Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 1 / APRIL 2010 / 56-70

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

Policy-based Data Integration for e-Health Monitoring Processes in a B2B Environment: Experiences from Canada

Benjamin Eze[1], Craig Kuziemsky[2], Liam Peyton[3], Grant Middleton[4] and Alain Moutthamm[5]

University of Ottawa, Canada, School of Information Technology and Engineering
[1] eze.ben@hotmail.com, [3] lpeyton@site.uottawa.ca, [4] middleton13@hotmail.com, [5] amouttham@site.uottawa.ca
[2] University of Ottawa, Canada, Telfer School of Management, kuziemsky@telfer.uOttawa.ca

Received 20 July 2009; received in revised form 2 November 2009; accepted 18 February 2010

Abstract

eHealth processes are data-focused, event-driven, and dynamic. They are systematically monitored for compliance with legislation, organizational guidelines and quality of care protocols. Community care, especially at home care, frequently requires the cooperation and integration of care processes across several providers and organizations. Service Oriented Architecture (SOA) through Web services and business process automation through Business Process Execution Language (BPEL) is emerging as a framework for business process integration over the Internet, but does not address all information management requirements of eHealth monitoring processes particularly with respect to policy compliance and event-based data integration. In this paper, we extend the traditional SOA framework to define a flexible policy-based approach for defining and monitoring streaming event data based on a general publish/subscribe model in a business-to-business (B2B) healthcare network. The work described here is design-oriented research where the purpose is to show the utility of the proposed framework. The approach is evaluated based on information management requirements drawn from a case study of palliative care and a prototype implementation.

Key words: eHealth, Business Process Integration, Policy Broker, B2B, Publish/Subscribe, SOA

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 1 / APRIL 2010 / 56-70

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

1 Introduction

e-Health processes are data-focused, event-driven, and dynamic. They are systematically monitored for compliance with legislation, organizational guidelines and quality of care protocols. Community care, especially at home care, frequently requires the cooperation and integration of care processes across several providers and organizations in a business-to-business (B2B) network. The flow of information between parties in B2B networks is crucial for effective and efficient collaboration. Applications, databases, processes and increasingly devices share resources, and need to exchange data on a continuous basis [19]. As a result, businesses are continuously seeking new, fast, inexpensive and secure means of sharing information on goods and services [37].

Service Oriented Architecture (SOA) [23] is emerging as the standard framework for systematic and extensible machine-to-machine interaction on the Internet. Using the Business Process Execution Language (BPEL) [32], one can systematically define on-line processes that orchestrate behavior and data processing across such web services. In certain circumstances, though, traditional SOA is limited by a point to point messaging framework that entails unnecessary procedural interaction and data polling that can limit flexible data integration [18]. This is especially true with respect to processes that run within a B2B network as opposed to a single enterprise [14]. Event-driven architecture [29] on the other hand is characterized by its ability to decouple service providers and consumers through messages [16]. That is, producers and consumers of data are contextually coupled by the data exchanges and not procedural calls. Publish/subscribe [17] is an event-driven model that allows many clients to subscribe to the same data source, and have event data sent to them as messages as they become available.

Both SOA and event-driven architecture are challenged to fully address the information management requirements of eHealth monitoring processes. In particular, both lack support for a common data model across a B2B network to support data integration. In an eHealth network that supports at home care, there may be many different types of data from many different organizational sources that must be integrated to provide a single consistent view of the network. As well, policy compliance is an issue with respect to privacy legislation, hospital guidelines, and procedures that dictate under what circumstances individuals can view data.

In this paper, we extend the traditional SOA framework to define a flexible policy-based approach for defining and monitoring streaming event data based on a general publish/subscribe model in a B2B healthcare network. The work described here is design-oriented research that consisted of problem identification, iterative framework design, framework evaluation, and communication and discussion of findings. Our contributions include:

1. Identifying the key information management requirements for eHealth monitoring processes

2. An analysis of the gaps that currently exist in leveraging traditional SOA in addressing those requirements

3. and a proposed framework that extends traditional SOA with a policy-based message broker, partner service components for publishing and subscribing to event data, and an event topic registry that defines a network wide streaming data integration model maintained and monitored by a surveillance portal.

This work is based on a review of existing processes and infrastructure for palliative care at the local health authority in Ottawa, Canada [30]. We have implemented a prototype of our framework and evaluated it based on a pain management scenario for palliative care at the local health authority in comparison with:

1. a proposed Entity Resource Planning (ERP) Web portal [26]

2. a proposed SOA framework for trusted support of BPEL in B2B networks [14].

2 Background

We summarize relevant work on e-health information systems related to palliative care and identify emerging technologies and related work for improving such systems.

2.1 e-Health Information Systems

Healthcare is an information intensive business that necessitates getting the right information to the right people at the right time. However the provision of integrated tasks such as coordinated care and distributed decision making are challenging for the fundamental reason that our existing healthcare system is not designed for it [10]. Healthcare has traditionally been delivered through a silo based model where data and processes are supported for an individual setting or provider. For our healthcare system to be sustainable in the forthcoming years it needs to be built as a set of integrated services that make data and processes accessible across different settings [10]. E-health technologies will play a crucial role in designing new approaches to healthcare delivery.

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

Increasing numbers of patients are being diagnosed with and living for extended periods with chronic diseases. In the United States over 125 million people have chronic illness or functional disability [1]. Healthcare delivery to manage chronic illness requires patients to receive care from multiple providers across multiple care settings. Such care delivery requires coordination of processes and sharing of information to provide safe, efficient care without duplication of efforts. The electronic health record (EHR) has received much attention for its anticipated role in supporting healthcare delivery but it acts as a channel for data and not a facilitator of clinical processes that act upon the data. It has been shown that EHR frameworks largely support individual physician practices rather than facilitating management of chronic illness or collaborative care activities [13].

An ERP Web portal [26] is a common type of information system architecture when sharing data between multiple providers across multiple care settings. All relevant data is shared through a web portal interface that allows providers to search and access the data they are interested in. Data is collected either by manual entry via the web interface, dynamic updates via web services, or batch updates in which data from external sources is extracted, transformed and loaded into the ERP system on a scheduled basis. This provides a common view of care processes to all concerned, but it requires providers to search and fetch data on their own. As well, there is typically much duplication of effort to maintain the ERP since most data must be copied and transformed from the primary source (typically one of the providers). Performance management of health care processes that use an ERP Web portal architecture is a challenge currently being researched [38] to ensure compliance with relevant legislation, hospital guidelines, and policies. PHIPA [35] is the personal health information legislation relevant to our case study based in Ottawa, Canada.

2.2 Service Oriented Architecture and e-Health

Service Oriented Architecture (SOA) groups functionalities around business processes and packages them as interoperable services. According to [41], SOA is “a set of components which can be invoked, and whose interface descriptions can be published and discovered”. It is characterized by business functions that are modeled as networked services, and a collection of self-describing application interfaces [15]. SOA organizes and describes Web services to support dynamic and automated discovery, and binding in heterogeneous computing environments [34]. It is a request/response type interaction pattern and its popularity stems from the fact that message exchanges can span beyond organizational domains and corporate firewalls. SOA through Web services supports platform independence, interoperability and communication between heterogeneous applications.

Service Oriented Architecture (SOA) frameworks have been proposed to address the sharing of data in business-to-business (B2B) health networks. Liberty Alliance is a consortium of vendors who have defined standards and specifications to create a Circle of Trust [8] in which providers can share data in a controlled fashion with user consent. Its use for health care applications has been examined [36], and extensions to support processes defined using Business Process Execution Language (BPEL) and integrated performance management have been proposed [14]. However, compared to an ERP Web portal, it is more difficult to establish a common data model across a B2B network to support data integration.

In assessing the usefulness of SOA for eHealth, it is important to distinguish the different types of clinical processes that might be supported. There are four groups of clinical processes (i.e. excluding administration and management of healthcare): diagnosis, monitoring, therapy/palliative care, and prognosis. Monitoring also supports the other processes by providing data collection, analysis, data sharing and alarm notifications.

Traditional SOA is starting to penetrate in new clinical information systems, and fits well in the therapy/palliative care and prognosis processes, because these typically involve one-to-one (point-to-point) and synchronous interactions. However, traditional SOA is not optimal for the diagnosis and monitoring processes which are typically driven by collected data, streamed and processed in real-time; these interactions are mostly one-to-many (point-to-multipoint), long-running and asynchronous. The system collecting the data doesn’t necessarily know where the data is processed next. Traditional SOA provides no inherent support for these types [25] of ad hoc and event-driven applications across B2B networks [28] as required in e-Health.

2.3 Related Technologies

There are a number of commonly used publish-subscribe frameworks available. MQ Telemetry Transport (MQTT) [33] is a TCP/IP based protocol that uses TCP ports as channels to support a publish/subscribe communication pattern. Java Message Service (JMS) is the publish-subscribe technology for J2EE applications [24], [40]. Both MQTT and JMS scale well with applications that run in the same network/domain but are not well adapted for application components that send and receive messages across domains and firewalls as is the case for B2B networks. Really Simple Syndication (RSS) [20] is based on HTTP with clients subscribing to feeds of interest from online sources using an aggregator. Its simplicity makes it popular with end-users and web sites, but it lacks features for managing data systematically across a B2B network. More recently, the PADRES system [9] supports a rule-based message broker for content-based routing to support distributed resource scheduling in the context of a business process management framework.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 1 / APRIL 2010 / 56-70

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

Rules-based policies govern system behavior by providing reactive functionalities [21]. Executed through policy engines, rule-based policies are very adaptable to a wide range of applications, such as adapting composite services with constantly changing business processes [42]. Event-Condition-Action (ECA) rules automatically perform a set of actions in response to events provided that certain stated conditions are met [3]. They are the most suitable pattern for building complex regulatory conditions [6] needed in setting up both internal and federated business processes. eXtensible Access Control Markup Language (XACML) [31] is a standardized common security-policy language that "allows the enterprise to manage the enforcement of all the elements of its security policy in all the components of its information systems". It has been applied to compliance including data sharing as determined by context [4].

2.4 Methodology

The work described in this paper is design-oriented research in the form of a "search process to discover an effective solution to a problem" [22]. We followed a five stage design research design consisting of: identification of problem relevance, framework design, framework evaluation, revaluation and improvement of framework, and communication and discussion of research [5]. Problem relevance was conducted through work with the local healthcare authority to identify the requirements for an information system to provide at home support for chronic disease management. We selected a representative case study to illustrate the types of issues we wished to address. The user requirements are detailed in the example scenario in section 3.1.1. Once the user requirements were gathered we conducted a literature review to identify the key information management requirements for eHealth monitoring. A gap-analysis was then done to identify the gaps in existing SOA for achieving those requirements. We then followed an iterative approach in developing a new framework to address the gap analysis and achieve the requirements described in our case study example. Once the framework was completed it was implemented as a prototype system and evaluated against two other common approaches for supporting eHealth monitoring processes in a B2B environment. In keeping with the design oriented approach, the evaluation is not an empirical evaluation but rather intended to show utility over the other approaches [22], [27].

3 Information Management Requirements for eHealth Monitoring Processes

We will now define the information management requirements for ehealth monitoring processes. In section 3.1.1 we illustrate with an example scenario of severe pain management for a patient receiving palliative home care.

3.1 Problem Relevance - eHealth Monitoring Processes and Palliative Care

Healthcare has unique characteristics that have no parallel in other sectors [2]. In particular healthcare service delivery often requires joint decision making and implementation of policies and procedures for assessment, referral and consultation that cut across multiple care providers and settings [2]. Furthermore, whereas other sectors are largely defined by transaction processing, healthcare is defined by continuous monitoring of events and the need for timely response to events through flexible allocation of resources and business processes in a controlled fashion. One particular domain of healthcare that requires integration of data and business processes for care delivery is palliative care, which is care provided to patients at the end of their life when curative therapies are no longer an option [11]. Palliative care business processes are complex for three reasons. One, palliative care service delivery incorporates multiple dimensions of medicine (physical, psychosocial, and spiritual) and emphasizes an interdisciplinary and collaborative team based approach to care [43]. Thus integration of multiple business processes such as a physician and nurse is often necessary. Developing technology to support palliative care is a significant challenge because much of it is provided as homecare. Unlike a patient in a hospital where data is accessed from the hospital's central dataset, a palliative patient will need to have data integrated from multiple data sources and displayed on different technologies such as a desktop at a physician's office or a mobile computer in the patient's home. Two, much of palliative care is monitoring of symptoms and responding to changes in symptom severity of which a key part is continual assessment and early patient identification when issues arise [7]. When a patient presents with increased pain a physician needs to be aware of the event and have access to various healthcare data sources and business processes to respond to the event. Monitoring and responding to symptoms can easily be done when a patient is in hospital but is far more challenging when a patient is at home. Three, palliative care delivery, particularly home care, is dependent upon policies that are very dynamic. For example, a physician may be monitoring several patients during the week but during the weekend another physician handles the monitoring. Polices are also used to identify who can perform certain tasks in response to an event such as changing a prescription dose or ordering a prescription refill.

3.1.1 Example Scenario: Severe Pain Management for Palliative Home Care

We now present an example scenario of severe pain management for a patient receiving palliative home care with a view to understanding how information related to care can be shared and managed in a B2B palliative care network.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 1 / APRIL 2010 / 56-70

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

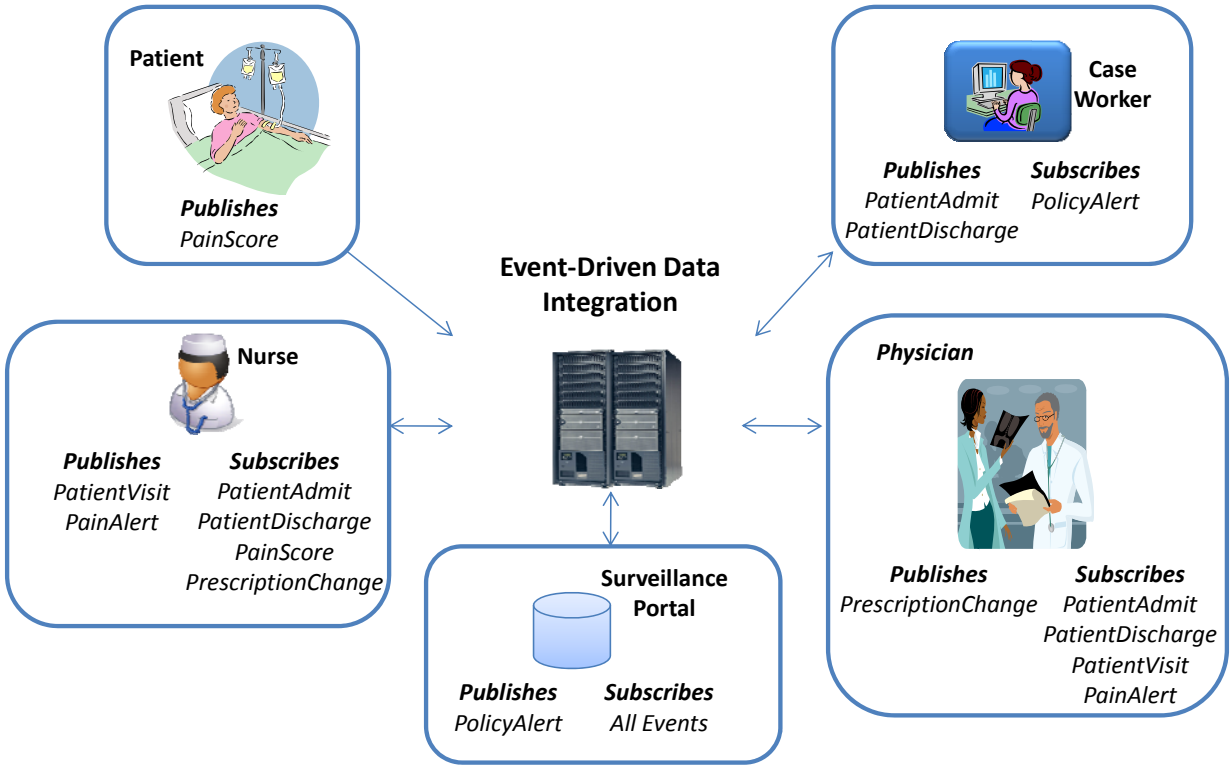

Figure 1: Palliative Severe Pain Management Scenario

The scenario was developed in collaboration with homecare nurses that are participating in our research. The scenario is shown in figure 1. Severe pain is defined as a pain score of seven or greater on a ten point scale. A patient is admitted into the palliative care network by a case worker who publishes a PatientAdmit notification that is received by her homecare nurse and her primary physician. Her PainScores are published periodically (using her home PC, PDA or through an automatic sensor) and tracked by her nurse to see how she is faring. If a nurse notices an anomaly with the patient`s pain scores, like an upward trend toward severe pain, then the nurse may publish a PainAlert event. The physician receives any PainAlert events related to the patient. Based on a PainAlert, the physician may decide to change the patient's prescription for pain medication and a corresponding PrescriptionChange notification is published. It is important to note that the Surveillance Portal captures all events that occur in the network in order to provide integrated reporting and performance management of care across the entire network. The Surveillance Portal is also able to raise alerts when policies related to timeliness of care are being violated. For example, a nurse is supposed to visit the patient at home at least once a week, and there must always be a visit with 24 hours following a PainAlert event or PresecriptionChange event. The nurse publishes a PatientVisit message for each such visit. The Surveillance Portal will publish a PolicyAlert event if a PatientVisit does not occur within the specified timelines. The CaseWorker subscribes to such events and can follow up to ensure proper care is received by the patient.

In this scenario, each participant (patient, nurse, case worker, physician, surveillance portal) potentially belongs to a separate organization each with their own platform for capturing and managing data. As a result, the communication and sharing of information is taking place across a B2B network in which the different organizations are cooperating. The event-driven data integration framework that ties them together must provide a common mechanism for publishing event data for use by others in the palliative care network. It must do so in a timely, convenient fashion so that the right people have the right data at the right time. It must do so in compliance with legislation, guidelines and defined procedures that protect patient data and govern quality of care.

Figure 2 illustrates an example of the *painAlert* messages that are published on a continuous basis in the system. It shows all messages originating from all nurses for all patients. However, a particular physician is only interested in seeing the messages that are relevant to them. For example in Fig.2, we have highlighted the messages that have been sent for patients whose primary physician is "Dr. Fred". Dr. Fred does not want to subscribe to all *painAlert* messages but just the messages that are relevant to him.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 1 / APRIL 2010 / 56-70

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

Sender	PatientID	Primary Physician	Pain Thresold	Date/Time	Description
NurseA	**P12345**	**Dr. Fred**	**10**	**10/10/2008 14:00**	**Pain is much worse**
NurseB	P24348	Dr. Wilma	7	10/10/2008 15:00	Patient requires prescription refill
NurseC	**P25789**	**Dr. Fred**	**8**	**10/10/2008 16:00**	**New pain in the left leg**
NurseD	P31268	Dr. Betty	9	10/10/2008 17:00	Prescription not working

Figure 2: Example painAlert data stream

Figure 3 shows an example of how this could be expressed in terms of a subscription to *PainAlert* events on behalf of Dr. Fred. The subscription is expressed as a policy, or set of rules that route only relevant *PainAlert* events to Dr. Fred. Dr. Fred requires all *PainAlert* events meant for him to be delivered immediately to his system. However the policy is only applicable from Monday through to Friday, 09:00 to 16:00 when Dr. Fred is available to respond to alerts. The policy stops applying to messages after December 20, when Dr. Fred will be on vacation. The framework will have to ensure that other subscription policies must be put in place to ensure that the on-call physician will receive pain alerts when Dr. Fred is not available (evenings, weekends, vacations). Another subscription would be defined for after-hours service in which all *PainAlert* events received for a group of physicians before 9am and after 4pm during the week, and 24 hours on weekends, would go to the on call physician for the group. Flexibility is needed so that every participant in the B2B network can define which messages are relevant to them, during what time period, and how the messages should be delivered to them. Alternatively Dr. Fred's subscription could be defined to accommodate hours where he is not available. At those periods, the messages meant for Dr. Fred are cached by the message broker and delivered when he comes online during the regular work hours.

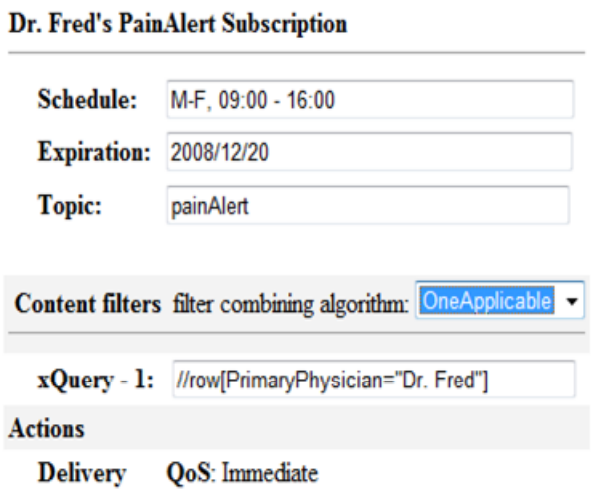

Figure 3: PainAlert subscription for Dr. Fred

3.2 Key Information Management Requirements

Based on a review of existing processes and infrastructure for palliative care in our case study with the local health authority in Ottawa, Canada and a literature review of health care monitoring processes and related ehealth information systems, we have identified the following key information management requirements for eHealth monitoring processes operating in a B2B environment. The first three requirements (event registry, platform independent, flexible data delivery) provide the technology platform for event-driven data integration (shown in Figure 1) which can feed the Surveillance Portal with the data it needs to provide the last two requirements (performance management and policy compliance). Many of these are problematic for traditional SOA frameworks. In section 4, we describe a framework which was able to adapt the traditional SOA framework to address them.

3.2.1 Event-Data Publishing and Registry

In order to provide a consistent view of events across a B2B network where data is shared between different organizations, there must be a coordinated (possibly centralized) mechanism for registering the events organizations publish and defining the data associated with them in a consistent data model. This includes a consistent approach to managing things like time and identification for resources and people. The registry must be made available in an open standardized fashion to ensure that all participants in the network are aware of the data available to them.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 1 / APRIL 2010 / 56-70

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

3.2.2 Platform Independent Data Integration

Typically, each organization in the network will have invested in its own platforms and processes for managing its data. Thus there must be an open, standard set of interfaces that allow organizations to both publish and receive event data in a format friendly to their chosen platforms and processes that is consistent across the network and does not impose major development efforts or performance penalties. It must be capable of supporting interfaces to a variety of hardware from remote locations e.g. mobile devices, laptops, computers, servers.

3.2.3 Flexible and Controlled Data Delivery

Traditional B2B interaction in an SOA is usually strongly coupled and point-to-point. The calling application calls a service directly and either waits for it to respond before continuing its own execution or expects an asynchronous response in near real-time. Strong coupling results in fast execution when all interacting services are online and responsive. However, whenever there is a disruption to communication as a result of internal or external factors, the entire execution process fails. Even under normal circumstances, strong coupling is problematic for eHealth monitoring processes, since it requires constant polling for all events that are potentially relevant, even if they are rare in occurrence.

Participants in an eHealth monitoring process should have a controlled, flexible mechanism for identifying or subscribing to the events of interest to them and receiving them at the time most suitable for them, in the form most appropriate. Having all events of a given type pushed automatically to subscribers as is typical of the most common publish/subscribe frameworks, is just one option. Participants should be able to filter the messages they want to receive, have them transformed into a format convenient for their processing and they should be able to flexibly specify whether they want them pushed automatically or cached for scheduled and on demand delivery.

3.2.4 Monitoring and Performance Management

Events in an eHealth network need to be monitored and reported on in terms of:

- The sequence of events that define the history of activity of an individual health care provider.
- The sequence of events that define the history of care for a single patient across the network.
- The aggregation of events across all providers and all patients that measure the overall quality of care provided by the network.

The monitoring needs to be continuous with support for generation of alerts where appropriate in near real-time. Full performance management reporting should be facilitated off a persistent relational store.

3.2.5 Policy Compliance and Enforcement

Health care must be agile and flexible in terms of resource assignment and team communication, while ensuring compliance with legislation, guidelines and defined procedures that protect patient data and govern quality of care. Business rules which control sharing of information cannot be hard coded but must be declaratively defined and dynamically configured. A systematic mechanism for enforcement and documenting compliance must be in place, which allows for centralized oversight while still permitting individual organizations visibility and input.

4 Design of Policy-based Publish/Subscribe SOA Framework

We now present a framework that has been specifically designed to address the information management requirements of ehealth monitoring processes in a B2B environment. It incorporates a few basic extensions to the traditional SOA approach. The framework has been implemented as a prototype to specifically address pain management for palliative care based on a case study at the local health authority [36]) in Ottawa, Canada. In our initial approach to the problem, we reviewed an ERP Web Portal system that had been proposed for the local health authority, with the intention of proposing a traditional SOA BPEL framework. However, we quickly found that such a framework was not well suited for applications focused on sharing and monitoring event data between different organization partners in a B2B network. As a result, we were forced to develop extensions to the traditional SOA work that focused more on data integration and policy automation and less on automation of transactional processes. These extensions are described in this section.

Figure 4 shows the architectural template for our framework which implements the event-driven data integration component shown at the center of the scenario in figure 1. All data sharing between the participants (physician, nurse, patient, case worker, surveillance portal) in the figure 1 scenario is accomplished through a publish-subscribe style interaction with a centrally controlled Policy-based Message Broker installed on a server and accessible via SOAP over HTTPs.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 1 / APRIL 2010 / 56-70

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

Each participant has a monitoring device, laptop, desktop, or server computer associated with them on which a Partner Service Component is installed that manages their interaction with the Message Broker. The Partner Service Component can publish a continuous stream of events associated with a participant to the Message Broker. At the same time, it also enables the participant to subscribe to a continuous stream of events, generated by other participants that are delivered to the Partner Service Component by the Message Broker. These events are transmitted as messages, with published messages being sent from an Outbox managed by the Partner Service Component, and subscribed messages being delivered to an Inbox.

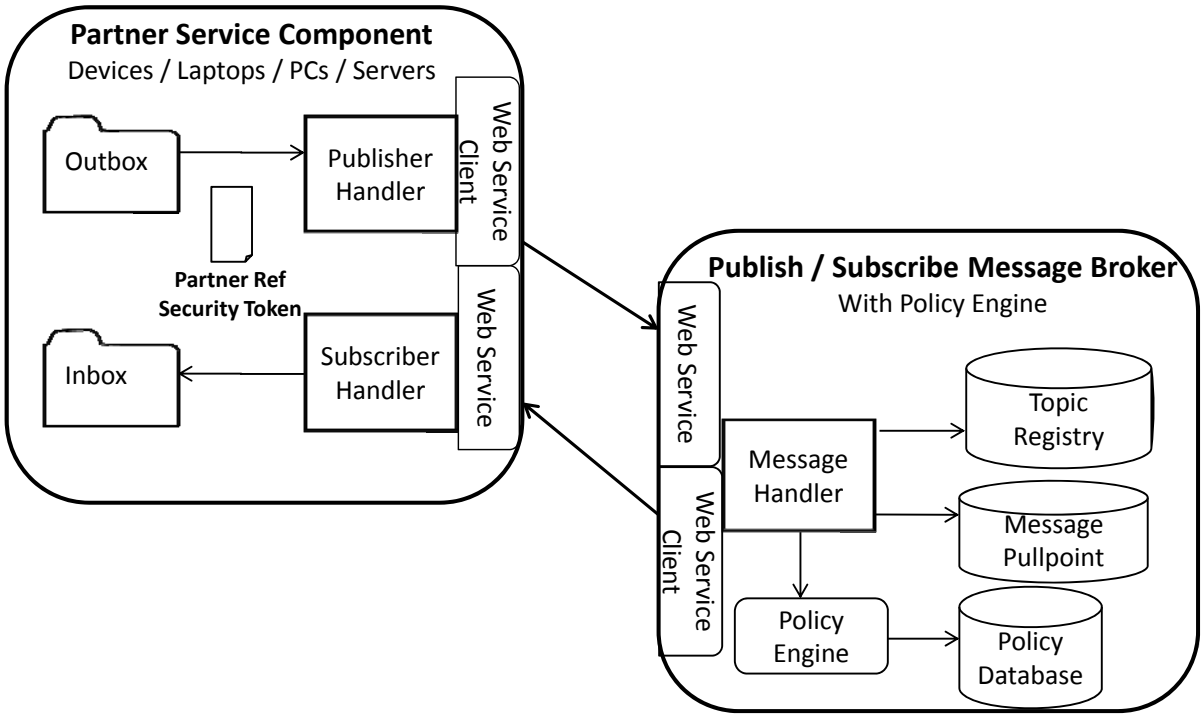

Figure 4: Policy-based SOA Publish/Subscribe Framework

The Message Broker maintains a registry of the types of events, called Event Topics, and the types of data, called Event Attributes, used to define those topics. This ensures a consistent event data model across the B2B network. A special Partner Service Component in the Figure 1 scenario is the Surveillance Portal. It automatically subscribes to all events in the B2B network and maintains a streaming data model of all events in the network that can be used to publish alert events as well as provide comprehensive performance management reporting.

The Message Broker is also the central policy enforcement point for the B2B network since all sharing of event data through Partner Service Component subscriptions must pass through it. Policies are defined and stored in a Policy Database and enforced by a Policy Engine that control who can publish what events and restrict who is subscribed to receive what data and when. The policies are also used to flexibly customize the format and timing of how event data is delivered for each subscriber (as was illustrated in figure 2 and figure 3). The Message Broker can cache messages in a Message Pullpoint to provide the greatest possible flexibility with respect to timing and format of delivery.

4.1 Partner Service Component

The Partner Service Component communicates with the Message broker through a single, generic set of SOAP interfaces. The Message Broker supports a SOAP interface that a Partner Service Component can invoke in order to *Register* an event topic and its attributes with the Message Broker, *Publish* one or more messages to the Message Broker for an event topic from the Outbox, and *Subscribe* to receive event messages for an event topic. The Partner Service Component supports a SOAP interface, *Notify*, which the Message Broker can use to "push" event messages to the participants Inbox, or the Message Broker will cache messages at a Message Pullpoint until the Partner Service Component collects the messages using the *getMessages* interface of the Message Broker.

Each Partner Service Component is uniquely identified and authenticated by the Message Broker by a Partner Service Security Token. This security token is an encrypted data structure that must be renewed each time a Partner Service Component comes online and registers with the Message Broker. Every event published from each Partner Service Component must have a valid Security Token or be discarded by the Message Broker. Note that

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 1 / APRIL 2010 / 56-70

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

policies may be predefined at the Message Broker (described in 4.4) which restrict what interfaces it is allowed to use in its interactions with the Message Broker, and restrict what event data will be published and subscribed to.

In our scenario from figure 1, the physician will receive in his Inbox all *PatientAdmit* and *PatientDischarge* events related to patients for whom he is listed as the primary physician. He will also receive all *PainAlert* events for these same patients as described in figure 2 and figure 3. If he responds to a *PainAlert* by changing a patient's prescription, then a *PrescriptionChange* event placed in his Outbox will be published to the Message Broker and distributed to all subscribers.

The publish/subscribe framework allows for dynamic routing of messages. In a traditional SOA approach subscribers would either have to poll and continually check if there were messages for them, or a predefined process would deliver messages to predefined recipients. For example, if Dr. Fred is the "on-call" physician one weekend, a second subscription policy could trigger delivery of *PainAlert* events for all patients that weekend, not just the ones he is the primary physician for.

4.2 Policy-based Message Broker

The behavior of the Message Broker is controlled declaratively by policies stored in the Policy Database and a Policy Engine which executes them in response to message events. Any request through its SOAP interface (*register*, *publish*, *subscribe*, *notify*, *getMessages*) is first processed by the Policy Engine to see what policies apply. These can result in:

- the message being rejected,
- the message being duplicated for routing to multiple subscribers,
- the message being cached for scheduled or on request delivery to a subscriber,
- the message being delivered to a subscriber.

In addition, when the Message Broker is about to send a message to a subscriber, subscription policies can apply so that the content of the message can be transformed or filtered. In that way, Dr. Fred only receives the *PatientAdmit*, *PatientDischarge*, and *PainAlerts* that are relevant to him, and only the particular information from the event that is of concern to him.

Figure 5: Notify Message Structure

Message pullpoints can be cached physically at the Message Broker. This is already done temporarily for each message in order to process the policies associated with it and ensure all subscribers receive delivery. Or message pullpoints can be defined virtually and implemented by interacting with the Surveillance Portal. We have implemented the former in our prototype, but are also investigating the latter.

Messages are defined as structured XML Documents consisting of a list of simply typed, attribute values pairs wrapped with specific tags for managing and processing the message. This XML structure greatly simplifies the application of policies, and transformation of messages. Figure 5 shows the structure of a Notify Message. The main event information is contained in a list of *MessageAttributes*. Any rich content (images, files, etc.) that does not

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

fit a simple attribute can be attached as *MessageContent*. *PublisherReference* (who sent the message), *SubscriptionRefererence* (which *PartnerServiceComponent* the notify message is going to), *MessagePolicy* (the subscription policy that controlled delivery), *TimeSent* and Topic are all information used to manage the message.

4.3 Topic Registry, Surveillance Portal, and Common Data Model

Before a Partner Service Component can start publishing events, an event topic and its associated event attributes must be registered with the Message Broker and stored in the Topic Registry. The Topic Registry, in effect, defines the common data model for all events in the network. With proper management and coordination of the registry, attributes will have a consistent meaning and can be used to link events. Effectively, each event type defines a dataset of streaming events, and certain event attributes (patientId, timestamp, nurse, primaryPhysician) can be used to link and correlate events along common threads.

The Surveillance Portal is a special Partner Service Component that automatically subscribes to all events published to the Message Broker. It is responsible for persisting and maintaining a complete history of all events in the network, and it is responsible for monitoring and correlating events in near real-time to detect and raise alerts related to policy compliance. The Surveillance Portal maps each event type to a relational table. Each simple attribute defined for the event type maps to a column in its relational table. The event attributes that can be used to link events are, in effect, foreign keys in this relational view. The Surveillance Portal manages a relational data stream model that reflects the event history of the network. Depending on the volume of event data streaming through the network, it may not be practical to persist the entire event history, but a detailed enough snapshot must be persisted to support performance management reporting.

For example, the data stream of PainAlert events shown in Figure 2, and the related PainScore events from patients have the following event attribute definitions:

- **Event** PainAlert

  Attributes:
  - Sender partnerServiceRef
  - Patient ID
  - PrimaryPhysician ID
  - PainThreshold integer
  - Date/Time timestamp
  - Description text

- **Event** PainScore

  Attributes:
  - Sender PartnerServiceRef
  - Patient ID
  - PainScore integer
  - Date/Time timestamp

But the Surveillance Portal might choose to only persist the following PainAlertSummary table that combines the two by adding columns that summarize the sequence of pain scores leading up to the PainAlert.

- **Table** PainAlertSummary

  Columns:
  - Patient ID
  - ReportedBy partnerServiceRef
  - PrimaryPhysician ID
  - PainThreshold integer
  - ReportDate timestamp
  - Description text
  - PeriodStartDate
  - MinScore integer
  - MaxScore integer
  - AverageScore integer

4.4 Policy Engine

The Message Broker uses the policy engine to execute subscription policies on messages. As each message is processed by the Message Handler, the message and its context is passed to the Policy Engine where the

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 1 / APRIL 2010 / 56-70

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

applicable polices are processed. Each policy has a set of rules containing specific conditions which, if matched, can results in some actions being taken in response. The actions can dictate message transformations and service delivery. Our policy engine is based on the XACML which is an open, standardized and widely used policy language, but other mechanisms, including other rule-based engines could be used. The key point is that the Message Broker provides a convenient Policy Enforcement Point, and by using a Policy Engine and Policy Database, policies can be declaratively defined and configured in a manner that is open to all organizations in the network and dynamically responsive.

The structure of a policy is shown by the definition in figure 6. Each <*Policy*> element uses a type attribute to describe the policy type. In this model, a <*Policy*> defines a set of rules, described by the <*Rule*> element. Each rule uses an optional <*Schedule*> as well as <*ExpirationTime*> to define the rule application schedule and expiration period. A rule supports *topic*, *content* and *publisher* filters. The <*TopicFilter*> element contains one or more <*Topic*> elements that define the topics for messages that it applies to. <*ContentFilter*> element contains one or more XPath queries to test the content of messages defined in <*Query*> elements and can be combined on the basis of the filter combining algorithm. Each <*Query*> is an XPath expression that executes to a Boolean result. If multiple queries are defined, the results of individual <*Query*> execution results are combined on the basis of the *filterCombiningAlg*. The <*Delivery*> element defines message delivery service quality. The <*SQ*> element values can be set as "BestEfforts" (default delivery mode and requires no receipt acknowledgment), "Reliable" (a message is considered as delivered only when there is a positive acknowledgement) or "Pullpoint" (messages are automatically cached for delivery at a later time).

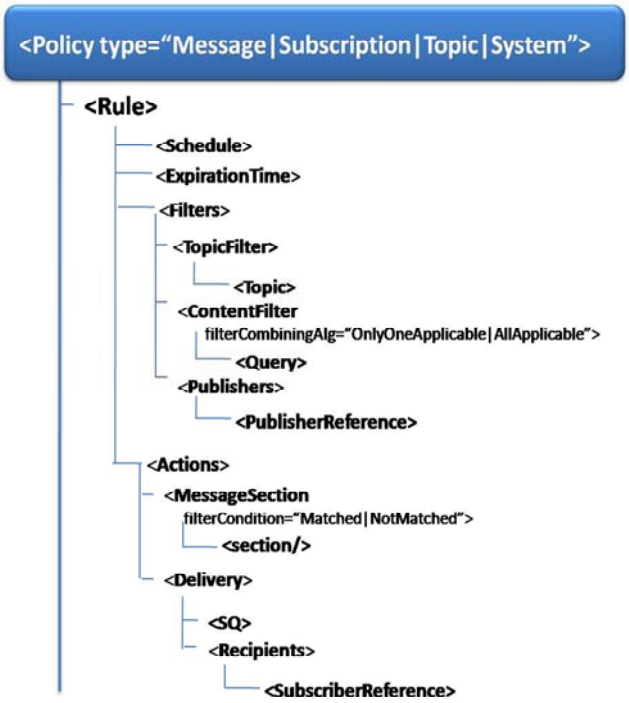

Figure 6: Policy Language representation

Using the subscription policy for Dr. Fred in Figure 3, we can trace how the Message Broker would handle incoming *PainAlert* messages such as the first message from NurseA in the *PainAlert* data stream shown in Figure 2. When the Message Handler receives the message from the NurseA Partner Service Component, it is passed to the Policy Engine which looks for policies that match the message. Dr. Fred's policy is a candidate since it applies to *PainAlert* messages, and NurseA has sent a *PainAlert* message. According to the <*Schedule*> and <*ExpirationTime*>, Dr. Fred's subscription policy only applies Monday to Friday, 9-6pm up to December, 2008. NurseA's message was sent at 14:00 pm on Friday, October 10, 2008 which matches Dr. Fred's subscription. Finally, the <*ContentFilter*> for Dr. Fred's subscription, indicates that Dr. Fred should only receive messages for which *PrimaryPhysician = "Dr. Fred"*. Again this is the case for the NurseA message, so all conditions of Dr. Fred's subscription policy have been met. There is only one action defined under Dr. Fred's subscription policy and that is to deliver the message immediately as is. Therefore the message handler is triggered to take a copy of the message and immediately send it to Dr. Fred's Partner Service Component using the *Notify* interface.

Many policies can apply to the same message. For example, there is a subscription policy for the Surveillance Portal which matches all event topics, with no restrictions on Schedule, ExpirationTime, or Content, with the same action to deliver immediately. So, based on that policy, the Policy Engine will trigger the MessageHandler to make a second copy of the NurseA message and forward it on to the Surveillance Portal Partner Service Component using the *Notify* interface as well.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 1 / APRIL 2010 / 56-70

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

5 Evaluation of Proposed Framework

In table 1 we compare our proposed framework with two other typical approaches to information systems that could be used to support eHealth monitoring processes in a B2B environment where data is shared between organizations, similar to what we have illustrated with our case study on palliative care at the local health authority in Ottawa, Canada. The evaluation of our framework is based on a prototype we have implemented to address palliative care. The comparison with an ERP Web Portal is based on our analysis of a system currently being evaluated at the local health authority in Ottawa, Canada [30], while our evaluation of SOA-BPEL is based on previous work to supported trusted B2B processes in a SOA-BPEL framework [14]. The difficulties we encountered in attempting to apply our previous SOA-BPEL framework to palliative care in fact has been the major motivation in the development of the new framework.

Table 1: Comparison of frameworks to address information management requirements for eHealth monitoring

	ERP Web Portal	**SOA-BPEL**	**Proposed Framework**
Event Data Publishing Registry	A centrally controlled portal provides access and reporting for data collected from contributing organizations	Individual organizations publish web services to provide data (and functions). Integration of data from web services is defined as needed in the context of individual processes.	Event topics and attributes defined by contributing organizations and registered with a central broker to ensure consistency across the network, including handling of time and resource/people identification.
Platform Independent Data Integration	A single platform is imposed by portal. Organizations convert, transform or re-enter data to use within their platforms.	Each organization has its own platform for its data. However, for shared data, it must connect directly to each source and use different web service interfaces and data formats.	Each organization has its own platform for its data. For shared data, it connects to a single broker and uses a single standard web interface to receive data according to a single standard data model in the format of its choosing.
Flexible & Controlled Data Delivery	Control defined by portal. Users must search or configure portal to receive data each time they want it (on request or scheduled) in formats provided by portal.	Control defined by each individual web service. Users must find source web services and build custom code or processes to receive data each time they want it (on request or scheduled) in formats provided by individual services.	Control via centrally registered declarative policies for whole B2B network which are reviewable by organizations. Users receive event data as it occurs, when requested, on schedule, or as needed from intermediary pull points in a format they select based on subscriptions.
Monitoring & Performance Management	Data access via portal is monitored but true event monitoring occurs at organizations before portal is updated. "Snapshot" performance management possible depending upon the data integrated into the portal.	Data access monitoring is done by each web service (possibly into a central logging service). BPEL does support detailed process monitoring, but true event monitoring occurs at organization. Performance management difficult without a consistent data model	Each topic published is an event stream that can be monitored in near real time or sampled as appropriate. Surveillance portal captures a complete consistent picture of all event traffic in network as a relational database, including detailed integrated performance management with real-time alerts.
Policy Compliance & Enforcement	Policy enforced centrally by portal and defined in terms of what data published to portal and who has access to portal. Limited enforcement related to real-time health care events.	Policy enforced and defined by each individual web service. Central definition and enforcement of policies possible for BPEL but difficult without a consistent data model.	Policy enforced centrally by broker and defined in terms of subscriptions that define who receives what events when. Real-time enforcement of health care policy is possible through the surveillance portal which monitors all events in the network.

5.1 ERP Web Portal

The ERP Web portal connects the clinical management, patient care, and severe pain management applications as well as a number of data sources into one application portal. All parties (patients, nurses, physicians, case works etc.) can access and use the application portal in order to update and see the current status of palliative care as it concerns them. Background system processes can extract, transform and load relevant data from other data sources. The portal does an adequate job of data integration in order to provide a view of palliative care in the B2B network within a single consistent data model, and a single centralized point of control where policy compliance can be managed and enforced. However, it does this at the cost of imposing a fairly rigid structure that requires much duplication of effort in order to convert, import and maintain the centralized copy of data. This processing imposes delays that ensure the portal provides at best a "snapshot" summary of events that took place sometime in the past.

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

Continuous monitoring and sharing of events in real time is not possible. For example when used for our scenario, Pain Scores have to be keyed-in by the patient from a web browser. This framework has no support for published data. Interactions are interface driven not data-driven. Data sources are statically linked and strongly coupled. Rules of exchanges and message transformation are defined in low-level business rules embedded in views, stored procedures, methods and objects that define the application logic. These are usually not dynamic enough to capture new requirements dynamically. The main issue is duplication of data, entering of data, and in particular delays in entering data originally captured on paper which can be measured in days and even weeks.

5.2 SOA BPEL Framework

The SOA BPEL framework supports integration across the B2B network by abstracting functions performed by different organizations as services and using BPEL scripts to define repeatable business processes that can execute services provided by different organizations in an integrated fashion. The SOA BPEL framework is more general and flexible then the ERP Web Portal. In particular, it allows organizations to avoid the effort of converting to and from a single monolithic platform. Instead each organization can maintain their own platforms and simply provide a web service interface (using standards like WSDL, SOAP and UDDI) to publish access to their services that any organization can use.

In many ways, however, it is this flexibility and generality that makes problematic the creation of a single, consistent, complete data model required for eHealth monitoring. Each organization defines its own interface and makes data accessible in the particular format it uses. There is no mechanism to ensure that data attributes are named and typed consistently, nor is there any mechanism to standardize how time or identification of people and resources is handled. Processes defined in BPEL can be centrally managed and monitored by the BPEL engine that executes them and a centralized logging service could be used in a systematic way to collect data for performance management. It is up to individual organizations to collaborate and negotiate this on a case by case basis. Access control and policy enforcement is largely done independently on a service by service basis.

5.3 Policy-based Publish/Subscribe SOA Framework

Our proposed framework provides platform independent data integration across an eHealth B2B network using web service protocols, while at the same time, ensuring a single, consistent, controlled, and flexibly evolving data model for continuous monitoring of events using a policy-based publish/subscribe message broker. The Partner Service component provides a single standardized mechanism for any event source to integrate into the eHealth B2B network or for any client to monitor the stream of events from that source. The message broker is not just an intermediary between event source and clients. It provides a central registry and a central policy enforcement point for coordinating a consistent, common data model across the B2B network in compliance with policies that control subscription access and delivery. The surveillance portal is a special client that can map the streaming events into a continuously monitored database of all eHealth events in the B2B network to support alerts and performance management for compliance with eHealth service targets.

6 Conclusions and Future Work

eHealth monitoring processes in B2B environments have specialized information management requirements that can be problematic in traditional approaches to SOA. We have shown how the traditional approach can be extended in a principled fashion to support policy-based data integration using a policy-based event-driven publish-subscribe framework. It should be noted that these extensions are complementary to traditional transactional process oriented integration using BPEL. Our framework is solely focused on event publishing and monitoring. So far, our work has only been applied to support palliative care information systems. More work is needed to confirm its applicability to other eHealth areas, as well as areas outside health care that require continuous event data monitoring like wireless sensor networks, energy distribution and billing, management of climatic sensors data, and supply chain management.

It has been beyond the scope of the paper to do a complete analysis of how best to encode policies and enforce them. The translation of legislation, regulations and procedures written in natural language into machine executable policies is an open area of research. Our use of XACML has been merely to illustrate that our proposed framework makes it possible. We recognize the importance of privacy to eHealth B2B data sharing. Access and authorization issues can be challenging for collaborative healthcare delivery [12]. Privacy compliance of shared data could be integrated into subscription polices by subscribers or policy administrators. However our framework did not delve into inter-organizational issues required for the management and enforcement of these polices. Future work is also needed to evaluate the scalability and reliability of the framework to provide real time monitoring of events. The processing and mining of large volume data streams, especially as they relate to service level agreements is a difficult and complex topic that is an active area of research. Finally empirical evaluation with users is necessary to test the implementation of the framework in actual healthcare settings to assess the patient, provider and organizational impacts of our research [39].

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

The work described in this paper is part of an initiative at the local health authority in Ottawa, Canada called 'Aging at Home' that has the mandate to develop technologies and policies to allow patients to remain in their homes through chronic disease management. The policy-based framework presented in this paper is part of the development of a comprehensive information system to support day-to-day homecare delivery for palliative care patients called 'Palliative Information System' (PAL-IS). As part of the communication and discussion of the research, the work from this paper has been presented to clinicians and decision makers and has been received positively. Future work will involve application of the framework to other symptoms (i.e. shortness of breath, nausea) as part of the design and evaluation of the PAL-IS system. The framework will also be applied to other areas of chronic disease management such as diabetes.

Acknowledgments

This work has been supported by the Champlain Local Health Integration Network, the Ontario Research Fund, and the Natural Sciences and Engineering Research Council of Canada. Dr. Morad Benyoucef provided us with useful guidance and feedback.

References

[1] G. Anderson, and J. R. Knickman, Changing the chronic care system to meet people's needs, Health Affairs (Millwood), vol. 20, no. 6, pp. 146-60, 2001.
[2] D. Avison, and D. Young, Time to rethink health care and ICT communications, in Proceeding of the ACM vol. 50, no. 6, 2007, pp. 69-74.
[3] J. Bailey, A. Poulovassilis, and P. Wood, An event-condition-action language for XML, In Proceedings of the 11th International Conference on World Wide Web, Honolulu, Hawaii, USA, 2002, pp. 486-495.
[4] A. Barth, A. Datta, J. Mitchell, and H. Nissenbaum, Privacy and contextual integrity: Framework and applications, in IEEE Symposium on Security and Privacy, 2006, pp. 184-198.
[5] D. Bell , S. Cesare , N. Iacovelli , M. Lycett, and A. Merico, A framework for deriving semantic web services, Information Systems Frontiers, vol. 9, no. 1, pp. 69-84, 2007.
[6] Y. Boglaev, Interchange of ECA Rules with Xpath expressions, in W3C Workshop on Rules Language for Interoperability, 2005.
[7] J. R. Brechtl, S. Murshed, P. Homel, and M. Bookbinder, Monitoring symptoms in patients with advanced illness in long-term care: A pilot study, Journal of Pain Symptom Manage, vol. 32, no. 2, pp. 168-174, 2006.
[8] C. Cahill, C. Canales, H. Le Van Gong, P. Madsen, E. M., and G. Whitehead. (2009, January). Liberty alliance web services framework: A technical overview, Liberty Alliance Project. [Online]. Available: http://www.projectliberty.org/specs.
[9] A. K. Y. Cheung, and H. Jacobsen, Dynamic load balancing in distributed content-based publish/subscribe, in Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware, Melbourne, Australia, 2006, pp. 141-161.
[10] E. Coiera, and E. J. S. Hovenga, Building a sustainable health system, IMIA Yearbook of Medical Informatics 2007, vol. 2, no. 1, pp. 11-8, 2007.
[11] I. Cummings, The interdisciplinary team, in Oxford Textbook of Palliative Medicine. 2nd edition. (D. Doyle, C. Hanks, and N. MacDonald, Eds.), Oxford University Press, Oxford; 1998 pp. 19-30.
[12] D. Daiqin He, and J. Yang, Authorization control in collaborative healthcare systems, Journal of Theoretical and Applied Electronic Commerce Research, vol. 4, no. 2, pp 88-109, 2009.
[13] D. A. Dorr, S. S. Jones, and A.Wilcox, A framework for information system usage in collaborative care, Journal of Biomedical Informatics, vol. 40, no. 3, pp. 282–287, 2007.
[14] C. Doshi, L. Peyton, Trusted information process in B2B networks, in Proceedings of the 10th International Conference on Enterprise Information Systems, Barcelona, Spain, 2008.
[15] B. Eidson, J. Maron, G. Pavlik, and R. Raheja, SOA and the future of Application Development, in Proceedings of the First International Workshop on Design of Service-Oriented Applications, Amsterdam, 2005.
[16] O. Etzion, M. Chandy, R. V. Ammon, and R. Schulte, Event-driven architectures and complex event processing, IEEE International Conference on Services Computing, Chicago, 2006, p. 30.
[17] P. T. Eugster, P. A. Felber, R. Guerraoui, and A. Kermarrec, The many faces of publish/subscribe. ACM Computing Surveys, vol. 35, no. 2, pp. 114-131, 2003.
[18] B. Eze, A Policy-based Message Broker for Event-driven services in B2B networks, M.S. Thesis, University of Ottawa, ON, Canada, 2009.
[19] I. Foster, C. Kesselman, J. Nick, and S.Tuecke, Grid services for distributed system integration, Computer, vol. 35, no. 6, pp. 37-46, 2002.
[20] T. Hammond, T. Hannay, and B. Lund. (2004, December) The role of RSS in science publishing: Syndication and annotation on the web. D-Lib Magazine. [Online]. vol. 10, no. 12. Available: http://www.dlib.org/dlib/december04/hammond/12hammond.html.
[21] I. Harrocks, J. Angele, S. Decker, M. Kifer, B. Grosof, and G. Wagner, What are the rules?, IEEE Intelligent Systems, vol. 18, no. 5, pp. 76-83, 2003.
[22] A. R. Hevner, S. T. March, J. Park, and S. Ram, Design Science in Information Systems Research, MIS Quarterly, vol. 28, no. 1, pp. 75-105, 2004.

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000100006

[23] M. N. Huhns and M. P. Singh, Service-oriented computing: Key concepts and principles, IEEE Internet Computing, vol. 9, no. 1, pp. 75-81, 2005.
[24] Java Message Service. (2008, December) SDN Library. Sun. [Online]. Available: http://java.sun.com/products/jms/.
[25] D. Kaye, Loosely coupled: the missing pieces of Web services, RDS Press, 2003.
[26] C. E. Kuziemsky, J. Weber-Jahnke, F. Lau, and G. M. Downing., An interdisciplinary computer-based information tool for palliative severe pain management, Journal of the American Medical Informatics Association, vol. 15, no. 3, pp. 375-382, 2008.
[27] S. T. March, and G. F. Smith, Design and natural science research on information technology, Decision Support Systems, vol. 15, no. 4, pp. 251-266, 1995.
[28] Y. Natis. (2008, December) Service-oriented architecture scenario. [Online]. Available: http://www.gartner.com/resources/114300/114358/114358.pdf.
[29] P. Niblett, and S. Graham, Events and service-oriented architecture: The OASIS web services notification specifications, IBM Systems Journal, vol. 44, no. 4, 2005.
[30] Ontario LHIN. (October, 2009) Local health information network. [Online]. Available: http://www.health.gov.on.ca/transformation/lhin/lhin_mn.html.
[31] OASIS Standard. (2008, December) OASIS eXtensible Access Control Markup Language (XACML). [Online]. Available: http://www.oasis-open.org/committees/download.php/2406/oasis-xacml-1.0.pdf.
[32] OASIS Standard. (April, 2007) Web services business process execution language version 2.0. [Online]. Available: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html.
[33] B. O'Connel, and A. Stanford-Clark, Using MQ Telemetry Transport with WebSphere Business Integration Message Broker, IBM DeveloperWorks, 2005.
[34] M. Papazoglou, and P. Ribbers, e-Business organizational and technical foundations, West Sussex, England: John Wiley & Sons Ltd, 2006.
[35] PHIPA Canada. (2009, October) Personal health information protection act, Government of Ontario 2004. [Online]. Available: http://www.e-laws.gov.on.ca/html/statutes/english/elaws_statutes_04p03_e.htm.
[36] L. Peyton, J. Hu, C. Doshi, and P. Seguin, Addressing privacy in a federated identity management network for e-health, in 8th World Congress on the Management of eBusiness, Toronto, 2007.
[37] G. Premkumar, K. Ramamurthy, and N. Sree, Implementation of electronic data interchange: An innovation diffusion perspective, Journal of Management Information Systems, vol. 11, no. 2, pp. 157-186, 1994.
[38] A. Pourshahid, D. Amyot, L. Peyton, S. Ghanavati, P. Chen, M. Weiss, and A. Forster, Toward an integrated user requirements notation framework for business process management, in International MCETECH Conference on e-Technologies, 2008, pp. 3-15.
[39] *Y*. Su, K.T. Win, J. Fulcher, and H. C. Chiu*, Measuring* end-users' opinions for establishing a user-centred electronic health record (EHR) system from the perspective of nurses, Journal of Theoretical and Applied Electronic Commerce Research, vol. 4, no. 2, pp. 55-63, 2009.
[40] N. Thomas. (2008, December) When should I use JMS? JDJ - Industry Commentary. [Online]. Available: http://java.sys-con.com/node/36813.
[41] Philippe Le Hégaret. (2003, December) The future of standards for web services and service oriented architectures, W3C. [Online]. Available: http://www/w3c.org/2003/Talks/1211-xml2003-wssoa/slide2-0.html.
[42] H. Weigand, W. J. Heuvel, and M. Hiel, Rule-based service composition and service-oriented business rule management, in Proceedings ReMoD, France, 2008.
[43] who.int (2009, October). World health organization definition of palliative care. [Online]. Available: http://www.who.int/cancer/palliative/definition/en/.