Design of a Clinical Notification System

Michael M. Wagner, M.D., Ph.D., Fu-Chiang Tsui, Ph.D., Jeff Pike, M.S. , Lori Pike, M.S.
Center for Biomedical Informatics, University of Pittsburgh School of Medicine

We describe the requirements and design of an enterprise-wide notification system. From published descriptions of notification schemes, our own experience, and use cases provided by diverse users in our institution, we developed a set of functional requirements. The resulting design supports multiple communication channels, third party mappings(algorithms) from message to recipient and/or channel of delivery, and escalation algorithms. A requirement for multiple message formats is addressed by a document specification. We implemented this system in Java as a CORBA object. This paper describes the design and current implementation of our notification system.

INTRODUCTION

Communication, as discussed by Coiera (1), is a common and important activity in clinical medicine with many inefficiencies that represent an opportunity for information systems.

We can think of communication as being synchronous or asynchronous. A synchronous communication, such as a phone conversation, is one in which two or more parties exchange information in real time. An asynchronous communication is one in which a message is left for the other party to read. This latter type of communication is often referred to as a **notification** in the medical informatics literature.

Coiera's observational studies suggest that notification is underutilized in medicine. Due to a hitherto difficult-to-satisfy requirement for acknowledgement in most medical communications, personnel resort to synchronous communication (i.e., they page each other).

Recently, technology has provided channels such as 2-way alphanumeric pagers, and email that can be applied to this problem. Departments with data to communicate (e.g., laboratory) are using these channels to address specific immediate problems. This trend is not optimal, however. For each channel, not only must interfacing software be developed, but so must databases of address information (e.g., the fax number for Dr. J). Thus if we have *n* applications and *m* communication channels, in the worst case we develop *nxm* interfaces and databases (Fig. 1). Furthermore, consideration of human factors suggest that there is an upper limit of message volume through any communication channel after which users will become overwhelmed and may start to ignore the channel. A single notification system makes it easier to manage the use of channels.

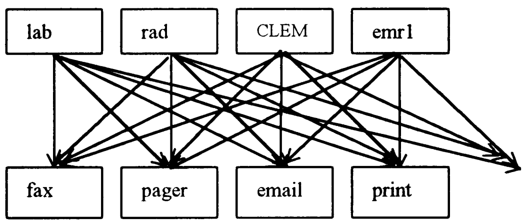

Fig. 1 Worst case scenario. *N* applications develop capability to communicate using each of *m* channels.

In this paper, we discuss our approach to notification. Our approach also supports synchronous communication as we shall discuss briefly.

MAPPINGS

The design of a notification system involves many issues such as security and fault tolerance but what is unique and fundamental to notification is the need for mappings. Notification requires mappings from (1) characteristics of the data (such as which patient it describes or the type of data) to a set of recipients, (2) recipients to communication channels that can be used to reach those recipients, and (3) to specific destinations in that channel. For example, laboratory result R maps to recipients Drs. X, Y, and Z using e-mail to addresses Q, R, and S.

Frequently, mappings occur to roles. For example, laboratory results for patient X map to the person filling the role of primary intern for X.

Other mappings are sometimes required. In a system described by Geissbühler (2), a notification to nursing staff of an order for patient X is mapped to the location of patient X and then to the object that maintains the status board screen of the physician order entry application running on workstation number 3. "Service pagers" are an existing notification scheme in which communications of certain types go to a designated device, which is assumed to be in the possession of the individual responsible for handling such communications.

Escalation schemes, which we discuss, specify multiple mappings that are applied conditionally and sequentially (e.g., first, map information type A to primary intern; If receipt isn't acknowledged within 30 minutes, map information type A to primary resident; If delivery isn't acknowledged in 30 minutes, map information type A to supervisor.). Another feature of the mappings in escalation might be described as the need for late binding. If the second recipient is specified as "the person currently filling the role of responsible intern" then the actual individual will be whoever happens to fill that role 30 minutes in the future.

We can think of mappings as the knowledge base of a notification system. Like knowledge bases, they vary in expressiveness from system to system, require extensive effort to develop and maintain, and there are reasons to keep them separate from other components of a notification system.

PREVIOUS WORK

Our design is influenced by three sources: (1) our own experience with notification in event monitoring and result distribution applications (3, 4), (2) published work, and (3) use cases that arose in design meetings with various groups in our institution. In this section, we discuss published work and in the next, use cases.

In Geissbühler's publish-subscribe paradigm (2), a notification message is published by, for example, a laboratory computer system. This notification includes a description of itself in dot notation (e.g., lab.123456789.K.abnl) that can be inspected by subscribing applications. An example of a subscribing application is a pager system. The subscribing application describes the class of message types that it wants to receive using the dot notation language, which includes asterisks as wild card characters. For example, to register to receive all abnormal lab results on patient 123456789 it would say lab.123456789.*.abnl. The description of this approach does not specify completely which mappings are supported and where they reside in the system. However, the need for multiple mappings, including ones with escalation properties, is articulated. Some limitations of this publish-subscribe model are (1) it does not provide a solution to the need for multiple message formats as would be required if more than one communication channel could be used for a notification and those channels had incompatible formatting requirements that could not be satisfied by automatic translation; and (2) it does not address access control, or encryption issues.

All clinical event monitors include some form of notification, but they do not have general models of the process. For example, in the Arden syntax the mappings from type of information to recipient occur within the medical logic module that processes the type of information. The notification logic is hard coded into the action part of the rule and there is no concept of escalation, fail safe delivery, or acknowledgement.

Brigham and Women's alerting system has a notifier that supports such features, however it has not been elaborated into a general model (5).

USE CASES

Use case 1: Current system to distribute results
A laboratory or radiology computer system generates an HL7 message which is received by a server that creates a notification message with delivery instructions. This server calls the notification system with the message and delivery instructions. The delivery instructions specify which mappings to use. The mapping to recipients is role-based (e.g., ordering doctor, primary intern, resident). The mapping to communication channel is based on recipients' preferences and recipient can have different preferences based on his or her roles.

Use case 2: Future distribution of results
As in 1, except we assume that systems can generate messages and call notifier directly (or via an ORB).

Use case 3: Critical lab alerts
As in 2, but also requires escalation with fail-safe property to satisfy JCAHO requirements.

Use case 4: Radiology preliminary results reporting
Radiologist uses an existing radiology application to record preliminary reading of image. This application then (1) composes a message that contains a pointer (hotlink URL) to the image on a radiology server, (2) calls notification system with message and unique ID of ordering physician, (3) receives acknowledgement and status messages from notification system, (4) tracks acknowledgement status of this and other notifications on a status screen for the radiologist.

Use case 5: Home health.
Home health uses a workflow engine to manage authorization paperwork. This engine calls notification system with URL of forms for clinicians to sign. Mappings and escalation algorithms are provided and maintained by home health.

Use case 6: Assistant who triages notifications

Dr. J's assistant triages new data for Dr. J. This assistant uses a client application that receives notifications destined for Dr. J. The assistant uses the application to send notifications back to the notification system for redelivery to Dr. J by the appropriate channel.

Use case 7: Integrated Call Center
A Center is charged with improving communication to referring clinicians. These referring clinicians may not be directly affiliated (and their addresses and identities are therefore known only to the Center). Center application programs generate messages, then call notification system to deliver by fax or email based on referring clinician's preference. Notification system uses Center's mappings.

Use case 8: Asynchronous communication to doctors
-Alert to pathologist located in Pittsburgh that images of pathological specimens obtained in Palermo, Italy are electronically available for rapid review. Client application calls notification system to page doctor.
-Nurse needs prn order for Tylenol. Uses client application similar to that in 4.

Use case 9: Commercial EMR system
Commercial EMR system collects important new patient data (e.g., nursing observation) or generates alert. Notification system provides delivery.

REQUIREMENTS

Types of data
The types of data that need to be disseminated by notification include text reports, tables, alerts, new orders, bedside monitor alarms, voice messages, images and pointers (e.g., hotlinks to URLs).

Communication channels
Support for current and future channels including fax, network printers, email, alphanumeric pagers, and subroutines within clinical applications such as "to-do lists". Ability to detect and handle failure of one channel or device by rerouting.

Guaranteed delivery with acknowledgment
Most applications in medicine require guaranteed delivery. Notification system must track notifications, process acknowledgements from those channels that can provide acknowledgements, and return status to calling application.

Escalation algorithms
By law, critical laboratory results must be delivered promptly to a responsible clinician. Escalation algorithms attempt serially to obtain acknowledgement from a sequence of recipients. These algorithms must be fail-safe, meaning that unless positive acknowledgement is received, the message will be delivered to a responsible party who is always available (e.g., hospital paging operator).

Multiple message versions
Display capabilities vary among communication channels (e.g., 20-character page width in pager versus 80 characters in email). Ability to specify and syntax check multiple versions of messages.

Message must be structured such that notification algorithms can modify the message (e.g., add an explanation as to why message is being sent).

Performance/reliability
Architecture must be scalable and fault tolerant.

Notification algorithms
Each use case described above requires different mappings. In general, the mappings may be complex because they involve business and workflow consideration. If a single development group had to maintain all notification algorithms, it would be an extreme bottleneck. Thus, ability to register and connect an external service is required.

Access/Security
End-to-end encryption or encryption from notifier to recipient is usually necessary.

In summary, the main requirements are support for multiple channels (and multiple message versions), acknowledgement, escalation with fail-safe, mappings provided by other parties, and security.

MODEL (Fig. 2)

Notification generating clients are modules in laboratory, radiology, and event monitoring systems that create messages for clinicians. They call the notifier (through the ORB) with message and delivery instructions (described below) and receive status messages and a list of recipients.

HL7 router/message generator for clients that can generate HL7 messages, but cannot call the ORB, we interpose a message generator that receives HL7 messages and creates calls to notifier.

Notifier receives calls for notification, checks message syntax and delivery instruction syntax and returns error message on failure. Otherwise, processes the delivery instructions which typically leads to a call to a notification algorithm specified by the delivery instructions. Notifier then processes a set of <message, channel, address> triples by calling the appropriate channel. Notifier receives status messages from channels and returns recipients and results to clients. Notifier logs messages.

Notification algorithms implement the mappings required by different types of information. A notification algorithm is called with the message and it returns a set of <message, channel, address> triples and a status code. Notification algorithms typically use databases and procedures to represent mappings.

Communication channels
Each communication channel is an encapsulation typically of existing software such as Unix sendmail.

Client application for Dr. J's assistant is a special case communication channel that is also a client that generates calls to the notification system.
Parameters:
Delivery instructions can either specify the individual and channel directly (i.e., specify a persons unique ID and the name of a channel), or specify the name of a procedure to invoke to determine either or both of these parameters.

Messages: Messages contain sections corresponding to each communication channel. For example, the pager section has a terse pager subject, a message that is terse with 20-characters-wide lines.

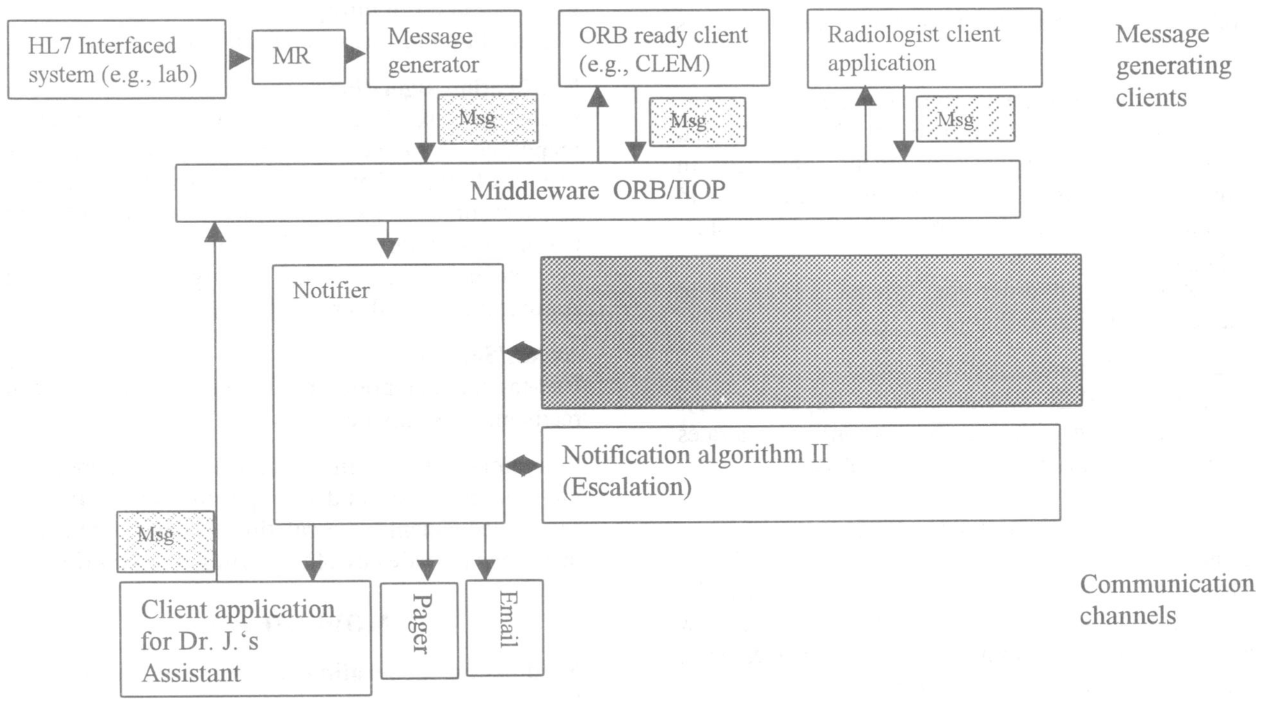
HL7 Interfaced system (e.g., lab)
MR
Message generator
ORB ready client (e.g., CLEM)
Radiologist client application
Message generating clients
Msg
Msg
Msg
Middleware ORB/IIOP
Notifier
Notification algorithm II (Escalation)
Msg
Client application for Dr. J.'s Assistant
Pager
Email
Communication channels

Fig., 2 Notification System. Legend: MR, HL7 message router; CLEM, clinical event monitor; Msg, message; ORB, object request broker.

IMPLEMENTATION

Our current notifier is written in Java and is architected as a CORBA object. It runs on an ES10000 Unix system and uses Oracle 7. We use Visigenics Visibroker ORB.

We have deployed two notification algorithms. Notification algorithm I maps from patient ID to primary doctor and resident, and depending on time of day, potentially to cross cover. Map from recipient to communication channel is by message type and set individually by user.

The second notification algorithm is fail safe delivery. We defined a syntax to describe a sequence of recipients and channels, and this algorithm processes such specifications.

To date, we have not added externally developed mappings although we would treat them similarly to our internally developed mappings.

There are two communication channels with message formats for each. For e-mailing, we encapsulate the Unix program "sendmail" to generate Internet Simple Mail Transfer Protocol messages.
For paging, we encapsulate the SkyTel® software development kit and use the V1 encryption server.

At present, the notifier returns the following error codes: No error, Invalid medical record number syntax, unknown medical record number, Invalid delivery direction format, Invalid doctor id, Invalid notification algorithm name, Invalid message, and unknown doctor id in coverage list.

In addition to notification, notifier provides functions that allow access to data within selected notification algorithms. For example, the radiology client application can call Notifier with a patient ID to ask for a digest of clinicians associated with that patient with their roles, and contact information.

DISCUSSION

We have developed a general purpose notification system. Key features of the model are:

- Support of multiple channels and corresponding message formats
- Concept of externally developed mappings
- Escalation with fail safe delivery

Our approach is similar to that of Geissbühler, the only other published description of a general notification system in medicine, in its use of multiple notification algorithms, which we view as separate servers not necessarily maintained by us. Our approach is more general in that it deals explicitly with the need for multiple message formats when multiple channels may be used for a message.

Although the potential benefit of the publish-subscribe method is that the publishers don't have to worry about delivery, it is not at all clear how security and access control can be implemented if any application can subscribe to any data.

Although our model is based on significant experience and analysis of use cases, the status of the implementation is limited. Therefore, the model should be viewed as perhaps the most general model available, but not a reference model for the notifier component in a standard architecture.

One might wonder whether progress in computing will simplify or eliminate the need for a general notification system. In particular, if we imagine a paperless environment and "personal communicators" for voice and data, then there would be just one communication channel for each user instead of fax machines, printers, pagers etc. An application needing to communicate with a clinician would simply send a message to the inbox of that device. The problem with this view is that a simple inbox would be overwhelmed and that there would be an immediate need to create microchannels within that inbox (e.g., stat folders, whenever folders) with different levels of latency and invasiveness for the user. The problem of mapping into these microchannels would be in all key respects the same notification problem. The only design issue would be whether the mapping should reside on the device or in the server layer.

Our notification model, which was designed for asynchronous computer-person notification can also support the simpler requirements of inter-person communication (synchronous or asynchronous) identified by Coiera, which basically require directory services that are role-based (e.g., I want to send a message to the infectious disease consultant).

CONCLUSION

Notification systems are just beginning to be recognized as principle components in health care computing. The design presented here provides extensive functionality. With addition of telephony and voice mail, we expect it to satisfy most of the communication requirements of the enterprise.

ACKNOWLEDGMENTS

Grants 5 R29 LM06233, 5 GO8 LM06625-01, and T15 LM/DE07059 from the National Library of Medicine supported this work.

REFERENCES

1. Coiera E. Clinical communication: A new informatics paradigm. JAMIA-Symposium Supplement 1996:17-22.
2. Geissbuhler A, Grande J, Bates R, Miller R, Stead W. Design of a general clinical notification system based on the publish-subscribe paradigm. JAMIA- symposium supplement 1997:126-30.
3. Wagner M, Pankaskie M, Hogan W, et al. Clinical event monitoring at the University of Pittsburgh. JAMIA- symposium supplement 1997:188-92.
4. Eisenstadt SA, Wagner MM, Hogan WR, Pankaskie MC, Tsui F-C, Wilbright W. Mobile workers in healthcare and their information needs: Are 2-way pagers the answer? JAMIA-Symposium Supplement 1998:135-40.
5. Kuperman G, Teich J, Bates D, et al. Detecting alerts, notifying the physician, and offering action items: A comprehensive alerting system. JAMIA-symposium supplement 1996:704-9.