement a mediator component that facilitates information exchange and orchestration between participating health information systems and applications in the NHIS, including point of service applications and shared registries and services.

In our previous work [21] 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 Health Information Mediator (HIM) 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 that informed the design of the HIM. The architecture of the HIM is presented in section 4 and section 5 gives an analysis of this architecture. In section 6 the implementation of the architecture is briefly described and we draw our conclusions in section 7.

2 Background: A National Health Information System for Rwanda

The Rwanda Ministry of Health (MoH) has already made significant progress in developing a country-level NHIS, that includes, among others, community health systems, health management information systems and the national roll-out of an electronic medical record application [20]. The Rwanda Health Enterprise Architecture (RHEA) project, led by the Rwanda MoH and supported by a consortium of partners and donors has developed an Health Information Exchange to facilitate interoperability between individual health information systems and applications. We follow Dixon et al [8] and define a health information exchange (HIE) broadly as "the sharing of clinical and administrative healthcare data among healthcare institutions, providers, and data repositories."

Implementation of the Rwandan HIE will be achieved in several phases. The first phase will implement foundational components, including client, professionals and facilities registries, a terminology service and a shared health record, to improve interoperability between two point of care information systems supporting maternal health in the Rwamagana district, including 15 health centers. The two point of care systems being implemented and maintained by the Rwandan MoH are implementations of OpenMRS [18,2,26], an Electronic Medial Record (EMR) system and RapidSMS, an SMS based data collection tool that is currently being used by community health workers. 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. In subsequent phases, the HIE will need to accommodate other applications and use cases and also scale, nationally.

The HIE's main function is to enable the point of care systems currently implemented in Rwanda to connect and inter-operate more easily. Using the HIE, the MoH 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 [11] to enable access to a patient's clinical information from different health facilities thus improving the tracking of patients and reducing the number of patients lost-to-follow-up.

The first phase involves deploying a set of foundational infrastructure services that provide services to point of care applications, initially, OpenMRS and RapidSMS. The HIE will allow the systems 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.

3   Interoperability: Challenges and Requirements

The 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 overall system. In the sections that follow we enumerate the challenges and requirements, suggest and explain a possible design of an architecture for this interoperability layer and give a preliminary analysis of it effectiveness when applied to the Rwandan HIE.

The design was informed by the following requirements and challenges that were identified from studying the situation in Rwandan and with knowledge of how health informations systems are deployed in low resource settings:
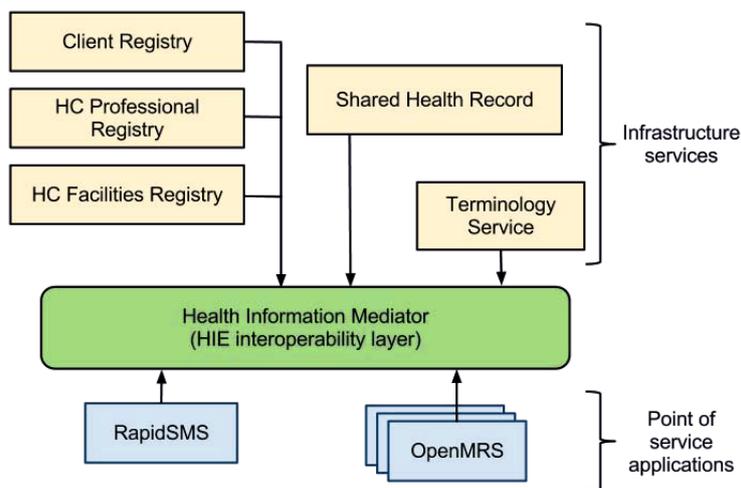
Fig. 1. The architecture of the Rwandan Health Information Exchange

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 the 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 [22,14].

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 also 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, the 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. Currently, there are 14 partners working on the Rwandan HIE with 7 different development teams working on the various participating systems that 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   Architecture of the Health Information Mediator

In order to overcome the challenges and fulfill the requirements for interoperability identified in section 3, we introduce a new component, the Health Information

Mediator (HIM) (figure 2). The design and implementation of the HIM draws heavily from two technologies that were evaluated in the initial stages of the Rwandan project. The first, Mirth Connect (Mirth Corporation), is an open integration engine for health information systems. However, the Rwandan project required complex orchestrations that Mirth Connect could not easily support and it was simpler to directly use the underlying Mule ESB [16] platform on which Mirth Connect is built to perform orchestration. We also reviewed and setup the reference implementation of the Canada Health Infoway (CHI) EHR Blueprint [7,19]. In the CHI HIE implementation the interoperability and orchestration functions are provided by Biztalk (Microsoft Corporation), supplemented by Everest, an HL7[1] version 3 adapter and open C# library. However, Biztalk is expensive to license and maintain and HL7 version 3 is a difficult messaging specification to implement in low resource settings due to its complexity and verbose nature.

In this section, we describe the architecture of the HIM using ISO 42010 architecture descriptions [17,10]. 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 views with each view 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 views of the HIM architecture and their associated concerns are described below.

4.1   Logical View

This view describes the overall functionality of the system. The model kinds include custom diagrams showing how transactions flow through the architecture. It 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 these requirements, we have designed the HIM as a middleware system to enable interoperability between participating systems and infrastructure services. The HIM is based on the Enterprise Service Bus (ESB) architectural model.

An ESB [5,25] is a middleware system that facilitates interoperability by providing a central bus that manages all communications between participating 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.

---

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

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 [24,15].

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:

HIM = {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.

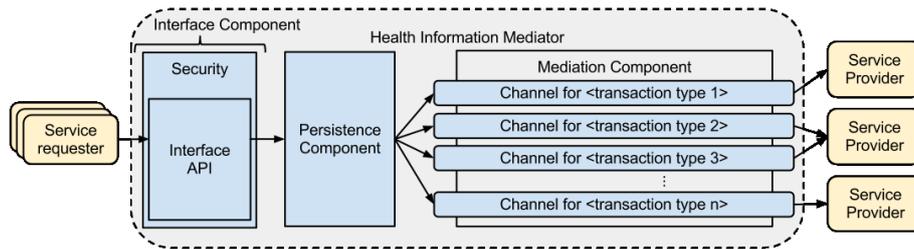Figure 2 shows the order in which transactions flow through each of the components.

Fig. 2. Overview of components in the HIM architecture

Each of these components are described below:

I - Interface Component. All interactions are carried out via the HIM. The interface component exposes an application programming interface (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 Objects). 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.

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 in order to focus on the architectural significance of security and not the implementation details.

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.

Transaction metadata 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.

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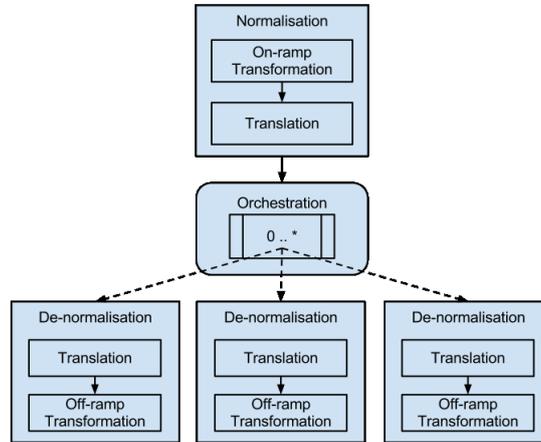
Fig. 3. The workflow of a transaction channel within the transaction mediation component

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 HL7 version 2 message. 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 [23]. 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 call. 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 describes how the architecture can scale and frames the following concern:

– The architecture must scale in terms of the number and volume of transactions
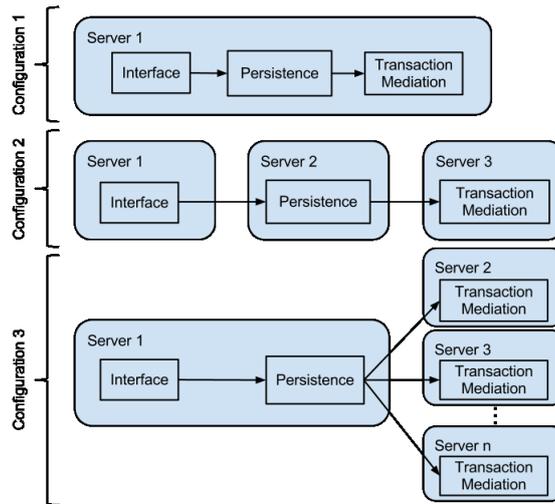
Fig. 4. Scalability configurations of the HIM architecture

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 across 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 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.

These configurations show two important aspects of the architecture. Firstly performance in terms of volumes of transactions, i.e. splitting the load between different servers increases the capability of the system to handle and process a higher volume of transactions timeously. Additional servers can be introduced as transaction volumes grow . Secondly, robustness. Since each of the three components are responsible for separate units of work and individual components can be replicated over different physical machines to provide redundancy. The number of instances of each component can be varied depending on the transaction types and processing requirements.

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:

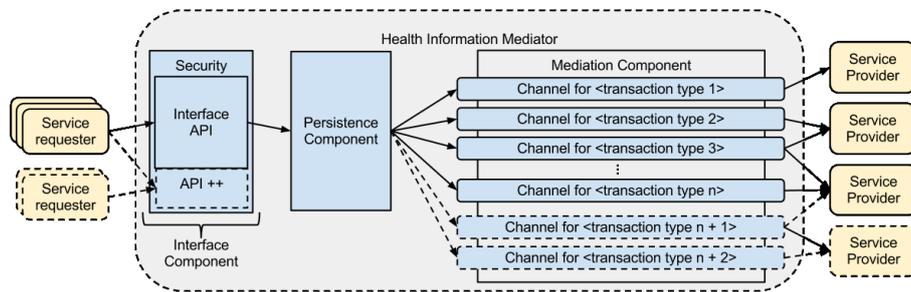– The architecture must be adaptable in a changing environment

Fig. 5. Adaptability of the HIM architecture

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 channel can easily be added along side the others 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. The implementation and deployment of the first phase of the HIE in Rwanda 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 or local 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 applications requesting services from changes that will inevitably occur to service providers, their API's or as a result of 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 was 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.

The proposed architecture has been shown 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 [6,13,9]) or aggregate health information for reporting (SDMX-HD [3]). A defacto standard for health care messaging has yet to emerge [9]. 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 HIM 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.

The HIM architecture was conceived by studying the challenges and requirements of NHISs in a low resource setting. These challenges led us to an architecture that relies on a central component (the HIM) that co-ordinates all the interaction within the HIE. This design choice has its benefits as well as its

challenges. Having a central component gives the benefit of easing the burden of implementing interoperability between HISs as the infrastructure only need to be deployed once and the HIM can simplify the burden of connecting to a HIE. It also gives a country central control over the transactions supported within the HIE. Having a central component that is responsible for orchestration of all the transactions also allows the client systems to be so-called 'dumb clients' and only interact with the system in a simple manner. This enables quicker and easier integration that will help resource constrained projects to connect their systems to the HIE. The design also keeps much of the communication between systems in the datacentre where communication is quick and responsive. Client systems in low resource setting are often on slow networks that are often unresponsive or out of order. Minimal communication with a single central component allows clients to communicate effectively with the little bandwidth that they have. On the other hand, having a central component also has certain negative aspects. A central component that the entire HIE relies on introduces a single point of failure. Also, if any changes need to be make to the transactions that the HIE supports the central component need to be changed and all other systems have to wait until these changes are implemented before they can utilise the new transactions. The HIM would likely be controlled by a government entity and the client systems are often controlled by a wide variety of organizations that can move much more quickly than a government entity. Thus, problems could be encountered if the government entity is not responsive enough to change requests.

Alternative design approaches could do away with a central component and expect the client to know how to communicate among themselves ('smart clients' or service choreography). In our case the central approach seemed most appropriate due to the fact that we are working in a low resource setting. The benefits for a low resource setting out-weighed the negatives listed above, however, the authors note that this will not always be the case in other settings.

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 as well as in production as the Rwandan HIE begins to be rolled out. 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   Implementation and Future Work

The HIM architecture, described above was implemented and successfully deployed with the other HIE components in Rwanda during September 2012. The current system connects two health facilities in the Rwamagana district to the HIE deployed in the national datacentre in Kigali[2].

---

[2] See the implementation blog at
  http://rwandahie.blogspot.com/2012/09/click.html

The Infrastructure services that form the rest of the Rwandan HIE were implemented by different parties utilising a wide variety of open source projects, which are listed below:

– Shared Health Record: OpenMRS (OpenMRS Foundation, Regenstrief Institute and Partners in Health)
– Client Registry: OpenEMPI (SYSNET International)
– Provider Registry: a custom open source webapp built on OpenLDAP (Intrahealth)
– Facility Registry: ResourceMapper (InSTEDD)
– Terminology service: Apelon DTS (Apelon Inc.) and a webapp frontend (Jembi Health Systems NPC).

The Rwandan HIM was developed on the open source Mule ESB [16] platform, and incorporates a RESTful web services approach [12]. The implementation and field experience sets the foundation towards creating an Open Health Information Mediator (OpenHIM). The architecture as well as the implemented components of the Rwandan HIM are general enough to allow their re-use in other settings. The aim is to release the Rwandan HIM as open source and for it to serve as the reference implementation for the OpenHIM. The next step is to establish an open community around OpenHIM to provide participation from other stakeholders and to promote its adoption and to facilitate the creation of Health Information Exchanges in other low resource settings.

7 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 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.

References

1. AbouZahr, C., Boerma, T.: Health information systems: the foundations of public health. Bulletin of the World Health Organization 83(8), 578–583 (2005)
2. Allen, C., Jazayeri, D., Miranda, J., Biondich, P.G., Mamlin, B.W., Wolfe, B.A., Seebregts, C., Lesh, N., Tierney, W.M., Fraser, H.S.: 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. Braa, J., Kanter, A.S., Lesh, N., Crichton, R., Jolliffe, B., Sæbø, J., Kossi, E., Seebregts, C.J.: Comprehensive yet scalable health information systems for low resource settings: a collaborative effort in Sierra Leone. In: AMIA Annual Symposium Proceedings, vol. 2010, pp. 372–376 (2010)
4. Braa, J., Muquinge, H.: Building collaborative networks in Africa on health information systems and open source software development - Experience from the HISP/BEANISH network. IST Africa (2007)

5. Chappell, D.: Enterprise Service Bus: Theory in Practice. O'Reilly Media (July 2004)

6. Chen, R.: Towards interoperable and knowledge-based electronic health records using archetype methodology. PhD thesis, Department of Biomedical Engineering, Linköpings universitet (2009)

7. CHI: EHRS Blueprint. An Interoperable EHR Framework. Executive Overview

8. Dixon, B.E., Zafar, A., Marc Overhage, J.: A framework for evaluating the costs, effort, and value of nationwide health information exchange. JAMIA 17(3), 295–301 (2010)

9. Eichelberg, M., Aden, T., Riesmeier, J., Dogac, A., Laleci, G.B.: A survey and analysis of Electronic Healthcare Record standards. ACM Comput. Surv. 37(4), 277–315 (2005)

10. Emery, D., Hilliard, R.: Updating IEEE 1471: Architecture Frameworks and Other Topics. In: Seventh Working IEEE/IFIP Conference on Software Architecture (WICSA 2008), pp. 303–306. IEEE, Washington, DC (2008)

11. Ferranti, J.M., Musser, R.C., Kawamoto, K., Hammond, W.E.: The Clinical Document Architecture and the Continuity of Care Record: A Critical Analysis. Journal of the American Medical Informatics Association 13(3), 245–252 (2006)

12. Fielding, R.T.: Architectural styles and the design of network-based software architectures. PhD thesis, University of California, Irvine, CA, USA (2000)

13. Garde, S., Chen, R., Leslie, H., Beale, T., McNicoll, I., Heard, S.: Archetype-Based Knowledge Management for Semantic Interoperability of Electronic Health Records, pp. 1007–1011. IOS Press (2009)

14. Gibbons, P., Arzt, N., Burke-Beebe, S., Chute, C., Dickinson, G., Flewelling, T., Jepsen, T., Kamens, D., Larson, J., Ritter, J., Rozen, M., Selover, S., Stanford, J.: Coming to Terms: Scoping Interoperability for Health Care. Technical report, Health Level Seven EHR Interoperability Work Group (February 2007)

15. IBM: IBM Enterprise Service Bus for Healthcare. Technical report (2010)

16. MuleSoft Inc.: Mule ESB (2012),
http://www.mulesoft.org/what-mule-esb

17. ISO: ISO/IEC FDIS 42010 IEEE P42010/D9. Systems and software engineering - Architecture description. Technical report, ISO (March 2011)

18. Mamlin, B.W., Biondich, P.G., Wolfe, B.A., Fraser, H., Jazayeri, D., Allen, C., Miranda, J., Tierney, W.M.: Cooking up an open source EMR for developing countries: OpenMRS - a recipe for successful collaboration. In: AMIA Symposium, pp. 529–533 (2006)

19. Duane, B., Yendt, M.: Developing an Open Source Reference Implementation of the Canadian Electronic Health Records Solution. Open Source Business Resource, Health and Life Sciences (November 2008)

20. Moh: Health Sector Strategic Plan (July 2009-June 2012)

21. Moodley, D., Pillay, A.W., Seebregts, C.J.: Position Paper: Researching and Developing Open Architectures for National Health Information Systems in Developing African Countries. In: Liu, Z., Wassyng, A. (eds.) FHIES 2011. LNCS, vol. 7151, pp. 129–139. Springer, Heidelberg (2012)

22. Ouksel, A.M., Sheth, A.: Semantic interoperability in global information systems. SIGMOD Rec. 28(1), 5–12 (1999)

23. Peltz, C.: Web services orchestration and choreography. Computer 36(10), 46–52 (2003)

24. Ryan, A., Eklund, P.: 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)
25. Schmidt, M.T., Hutchison, B., Lambros, P., Phippen, R.: The Enterprise Service Bus: Making service-oriented architecture real. IBM Systems Journal 44(4), 781–797 (2005)
26. Seebregts, C.J., Mamlin, B.W., Biondich, P.G., Fraser, H.S.F., Wolfe, B.A., Jazayeri, D., Allen, C., Miranda, J., Baker, E., Musinguzi, N., Kayiwa, D., Fourie, C., Lesh, N., Kanter, A., Yiannoutsos, C.T., Bailey, C.: The OpenMRS Implementers Network. International Journal of Medical Informatics 78(11), 711–720 (2009)