A FLEXIBLE POLICY ARCHITECTURE FOR MEDICAL INFORMATION MESSAGING

Edward Brown
Department of Computer Science, Memorial University of Newfoundland, St. John's, Canada

Jamie Goodyear
Progress Software Corporation, 570 Newfoundland Drive, St. John's, Canada

Keywords: Policy, Messaging, Architecture, Middleware, Medical context.

Abstract: We describe a design solution for the problem of changing policies regarding information management in the health care environment. Frequent policy modification regarding security, privacy and workflow through institutional re-organization and policy revision occur at multiple levels of administration, which can leave the health care information users with non-compliant systems (such as electronic medical records) and procedures which are expensive and difficult to update. Our solution is a medical information messaging infrastructure designed to respond flexibly to changes in information policy. Instead of embedding fixed policy into static application code, our architecture provides configurable policy rules as part of the communications framework. This entails two critical components: the dynamic router, which routes messages according to the policy rules, and the medical context header, which attaches policy-relevant information to all communication messages. All information applications are automatically compliant with policy, since it is enforced at the communications level of the system.

1 INTRODUCTION

Large scale communication infrastructure for medical information applications is essential if different medical applications, such as electronic medical records and electronic prescription services, are to be interoperable. In other words, information needs to be shared seamlessly across different applications, vendor systems, jurisdictions, and health care providers. The Canadian health care system, being substantially government funded and therefore having a close relationship between funding and regulation, is one context in which such an infrastructure can be designed and built as a federal initiative. The underpinning of this infrastructure in Canada is a common communication layer, called the Health Information Access Level (HIAL). With a common information exchange architecture, the HIAL solution provides for different applications to share and retrieve information stored in repositories throughout the system. (CHI, 2003)

However, questions of the legitimacy of information access are not addressed by any specific architectural feature. Policies regarding who, when and why information may be accessed and used are not explicitly represented in the system architecture. Instead, security and privacy solutions and services that can be deployed when policies are developed. These follow traditional security monikers and solutions, such as authentication, identity management and other software mechanisms. (Brown and Wareham, 2007)

The concern with these solutions is that policies regarding information exchange and access are re-created as part of each application. In each case, procedures for implementation of these policies, such as accessing an authentication service, are coded into the application programs. When policy changes occur, the applications may not be suited to any changes and the cost of updating the applications may place an unnecessary burden on health care budgets. We contend that such policy changes are in fact frequent, particularly at the level of individual organizations such as hospitals. (see Brown and Wareham, 2007, at 46)

Brown E. and Goodyear J. (2010).
A FLEXIBLE POLICY ARCHITECTURE FOR MEDICAL INFORMATION MESSAGING.
In *Proceedings of the Third International Conference on Health Informatics*, pages 70-76
DOI: 10.5220/0002737100700076

Consider, for example, the case of "reportable" diseases which require notice to a public health authority. The obligation to report such diseases typically falls on the primary care physician. (see HPPA, 1990 for one Canadian jurisdiction) An electronic information system could help the physician's information management burden, by automatically reporting such diagnoses. However, there are variations on regulations, process, and responsible authority between provincial jurisdictions. These policies may change or be amended fairly quickly, such as in response to a particular epidemic threat. If policy changes entail reprogramming of application code or circumvention of legacy procedures that are embedded in software applications, then software hinders, rather than helps, such reporting.

There are even more frequent information management policy changes on a more detailed institutional level. A particular lab or individual may be censured, health service units may be restructured, information workflow may be re-organized, institutional analysis procedures may be revised: these all have implications for how information is shared and accessed, which we refer to as *information policy*.

The tendency is to implement applications based on existing policies or regulations, without regard for the frequent re-interpretation or change of policy, due to administrative decisions (as per the preceding paragraph), new regulation, new technologies, innovation in health care strategies or even court decisions which alter obligations or liability. The ability to respond to such changes is compromised when the system enforces procedures and policies that were current or thought appropriate at the time the system was designed. Vendors that design, create and/or implement systems have little incentive to design to accommodate such future changes, particularly if they can look forward to being engaged and paid to overhaul their application program each time changes do occur.

Our prototype system deploys information policy at the communications layer of the system rather than within each application, making the system aware of the medical context of each communications message. This relieves the application code of the burden of compliance with (possibly changing) information policies.

Active use of medical context has precedent, including context-aware computing (Bricon-Scouf and Newman, 2007), linking of related medical events (Clerq, Bangels and France, 2004), annotating EHR records with disambiguating context (Manzoor, Ceusters and Rudniki, 2007), mobile access through user and location context (Hägglund et al, 2007), adaptive information for telemedicine communications (Doukas, Maglogiannis and Karpouzis, 2008), and hospital applications such as context aware pill containers or hospital beds (Bardram, 2004). This literature illustrates the importance of context not only in the operation of health care tools and software systems, but also in understanding needs of the user and patient.

Although there are application specific uses for this medical context information, such as tracking and logging medical events and data security forensics, we limit this discussion to messaging infrastructure advantages.

2 CONTEXT-BASED MESSAGING

Our problem is to provide for policy changes within the architecture of the medical information infrastructure itself. By making information policy explicit in the system architecture, rather than implicit (and possibly hidden) in the coding of each individual software application, we are able to change information access and information sharing characteristics as a system configuration exercise. Policy changes are new system configuration directives (that is, messaging rules), rather than re-coding of applications.

This architecture introduces two critical elements: first is the use of a *dynamic router* for messaging within the information infrastructure used by all applications; the second is attaching *medical context* headers to each message to which the dynamic router can respond.

2.1 Dynamic Router

A dynamic router will route messages according to a rules base which can be configured manually or automatically, and changed dynamically without halting any system operations or services. The router makes decisions about where specific messages are to be sent based on the message itself and the current rules in its rules base. Since the router is dynamic, the rules can be modified manually or through software that is appropriately authenticated for making such alterations. Routing messages based on current information management policies is a matter of translating the policies into routing rules which are added to the current rules base.

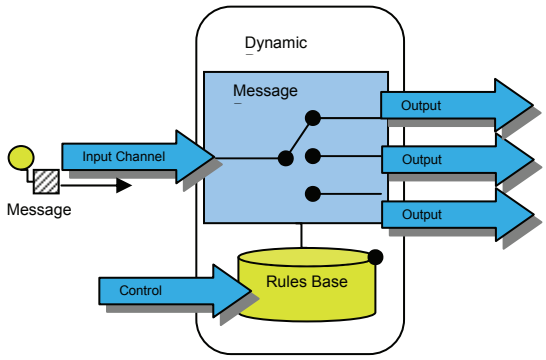

Figure 1: Dynamic Router.

The dynamic router appears in architectural design texts as an enterprise integration pattern whose application is well understood. (Hohpe and Woolf, 2003) Its use allows the explicit expression of routing policies in a consistent manner, and at a known and configurable point in the overall system. All applications will be operating in accordance with the policies encoded by the routing rules.

In the dynamic router architectural pattern (shown in Figure 1) the routing rules are retrieved from a rules data store. (Hohpe and Woolf, 2003) This frees message distribution from complete preplanning as routes may be added, modified, or removed at run time.

The dynamic router includes a rules engine (not shown). This component is responsible for accessing, modifying and storing rules for the router. For example, the rules engine may be directed to modify the rules base by adding or deleting rules, or retrieve the currently stored rules for inspection. Messages for the rules engine relating to administration of the rules base itself use a special "control" channel for its own messages.

The concept of a dynamic router is not itself original; it is in common use in financial services software and can even be seen in e-Health information frameworks. (Krasser, 2009) Our contribution is using the rules base for explicit information policies, rather than mere application configuration.

```
stjohns;Results;dr.farrell.response.que
ue;tcp://localhost:61616
stjohns;radiology;radiology;tcp://local
host:61612
stjohns;microbio;microbio;tcp://localho
st:61620
stjohns;clientreq;Q_RecordRequest;tcp:/
/localhost:61628
stjohns;ReportD;pub.health;tcp://localh
ost:61624
stjohns;Patient_Record;*meddrid;tcp://l
ocalhost:61616
```

Figure 2: Some routing rules as they appear in one test of the policy-oriented rules base.

Our implementation adopts the Apache Camel messaging framework, which provides information routing as an extension to server functionality. At present, the routing technique is simple: messages typically specify their intended destination, and further routing constraints or destinations are chosen by the dynamic router as each message arrives, by matching properties of the message against the rules base. (Some simple rules are illustrated in figure 2) More complex routing capabilities are feasible, but even this simple scheme allows intricate routing behaviours and basic policy statements to be represented.

The dynamic router is not sufficient by itself to allow the system to respond to the type of information policies we have been discussing. These policy directives (implemented as routing rules) are intended to operate across applications, and may be related to jurisdiction, treatment, nature of the medical event, type of information requested, origin of the request, identity of the individuals or organizations involved, intended use of the information, and other policy related characteristics

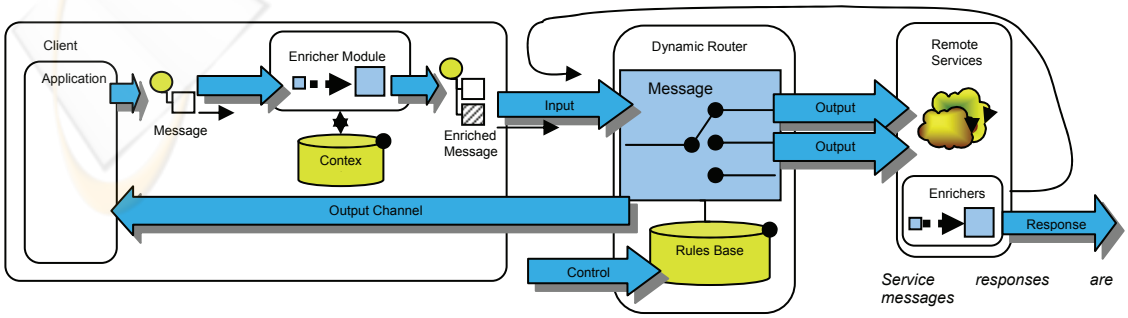

Figure 3: Messaging context affects routing.

of the message. This additional information, which we refer to as the *medical context*, must appear in a message in order for the router to have something to match its policy based rules. In a conventional software deployment, some of this information (such as intended use) would be implicit due to the particular application being used. Under our system, however, messages are not routed according to the particular software application, but according to multiple policies that apply regardless of the particular software application in use.

2.2 Medical Context

The second critical element for policy-based message routing is enrichment of messages to include the medical context information that might trigger a policy rule. Information about the source and destination of the message, and the function of the message in terms of data access, updates and requests would already be included in conventional message definition; only the medical context need be added. Once a message arrives at the dynamic router for delivery, the medical context for that message is matched against the dynamic router's set of rules, to find applicable policy rules that indicate how the message should be routed. The router then proceeds to deliver the message using the communication components of the architecture.

The content enrichment module (included in Figure 3) is used to add medical context information to base messages. This pattern is normally used in integration scenarios in which the message originator does not have all the required data available, so that additional information has to be injected into the message. (Hohpe and Woolfe, 2003) Some of the headers deployed in our current prototype are shown in figure 4. When applications need to communicate, the content enricher is called to add the medical context headers to their base messages. As a separate code package or module, the enrichment feature is available to all application programmers.

Our current prototype implementation uses Java Message Service (JMS) as the communication infrastructure. JMS is a well supported middleware package that provides messaging infrastructure among multiple clients. It is a fairly simple matter to enrich a JMS message with medical context information, simply by adding an additional header section to each JMS message. A simple message header after injection in our prototype system is shown in figure 5. The medical context header becomes a part of each JMS message, and is carried and delivered to routers and application code

medmsgtype: Control or Default type. Control messages are sent to the router's control channel.
medcommand: Commands issued to a router.
medpolicy: A policy in the form of routing rules.
meddrid: Healthcare provider identification. Used by router to construct a unique output queue.
medptid: Patient identification.
medwhere: Location of medical event (scoping is currently unstructured).
medaction: Situational information about medical event.

Figure 4: Some headers currently defined to inject medical context information.

transparently by the JMS infrastructure.

```
{commandId = 7, responseRequired =
true, messageId = ID:jamie-
goodyearsmacbook, Local-50300-
1240760119906-0:2:1:1:1,
originalDestination = null,
originalTransactionId = null,
producerId = ID:jamie-goodyears-
macbook.local- 50300-1240760119906-
0:2:1:1, destination =
queue://Q_Default, transactionId =
null, expiration = 0, timestamp =
1240760120317, arrival = 0,
brokerInTime = 1240760120318,
brokerOutTime = 1240760120319,
correlationId = null, replyTo = null,
persistent = true, type = null,
priority = 4, groupID = null,
groupSequence = 0, targetConsumerId =
null, compressed = false, userID =
null, content = null,
marshalledProperties =
org.apache.activemq.util.ByteSequence
@f7b44f, dataStructure = null,
redeliveryCounter = 0, size = 0,
properties = {medptid=555-555-5555,
meddrid=dr.farrell.response.queue,
medmsgtype= default,
medaction=microbio-TBTestReq-sputem-
HL7-2.4, medwhere=stjohns},
readOnlyProperties = true, readOnlyBody
= true, droppable = false}
```

Figure 5: A JMS message header after injection of medical context information.

These two critical architectural elements, the dynamic router and the enriched message, are not

radical new architectural design concepts; in fact, each of these elements are conventional, "off the shelf" ideas. The dynamic router is a published and understood enterprise integration pattern familiar to software architects, and extended message headers is a feature common to all electronic messaging frameworks, including JMS. What makes our system different is routing based on medical context information, at a level usually limited to technical considerations, such application, network load, client jurisdiction or data protection.

Figure 3 illustrates the combination of the two critical architectural elements. The medical context enrichment must be installed at the client site where the message originates, as this is where the context is known; it is effectively an add-on to the application. The dynamic router (which may be remote or local) becomes the means for that client to access the communications infrastructure.

3 A USE SCENARIO

Our initial implementation effort is framed as a response to the Canadian Health Infoway Reference Implementation Suite (CHIRIS, see CHI, 2005). CHIRIS is intended to simulate the construction of a cross-jurisdictional electronic health record, and includes Admit, Dismiss, Transfer (ADT) and EHR viewer applications. Unlike CHIRIS, our design is conceived around medical context and messaging rather than data access and modification. Rather than data access or retrieval calls, we conceive of messages as parts of a medical narrative. Handling of these stories about patient care is defined by the directives contained within a dynamic policy rules base. Polices are understood as rules about how to treat the messages that constitute the medical narrative.

The remainder of this section traces the operation of our prototype implementation for a particular use case sequence, or more specifically, a particular use of the system in which the physician consults with a patient, updates the patient chart, and requisitions a number of lab tests, one of which results in diagnosis of a reportable disease. A more formal description of the use case and its implementation can be found in Goodyear, 2009. The following description tracks the system operation in terms of the medical context and dynamic router: we are not interested in the particular application software or screen interfaces that the various users employ, but only in describing the messaging architecture.

Initially, the physician requests a patient record and receives the relevant record. The message transactions are controlled by rules in the router's rules base. The medical context is injected by the software client into a message to the dynamic router. The router, according to its rules, will pass the message on to the patient record repository (available as a remote service). The repository site injects its medical context and returns the record to the router as a response message. The response is routed back to the requesting physician. Several rules were involved is this exchange – the absence of any rule to pass the messages would have caused the data retrieval to fail.

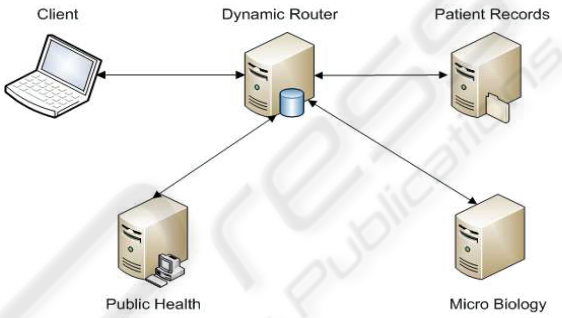

Figure 6: All messages in the scenario are sent through a dynamic router.

In the second portion of the scenario the physician requisitions a chest xray. The rules base contains an entry for xrays, so the request is forwarded to a radiology lab. After completing the xray procedure, the results are sent back to the requesting physician as a response message with the appropriate medical context injected by the lab software site. When the response is routed, a policy rule is matched for updating the patient record so the lab results are automatically copied by routing a message to the patient record repository as well as the requesting physician. The repository responds by sending a message notifying the primary care physician that their patient's record has been updated: in this test scenario, this is the same physician that ordered the lab in the first place. The physician receives two notifications in this scenario; first the lab results, and second notification of the patient record update. Rules that provide these notifications begin to show the benefits of our proposed architecture: in the CHIRIS reference system notifications occur only if the application developers program them into the application in use. The new design allows notifications to be routed by a simple policy rule change that automatically affects all applications.

In the final portion of the scenario, the physician requisitions a sputum test. The rules base contains an entry for sputum tests, so the request is forwarded to

the microbiology lab. The lab results are sent back to the requesting physician; however in our actual test we simulated a positive result for a reportable disease. The reportable disease is injected into the message as part of the medical context at the lab before the results message is sent to the dynamic router for delivery. When this results message is processed by the dynamic router, several rules are matched; the physician will get immediate notification, the patient record will be updated, and the public health authority is notified of the reportable disease. The physician will ultimately receive notification of the patient record update and an additional notification from public health that they are aware of the test result. The notification message from public health is also copied to the patient repository according to another routing rule. In this portion of the scenario, the medical context information is used multiple times to provide appropriate messaging. Each time a message is passed, the rules base was consulted to provide directives to where messages should be sent, without any programming of the client applications. Instead, the policy rules base provides a single consistent source of messaging directives.

4 DISCUSSION

Our proposed design requires all client applications and services to inject medical context into their messaging. The test of our prototype dynamic router involved the simulation of different health system deployments, including the lab and record repository services: these were simulated, not actual field deployed systems. In practice, the messaging facilities used by real world units would have to be modified to include the medical context, preferably by adding the message enrichment module to their softwares. Our team is currently pursuing support for a larger scale prototype and field testing of this architecture.

There are elements of our approach that need further design work. Most obvious is data security, which is usually handled on an application-by-application basis. With flexible routing of messages with added medical context, a comprehensive security solution which is responsive to rule changes is needed. Part of the security solution would have to consider to what extent a “bad actor” could hijack the system by inserting invalid rules, similar to server attacks currently familiar to internet service providers. Tools to help translate medical information policy to routing rules would be helpful as well, including some analysis capability to identify conflicts in policy or rules, eliminate circularity in routing, and ensure the rules were not used to subvert laws, security or privacy concerns.

An interesting question is how intelligent to make the dynamic router. It could be built to make fairly sophisticated decisions regarding conflicting policies from different agencies or levels of authority, to avoid or produce notification events, to monitor and track information sharing and medical decisions, or it could leave all important decisions to specific human intervention. It could also apply meta-policies about what policies can be changed or updated.

A related question is where the dynamic router and the rules based are physically housed. In particular, it is interesting to consider whether a single router with a single rules base offers advantages in predictability and reliability (but a single point of failure), is preferable to a distributed system of multiple routers with multiple rule sets for different jurisdictions and different levels of authority.

In the meantime, our prototype system with simulated health information services demonstrates the technical feasibility of using a dynamic router to implement health information policy, and realize the advantages recited in this paper: flexible change in information handling without the cost of redeploying new software; explicit articulation of health information policies, rather than implicit enforcement by individual applications; and no risk of being stuck with legacy software and processes that cannot be updated to reflect organizational or regulatory changes. Additional advantages in traceability of medical events and decisions, largely due to enriched medical context information, are discussed in Goodyear, 2009.

ACKNOWLEDGEMENTS

Thanks to Dr. Gerard Farrell and the members of Memorial University of Newfoundland's e-Health Research Unit and Medical Informatics Group for their support of the work herein described.

REFERENCES

Bardram. J.E., 2004. Applications of context-aware computing in hospital work: examples and design principles. *ACM symposium on Applied computing,* New York

Bricon-Souf, N. and Newman, C.R, 2007. Context

awareness in health care: A review. *International Journal of Medical Informatics*, 76, Elsevier

Brown, E., Wareham, H., et al., 2007. Technology Choices and Privacy Policy in Health Care. *Report to the Privacy Commissioner of Canada*, Medical Informatics Group, Memorial University, 2007, online: http://cpig.cs.mun.ca/TechnologyChoices

CHI, 2003. EHRS Blueprint, an inter-operable EHR framework Canada Health Infoway Inc, version 1.0 edition.

CHI, 2005. Canada Health Infoway. *CHIRIS User Manual.* Canada Health Infoway Inc, version 0.2 edition. online: http://sourceforge.net/projects/crrs.

Clercq, E.D., M. Bangels, M. and F. R. France, F.R., 2004. Integration of electronic patient record context with message context, *Studies in health technology and informatics*, 107(2), IOS Press.

Doukas, C., Maglogiannis, I., and Karpouzis, K., 2008. Context-aware medical content adaptation through semantic representation and rules evaluation, *IEEE International Workshop on Semantic Media Adaptation and Personalization,* Washington.

Goodyear, J., 2009. Extending the Health Information Access Layer with a medical context messaging framework. Honours Dissertation. Department of Computer Science, Memorial University of Newfoundland.

Hägglund, M., Scandurra, I., D. Moström, D., and Koch, S. Koch, 2007. Bridging the gap: a virtual health record for integrated home care, *International Journal of Integrated Care* 7(26), Igitur Publishing.

Hohpe, G. and Woolf, B., 2003. *Enterprise Integration Patterns : Designing, Building, and Deploying Messaging Solutions,* Addison-Wesley Professional.

Krasser, M., 2009. Introduction to the open eHealth integration platform, online: http:// architects.dzone.com/articles/introduction-open-ehealth, accessed July 7, 2009.

Manzoor, S., Ceusters, W.M. and Rudnicki, R., 2007. A middleware approach to integrate referent tracking in EHR system, *American Medical Informatics Association Symposium Proceedings.*.