Alerts in Mobile Healthcare Applications: Requirements and Pilot Study

Eleanna Kafeza, Dickson K. W. Chiu, *Senior Member, IEEE*, S. C. Cheung, *Member, IEEE*, and Marina Kafeza

***Abstract*—Recent advances in mobile technologies have greatly extended traditional communication technologies to mobile devices. At the same time, healthcare environments are by nature "mobile" where doctors and nurses do not have fixed workspaces. Irregular and exceptional events are generated in daily hospital routines, such as operations rescheduling, laboratory/examination results, and adverse drug events. These events may create requests that should be delivered to the appropriate person at the appropriate time. Those requests that are classified as urgent are referred to as *alerts*. Efficient routing and monitoring of alerts are keys to quality and cost-effective healthcare services. Presently, these are generally handled in an ad hoc manner. In this paper, we propose the use of a healthcare alert management system to handle these alert messages systematically. We develop a model for specifying alerts that are associated with medical tasks and a set of parameters for their routing. We design an alert monitor that matches medical staff and their mobile devices to receive alerts, based on the requirements of these alerts. We also propose a mechanism to handle and reroute, if necessary, an alert message when it has not been acknowledged within a specific deadline.**

***Index Terms*—Alerts, capability, exception handling, healthcare alert monitoring system, mobile devices, monitor, requirements engineering, role.**

I. Introduction

In DAILY routines, physicians, nurses, and other staff of a hospital have to be reached and be updated of new incidents and information while they are commuting in their working environment. Tasks like medication monitoring, emergency hospitalization of patients, laboratory examination results, shipment of drugs, exchange of information among physicians, etc., produce a large number of messages. Those requests that are classified as urgent are referred to as *alerts*. Alerts have a broader coverage than *alarms*, which refer only to *critical events*. Most medical alarms have to be handled within a time period. For example, a patient who has tachycardia (which is a medical alarm) should be examined by the physician and be given treatment within a specific period of time. Thus, an alert is created to request such examination. In addition, there are other types of alerts that are not alarms involved in daily healthcare operations. For example, cancellation of an appointment can trigger a notification alert to the corresponding physician, or a meeting can trigger an alert to notify the involved parties, and so on.

Although there are alerts that are informative in nature and need not be acknowledged (for example, a physician may receive the hospital newsletter every week), most alerts have to be handled within a time period. The existing practice tends to use cellular phones and pagers for communications. This is not adequate for seamless integration with existing and future healthcare information systems. The use of personal digital assistants (PDAs) for ubiquitous computing are getting popular, but mainly just for storing addresses, scheduling, and organizing tasks. With recent advances in mobile technologies, PDAs and portable personal computers (PCs) have also been used for Internet accessibility. In addition, new mobile "smart devices" featured with different software and hardware capabilities are kept introduced into the market. When these mobile devices [14] are used in a healthcare environment, they are called *mobile healthcare computing devices* (MHCDs). These MHCDs now become part of daily life, thus making ubiquitous computing a possibility for healthcare environments as well. Awareness, accessibility, and responsiveness are the key relationships between clients and organizations in the world of ubiquitous computing [8]. In particular, healthcare applications must respond actively and very timely to patients' needs, which can be life critical.

There are several issues to be considered with this problem. An alert model should include various alert types and parameters that qualify the person to receive an alert, according to the requirement of a medical event. Apart from medical qualifications, the availability of medical staff members and their reachable devices should also be considered. In addition, functionalities such as routing, monitoring, and logging the alerts are also mandatory, in order to shift the burden of these communications from manual work to an automated system. To take advantage of the anyplace and anytime characteristics of mobile computing environment, we propose a healthcare alert management system (HAMS) to address this problem. We further propose a mechanism to effectively convey these alert messages to the right person(s) at the right time through the right device(s), thus minimizing delays and providing a monitoring system for assuring service quality.

Raghupathi *et al.* [17] point out that information technology (IT) is important to healthcare and the estimated expenditure of IT on healthcare in 2002 was 21.6 billion in the U.S. New healthcare applications supporting IT-based strategy are

Manuscript received February 25, 2003; revised September 25, 2003. This work was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China, under Project HKUST6170/03E.

E. Kafeza is with the Department of Marketing and Communications, Athens University of Economics and Business, Athens 10434, Greece (e-mail: kafeza@aueb.gr).

D. K. W. Chiu is with the Dickson Computer Systems, Homantin, Kowloon, Hong Kong (e-mail: dicksonchiu@ieee.org).

S. C. Cheung is with the Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong (e-mail: sccheung@cs.ust.hk).

M. Kafeza is with the Department of Vascular Surgery, University Hospital of Heraklion, Crete 71110, Greece (e-mail: mkafeza@yahoo.com).

Digital Object Identifier 10.1109/TITB.2004.828888

required for meeting competitive challenges. Ammenwerth *et al.* [1] also report that one of the major issues that mobile technologies can help in hospitals is communication and reachability management. These communications should include the information of the patient, the message sender, and the urgency. Hripcsak *et al.* [11] preliminarily identify the need for event monitors, and describe some of the requirements of such monitors, such as tracking medical events, looking for clinically important situations, and sending messages to the providers. Eisenstadt *et al.* [7] further categorize messages as *alerts*, *results*, and *replies*. Alerts can be caused by drugs and blood elements, drugs and renal functions, geriatric care, potassium, therapeutic drug monitoring, practice guidelines. Results can be radiology impressions, drug levels, lab values, and so on. The limitation of this approach is that they only focus on alerts that can be handled by two-way pagers. Ride *et al.* [18] argue that the problem of figuring out to whom the message should be sent is a difficult one. They only suggested some ad hoc solutions, for example, by sending a message to whoever has recently examined the electronic patient record.

Although various attempts have been made to handle exceptions in workflow process [3], [9], the issues of urgency in exception events have not been systematically studied. In the context of workflow management systems (WFMSs), we have recently proposed to separate user alerts from user sessions with the WFMS to improve the flexibility in our mobile e-commerce advanced object modeling environment (ME-ADOME) system [2]. Online users are alerted through ICQ (I Seek You) [19], with the task summary and universal resources locator for reply as the message content. If the user is not online or does not reply within a predefined period, the WFMS will send the alert by e-mail. At the same time, another alert may be sent via short message service (SMS) to the user's mobile phone. Whatever the alert channel has been, the user need not connect to WFMS on the same device, or even on the same platform. For example, after receiving an SMS alert, the user may use the handset to connect to the WFMS via wireless application protocol (WAP), or reply with an SMS message. Alternatively, the user may find a PC with Internet connection or use a PDA to connect to the WFMS. To our knowledge, there are no other WFMSs employing this approach. Motivated by this approach, we factor out and extend our alert modeling, in particular, for handling urgency requirements.

The contributions of this paper are as follows: 1) development of a conceptual model for specifying alerts based on the requirements of the associated medical tasks and a set of routing parameters; 2) a practical architecture for the HAMS; 3) an algorithm for matching devices to alert requirements; and 4) a mechanism for (re)routing alerts and increasing their urgency when alerts are not acknowledged within deadline. The rest of our paper is organized as follows. Section II discusses a motivating example drawn from a surgery process to demonstrate our approach throughout this paper. Section III presents an overview of our methodology and the architecture of our HAMS. Section IV describes our alert concept model that consists of two parts, namely, the process and the alerts. In Section V, we present the mechanisms for monitoring and routing the alerts, together with a system integration framework. We discuss the conclusion in Section VI with our future work.

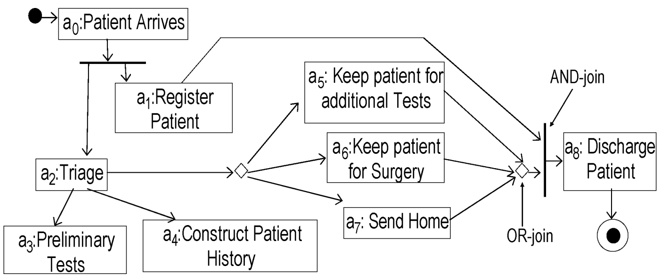


Fig. 1. Emergency room process in UML activity diagram.

II. Medical Process Example

A medical process abstracts the procedures in a hospital. Usually, each process consists of a set of *tasks*. In the trivial case, a process may consist of only one task. Each task is associated with a *role* to be played by some staff members. Fig. 1 depicts an example of an emergency room process (adapted from [10]) in unified modeling language (UML [16]) *activity diagram*.

The emergency room process consists of several tasks. Though some of them are clinical ones while the others are administrative ones, we are mainly concerned with the urgency requirements involved in the management of related personnel for effective enactment of these tasks. Each task has a unique identification, a description, and a set of roles that are capable of executing the task. For example, task a_4 is "construct patient history" and the role capable for executing this task is the *physician on duty*. The physician who is assigned to be responsible for carrying out a_4 will receive an alert message. The message could be made through a pager, a mobile phone, e-mail, and so on, as determined to be the most appropriate by the HAMS. If the person acknowledges this task assignment, the task is added to the person's worklist. However, if the person does not acknowledge within the deadline, the HAMS will take appropriate alternative actions, such as resending the alert to the same or different person, as detailed in later sections.

In the rest of the paper, we demonstrate our approach with the example from task a_6, "keep patient for surgery." Fig. 2 further expands this task and describes a typical surgery process in a UML activity diagram. When a patient is scheduled for surgery after being admitted to the ward, he/she has to go through a radiology and uroscopy test. Then a surgery time is assigned. Before the surgery, the head nurse initiates the procedures. She calls the ward, requesting the ward officer to prepare the patient. In the fetch task, the nurse calls the hospital porter to fetch the patient. The anesthesiologist is paged and reminded to go to the preparation room at a specific time. If he does not show up, the nurse has to page him again requesting his presence immediately. After the anesthesia, the patient enters the operating room with the surgeon and the assisting nurses. If the surgeon does not show up, the nurse has to send the patient an urgent message, asking for his immediate presence.

At any time, there are usually several surgeries scheduled. The time schedule for an operation might not be fixed until the last minute. In the case where patients do not show up or some

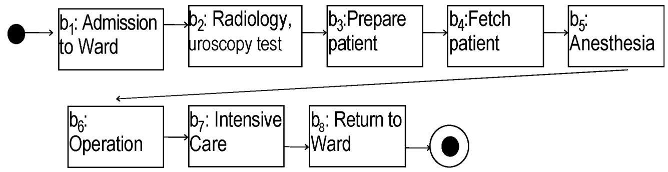


Fig. 2. Surgery process.

urgency cases have occurred, the schedule is revised, canceling or postponing the prescheduled ones. However, any delays in operations are crucial since hospital resources are usually few and expensive. Therefore, the medical staff member involved in the operation room should be informed urgently about their duties in very short notices.

Nevertheless, ensuring timely execution of these processes is critical to the provision of quality healthcare services. For example, if the radiology is not ready before the operation, the hospitalization of the patient will be prolonged, resulting in a substantial increase of the overall time cost of the operation for the hospital. This is the reason why in real life practice the head nurse often makes a phone call to the lab and reserves the time slot in advance. All these procedures could be significantly streamlined if they are automated and monitored by an HAMS.

III. Methodology and System Overview

Motivated by the general lack of alert management systems at hospitals in Greece and Hong Kong, we start off our study by gathering the objective and requirements of the physicians. Today, the progress in the medical field has resulted in the hyper-specialization of the doctors, the introduction of new and advanced types of examinations and processes, and the increasing request of the patients for better quality of medical care. At the same time, recent advances in IT are being deployed to facilitate this new complicated healthcare environment. One of the most prominent objectives within the modern hospital is the need for accurate, safe, and continuous communications among departments and highly specialized medical staff. There has been a great demand among the physicians for an alert management system that is *robust*, *efficient*, *cost effective*, *simple* and *user friendly*. Such a system will improve the communications within the hospital and outside, between the medical staff, and if possible among the medical staff and the patients as well.

Based on these objectives, detailed requirements were elicited and formulated into an alert conceptual model as presented in the next section. Then we sketched overall system architecture for an HAMS and worked out the detailed mechanisms for each components of the system. In the design, we also have to pay attention to flexibility so that alert management policies could be adapted to other hospitals, departments, and situations (such as emergency situation of major accidents or epidemic outbreak). According to these designs, we built a prototype to demonstrate the functions to the medical staff for evaluation.

As for deployment, we plan to split it into phases. The first phase is to establish a computerized hospital call center to manage all the alerts of the staff, replacing the current manual system. After getting used to the new arrangements and fine tuning of the alert management policies, the second phase is to integrate the HAMS with existing hospital information systems, such as surgery and nursing scheduling programs (if any). The third phase, if necessary, is to interoperate with other hospitals, physician's clinics, suppliers, other healthcare partners, and related government authorities.

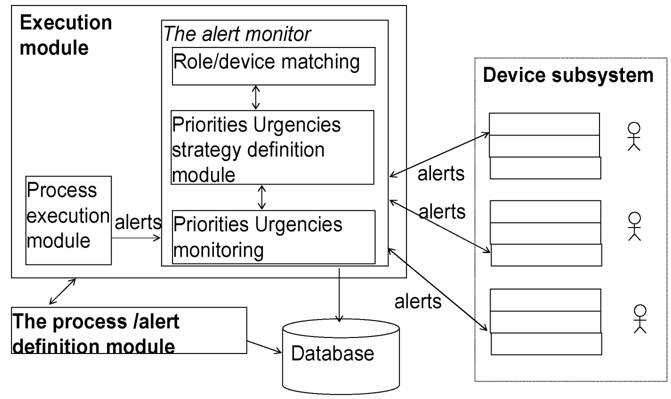


Fig. 3. HAMS architecture.

Before proceeding to our design details, let us first present an overview of the system architecture, as depicted in Fig. 3. The HAMS consists of three major parts: the *process and alert definition subsystem*, the *execution subsystem*, and the *device subsystem*. The *process and alert definition subsystem* is a tool with which users may define the tasks and their associated alerts, according to our alert conceptual model. The *execution subsystem* is responsible for initiating, routing, and monitoring the alerts. It consists of three modules: the *urgencies strategy definition*, the *staff–device matching*, and the *urgencies monitoring* modules. The *urgencies strategy definition module* specifies the policies that will be followed if the alert is not acknowledged within the deadline. The *staff–device matching* module is responsible for assigning the first staff member and device that the alert will be forwarded to. The *urgencies monitoring module* is responsible for applying the strategies defined at the urgencies strategy definition.

The *device subsystem* is a representation of the mobile environment, in which each queue corresponds to a device that can hold one or more alert messages. A staff member may possess several devices. The queue length and the device capability are specified upon device registration to the HAMS. We now proceed to detail the mechanisms inside the execution subsystem for monitoring and routing of alerts.

IV. Alert Conceptual Model

Our *alert conceptual model* consists of two parts, the process and the alerts model. This is because besides an alert itself, we must also consider the requirement of the tasks associated with the alert. The process model describes a healthcare process that abstracts the procedures of a hospital. As an extension to existing process models such as [20], our process model abstracts information regarding roles and their schedules of persons possessing these roles. A preliminary version of our process model in the context of WFMS is available in [3]. We further incorporate device characteristics and alert routing in our research.

Definition 1: A *task* is a tuple $(t_{\rm id}, D, \{R_i\})$, where $t_{\rm id}$ is the identifier of the task, D is the description of the task, and $\{R_i\}$ is the set of roles that is required to accomplish the task.

Definition 2: *Capability Profile* is a tuple $(e_{\rm id}, \{R_j\})$, where $e_{\rm id}$ identifies a staff member who has been assigned the roles specified at $\{R_j\}$.

In Fig. 2, for example, the task (b_2, "radiology, uroscopy test," "lab technician") is scheduled and it requires a lab technician to do the tests. If we are also given the following capability profile: (Jones, "lab technician"), (Brown, "lab technician"), then either Jones or Brown can be assigned to do the task. The task–staff matching can be computed using a matching algorithm, such as that in Section IV (see [3] for a more general treatment of matching algorithms). Because a person who is assigned to a task must be contacted and informed, the system should maintain for each staff member a duty/leave schedule and a set of (mobile) devices for contact.

Definition 3: A *schedule* is a sequence of steps $\{(e_{\rm id}, A_i, T_i, \{V_i\})\}$, where $e_{\rm id}$ identifies a staff member, A_i denotes his/her availability, T_i denotes the time slot, and $\{V_i\}$ denotes the set of devices the staff member has access to within the specific timeslot.

Example values for the availability are: *fully available*, *available only for very urgent events*, *unavailable*. The device could be *mobile phone*, *fax*, *PDA*, *laptop*, *desktop computer*, *pager*, *telephone*, and so on. For example, an entry to a schedule represents that the lab technician, Jones is available for critical tasks from 7 to 11 P.M., but the service of accepting though his PDA and fax is only available from 7 to 9 P.M..

During the execution of a medical process, alerts must be sent to the medical staff member who plays the requested roles and are available, using the necessary devices. Besides identification information, an alert has an urgency level and a response flag that indicates whether it has to be acknowledged by a deadline. In addition, an alert message may carry supplementary information about a patient. This information must be supported by the target device in order to be displayable. For example, an alert may include an image file that only some devices can display. If there is a long document, the device should either be able to display it conveniently (e.g., a PC) or a summary should be sent instead (e.g., a PDA or SMS mobile phone). In [2], we describe a view-based approach to facilitate this conversion.

We further refine the concept of availability for a person at a certain time to perform a certain task to be relative to urgency of the *requested* and *assigned* tasks. For example, if a surgeon is assigned to an operation, then he/she is not available during the operation. However, a surgeon examining out-patients is available for very urgent alerts. Besides the relationship between the availability and the urgency, there is a relationship between the urgency and the devices. The more urgent an alert, the more devices and the more expensive/disruptive device could be used. For example, if a surgeon is urgently alerted for his specialty, not only his pager, but also his mobile phone will receive the alert.

Definition 4: An *alert message* is a tuple $(\alpha_{\rm id}, E, D_a, \{R_j\}, U, B, T)$, where $\alpha_{\rm id}$ is a unique identification number, E is the alert subject, D_a is the content, $\{R_j\}$ is the set of the roles required to execute the alert, U is the level of urgency of the alert, B is a boolean variable denoting whether there is a need for a response or not, and T is a deadline.

The urgency of the alert U could be a function of time, indicating how the importance of a message changes with time. Normally when not acknowledged, the HAMS increases the alert urgency with time, as further explained in the next section. E is the alert subject, i.e., the entity associated with the alert. In most cases, E refers to the patient concerned. Values of the content field D_a could be attachments such as the patient's record, a text message, X-ray images, locations, and so on. The set of roles in the alert message represent the capability requirements of the task. The HAMS will select the most appropriate medical staff based on a matching algorithm, as discussed in Section IV.

In the example of Fig. 1, we can define the following alerts for the task a_6, "keep patient for surgery." Assume that t denotes the time at which the alert message is sent.

***Preoperation alerts*:**

($\alpha_{\rm id1}$, Patient X, "radiology exam," Central Lab officer, Normal, NA, one week before the surgery);
($\alpha_{\rm id2}$, Patient X, "uroscopy exam," Depart Lab officer, Normal, NA, five days before the surgery).

***Operation alerts*:**

($\alpha_{\rm id3}$, Patient X, "Prepare Patient," Ward Manager, Urgent, R, $t + 1$ h);
($\alpha_{\rm id4}$, Patient X, "Fetch Patient," Hospital Porter, Urgent, R, $t + 2$ h);
($\alpha_{\rm id5}$, Patient X, "Anesthesia," Anesthesiologist, Urgent, R, $t + 2.5$ h);
($\alpha_{\rm id6}$, Patient X, "Surgery," Surgeon, Urgent, R, $t+2.5$ h);
($\alpha_{\rm id7}$, Patient X, "Surgery," Intensive Care Unit Officer, Urgent, R, $t + 3$ h).

After defining the tasks and the alerts, we have to associate them, that is, for every task there is an associated set of alerts. For example, the task "admission to ward" has as an associated alert $\alpha_{\rm id1}$. The task "fetch patient" has the alerts $\{\alpha_{\rm id5}, \alpha_{\rm id6}, \alpha_{\rm id7}\}$, because when the anesthesiologist is alerted, the surgeon and the nurses have to be reminded about the operation as well.

Definition 5: An *association* is a tuple $(t_{\rm id}, \{\alpha_{\rm id}\})$, where the task with task identification $t_{\rm id}$ is associated to a set of alerts $\{\alpha_{\rm id}\}$ that have to be initiated before the task can be completed.

Some of the attributes of the alert could be derived. A derived attribute can be calculated from other attributes.

Definition 6: Given an alert message $(\{\alpha_{\rm id}\}, E, D_a, \{R_j\}, U, B, T)$, we say that an attribute say X is *derived* by a basic attribute say Y, if there is a function f such that $f(X) = Y$.

For example, if E is the patient's name, $\{R_j\}$ is the physician's role, *surgeon*, and f is an structured query language query on the patient database (for example SELECT Physician FROM PATIENT_TABLE where PatientName="E" and Role="surgeon"), then the name of the physician can be retrieved automatically and added to the alert tuple. A derived attribute denotes that this information can be extracted from the existing system and the associated function implements the extraction. Thus, derived attributes can facilitate the assignment of values in a diverse set of environments automatically.

So far, we have dealt with the problem of defining alerts in order to communicate a message to the right person at the right time. Alerts are parts of daily medical routine, and thus, are associated to medical tasks. The next important issue is to consider the acknowledgment of an alert. The HAMS needs to deal with alerts that are not acknowledged by the deadline. Ride *et al.* [18] have implemented an alert system mainly for adverse drug events. Upon occurrence of the event, an alert is sent to all physicians who have accessed the patient's record within the three days preceding the event and to the patient's attending physician. If the alert is not marked "taken care of," it is also sent electronically to any additional physicians who have accessed the patient's record within three days after the event. In our model, we propose a more general approach in which a strategy can be defined by the administrator.

Definition 7: A *response* to an alert is a tuple ($\alpha_{\rm id}$, ack, t), where $\alpha_{\rm id}$ is the identification number of the alert, ack is the response (such as, a telephone call, a fax, an e-mail) that results in a task assignment, and t is a timestamp when the HAMS receives the response.

From reported practice, we observe that as long as an alert is not acknowledged, the medical staff assigned to the alert may change. As a result, the HAMS can revise the matching between the alert and the medical staff. Extending this paradigm to mobile technologies, as long as the alert is not responded, the HAMS could use other device(s) to reach these medical staff as well. We further propose that given an alert, as the time passes without any acknowledgment, the urgency of the alert should increase as well. Thus, this will in turn change the medical staff being alerted and/or the devices being used.

Definition 8: A *device* is a tuple ($d_{\rm id}$, $\{C_{di}\}$, P, Q), where $d_{\rm id}$ is a unique identification number, $\{C_{di}\}$ is the set of capabilities for the device, P is a default urgency assigned to this device, and Q is the queue in which the set of alerts are waiting to be served by the corresponding person that possesses the device.

Definition 9: Each staff member has an associated set of device queues called the *device table*.

For example, a staff member might have a telephone device queue that keeps the voice messages, a mail device queue that keeps the e-mail messages, a fax device queue that keeps the faxes sent to him, a mobile device queue, an SMS device queue, etc. All these queues constitute the *device table*.

Definition 10: A *device subsystem* is the set of all device tables of all persons.

V. HAMS Logic Design and Implementation

In this section, we first present the control flow of an alert. We then proceed to the detail logic necessary for a flexible HAMS, which supports sophisticated alert monitoring and routing, such as an example staff–device matching algorithm. We also discuss how to deal with the problem of unacknowledged alerts with a flexible urgency strategy definition schema and raising urgencies, together with an alert monitoring algorithm. Then we present an overall implementation architecture, with a user front-end prototype.

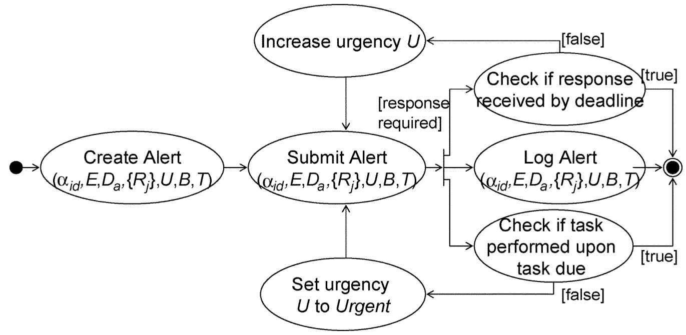


Fig. 4. Control flow of the alerts in UML activity diagram.

A. Alert Control Flow

Fig. 4 describes an overview of the control flow of alerts. The "create alert" node implements the creation of the alert as well as the value assignment (i.e., the execution of the staff–device matching module). The "submit alert" node sends the alert to the assigned personnel and updates the alert to the active alerts table that keeps track of the currently pending alerts. The "check if response received by deadline" is done by the alert monitoring algorithm as well as the "increase urgency U" node that results in the resubmission of an unresponded alert. The "check if task performed upon task due" node is done by the clinic staff physically, i.e., if the staff finds out that the personnel that responded to the alert has not performed the associated task, then it generates another alert. This step also verifies if the task has been performed by the assigned personnel. The "log alert" node keeps the logging information.

B. Staff–Device Matching Module

The staff–device matching module is responsible for assigning a staff member and a device to each alert. The staff–device matching algorithm (see Fig. 5) searches for those staff members who can play the roles required for the alert. The algorithm then selects staff members among those available according to their schedule recorded in the database of the system. This further restricts the staff members who can receive the alert. The next step is to decide upon the device, based on the alert urgency. If the matching is successful, one staff member together with an appropriate device is selected (see [12] for applying various cost functions and strategy to select one among other feasible employees). In case there is no match (i.e., there exists no available staff with the requested role), the algorithm upgrades the alert by expanding the roles. For example, if there is no available physician to treat an emergency, the alert will propagate to the consultant, and finally to the director of the clinic in the end. On the other hand, if no device can handle the requested urgency, the next available device associated with a higher urgency (see Table I) will be assigned instead.

The above entry describes the alert identified as 0011. It is associated with the patient J. Smith and a message "prepare patient" has been sent to P. Brown. It has been forwarded to an urgent device because the urgency of the alert initially is *Urgent*. A response for this message is expected to be received before

Input: alert requests
Output: (e_{id}, d_{id})
Staff/device matching
For every alert (α_{id}, E, D_a, $\{R_j\}$, U, B, T)
// find the staff playing the requested roles
Compute the set C={ e_{id}: such that $\forall(e_{id}, \{R_j\})$ $\{R_i\}=\{R_j\}$ }
// if no such staff exists, find a staff member playing a superset of roles
If $C = \varnothing${
Increase urgency
Upgrade alert (α_{id}, E, D_a, $\{R_i\}$, *Urgency for role substitution*, B, T)
exit
}
Let S= {{(e_{id}, A_i, T_i, $\{V_i\}$)}$_i$ where $e_{id} \in C$ } *// the schedules of the selected staff member*
// find the staff that are available to receive the alert before its deadline
Compute the set C_{sub}={ e_{id} : $e_{id} \in C$ and in its schedule{(e_{id}, A_i, T_i, , $\{V_i\}$)}the A_i is true for T
// if no such staff exists, find a staff member playing a superset of roles
If $C_{sub}=\varnothing${
Increase urgency
Upgrade alert (α_{id}, E, D_a, $\{R_i\}$, *Urgency for role substitution*, B, T)
exit
}
for every element e_{id} of C_{sub}{
// find the devices that the staff has at the available time-slot
// and are assigned to receive alerts at the requested urgency
Compute the set D_{eid}= {d_{id} : (d_{id},{C_{di}},P,Q) with $d_{id} \in \{V_i\}$ of the schedule of the staff and P=U and {C_{di}}=capabilities(D_a)}
If all $D_{eid}=\varnothing${
Upgrade alert (α_{id}, E, D_a, $\{R_j\}$, *Urgency for role substitution*, B, T)
exit
}
}
Select the first device of the first D_{eid} set and set the staff to e_{id}

Fig. 5. Device–role matching algorithm.

TABLE I
PROCESS TASK, CAPABILITY PROFILE, AND SCHEDULE EXAMPLE

Task		
Task id	*Description*	*Role*
1	Admission to Ward	Ward Manager
3	Test	Lab technician
4	Prepare patient for operation	Ward Officer
5	Anaesthesia in operation X	Assigned Anaesthesiologist for operation X

Capability Profile	
Staff	*Role*
Jones	Lab technician
Brown	Lab technician
O'Ttool	Ward Officer
Smith	Anaesthesiologist,

Schedule			
Staff	*Availability*	*Timeslot*	*Devices*
Jones	Fully available	9.00am-12.00am	Desktop computer, telephone
	Available for emergencies	1.00am-5.00pm	Mobile
	Available for critical tasks	7.00pm-9.00pm	PDA, mobile, fax, pager
	Available for critical tasks	9.00pm-11.00pm	Mobile, pager
	default	anytime	Pager
Brown	Fully available	9.00am-5.00am	Desktop computer

2/5: 17:00. The device to receive the alert is identified as 00A, which can accept SMS messages. Till the current time, there is no response. This entry is generated after the execution of the matching algorithm.

After the matching, the *active alerts table* keeps all instantiated alerts, their corresponding devices, and whether the alert has been acknowledged or not. A typical entry of the active alerts table is shown as follows:

TABLE II
ACTIVE ALERTS TABLE

Active alerts table : *Alert*	*device*	*Resp.*
(0011, J. Smith, "Prepare patient", P. Brown, U(t_0)=Normal, true, 2/5: 17:00)	(00A,text,Urgent,SMS)	∅

Alert: (0011, J. Smith, "Prepare patient", P. Brown, $U(t_0) = \text{Normal}$, true, 2/5: 17:00), Device: (00A,text,Urgent,SMS), Response: non.

Urgencies Strategy Definition Module: The urgencies strategy definition module is a tool for defining the policies according to which the urgencies of the alert will evolve. Moreover, this module is responsible for keeping and updating status information for the alerts. In our alert model, we associate every alert with an urgency value. With every device, we also associate a minimum urgency that the specific device can serve to alert.

During the specification phase, the administrator has to specify the *urgencies strategy tables*. If an alert is not acknowledged, the HAMS raises its urgency. An *urgencies strategy table* defines the policies for every urgency increase and the additional actions that should be taken. The administrator may define different urgency strategy tables, so that they can be associated with different types of alerts. For example, we could define the urgency values from the ordered set {Low, Normal, Urgent, Very Urgent, Critical, Very Critical} and a default *urgency function* as follows:

$$U(t) = \begin{cases} \text{Urgent}, & t \le T \\ \text{Very Urgent}, & T < t \le T + dt1 \\ \text{Critical}, & T + dt1 < t \le T + dt1 + dt2 \\ \text{Very Critical}, & T + dt1 + dt2 < t \\ & \quad \le T + dt1 + dt2 + dt3. \end{cases}$$

Table I shows an example urgencies strategy table. Here, let us consider the association of alert 0011 of Table II with this table. Initially an e-mail is sent to a specific staff member. If he does not acknowledge by the deadline, a call is made to his mobile and a fax is sent to him as well. If there is still no response at time $T + d_{t1}$, a home call is made and the alert is also forwarded to another available staff member. In the end, if there is still no response, the alert is forwarded to other staff members with a superset of roles.

C. Staff and Device Monitoring Module

The staff and device monitoring module is responsible for the sending of the alert message, receiving of the acknowledgments, maintenance of alert status, and logging of information.

For every acknowledgment message received, the staff and device monitoring algorithm (cf. Fig. 6) updates the status information of the associated alert. It removes the alert from the active table and updates the log file, since the alert has been "taken care of." If the alert message has been sent to several staff members, the first one to acknowledge is assigned to the task and then reconfirmed. Other staffs will receive a cancellation message instead. Only upon reconfirmation should the staff start carrying out the task. This can avoid situations like two physicians traveling to attend the same patient.

Begin
  For every (α_{id},*ack*,*t*), DELETE α_{id} entry form Active table and INSERT it to
    Log table,
    Send confirmation/cancellation alerts.
  For every entry of the Active table [(α_{id},*E*,*D*$_a$, *Staff*, *Ucurrent, B,T*),
  (d_{id},Dev.Cap,*P*,Q),∅] such that CurrentTime > *T* do:
    Ucurrent =U(CurrentTime)
    UrgenciesActionTable.Action(Ucurrent);
    Update alert (Ucurrent, T+dt, NewStaff, Newdid);
End

Fig. 6. Alert monitoring algorithm.

TABLE III
EXAMPLE URGENCIES STRATEGY TABLE

Urgency	*Action*
Normal	Email
Urgent	+mobile +fax
Very Urgent	+home telephone +staff substitution: send to another staff member who can be assigned the same roles.
Critical	Multiple-staff substitution: send to all staff who can be assigned the same roles.
Very Critical	Role Substitution: send to all staff with a superset of roles.

Then for every alert of the active table with its deadline expired, the algorithm checks the *urgencies strategy table* (cf. Table III), executes the associated action, and updates the status information. In the following algorithm, the procedure UrgenciesActionTable. Action(Ucurrent) executes the action that is associated to urgency Ucurrent (as defined on the urgencies strategy table). This procedure updates the entry of the active table, indicating that the urgency is increased, the deadline is adjusted, other staff member(s) will receive the alert (if applicable), and a new device(s) will receive the alert as well.

This alert monitoring algorithm has a lot of flexibility as defined through the actions (functions) specified in the urgency strategy table, such as the number of staff members to be alerted in parallel. Further customizations can be implemented. For example, consider if a message with an image is to be propagated to an SMS device. Hence, the capabilities of the device and the message do not match. An alternative could be to send an SMS instructing the staff member to read an e-mail with the image as attachment. In the definition of the device capabilities, there could be functions for translating a long document to a summary for SMS transmission. Alternatively, if bandwidth is also concerned, we can also implement such an alternative in the message.

Finally, it is also possible that a person has received and acknowledged an alert but in the end the person does not carry out the promised task. Then, the HAMS or other persons that identify such inconsistency can create a new alert with appropriate urgency (usually higher that before) to deal with this situation.

D. Prototype and Integration Issues

We have built a prototype on the Java and Oracle platforms for portability and scalability [13]. Fig. 7 presents a prototype user interface for the proposed HAMS. The interface is made up of two lists: the pending alerts list and the worklist. The pending alerts list contains those alerts that have not been acknowledged or confirmed. Pending alert items are ordered by their urgency.

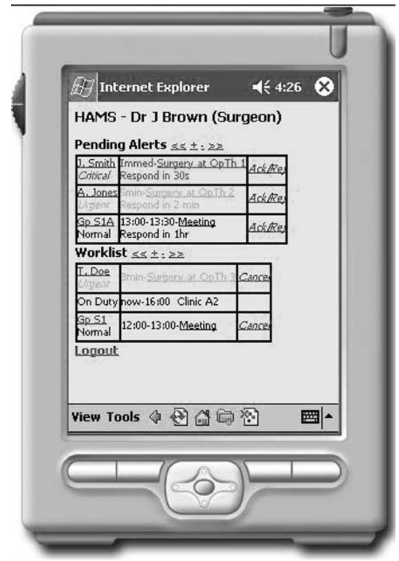


Fig. 7. Prototype user interface supporting the HAMS.

The worklist contains the tasks that remain to be carried out by the user.

Since the HAMS can manage alerts from different sources, it can also be used as middleware for integration with other existing hospital information systems in the second phase of deployment. While other systems feed alerts into the HAMS, the HAMS can also trigger resultant actions by the process execution module to carry out timely and appropriate actions in response to the alerts. For legacy systems, wrappers can be built around them to encapsulate the required functions and interfaces. This helps maintain existing systems as well as speed up implementation and deployment. Another advantage is that users need not reenter information already existing somewhere else, thereby increasing user acceptance.

In the third phase, these functions and interface specifications can be implemented with web services technology (for example, the Java Web Services Development Pack) [6] for interoperations with other organizations, such as other hospitals and medical partners. To maintain maximum reusability of code, all external access is performed through the web service interface. This approach is further justified because the extended markup language (XML) messages returned with the web services can immediately be rendered with XML stylesheet language technologies for users at different platforms. As such, access to existing hospital information systems from mobile devices can also be facilitated. For example, different hypertext markup language outputs are generated for web browsers on desktop PCs and PDAs, respectively, while WAP markup language outputs are generated for mobile phones [14]. In addition, this approach maintains maximum code modularity and reusability.

VI. DISCUSSIONS AND CONCLUSIONS

Based on the prototype and system descriptions, we have discussed with physicians in Pepagni University Hospital of Iraklion to have their evaluation. A summary is as follows. In the existing practice, alerts are communicated mainly through phones and pagers. In today's hospitals, physicians do not work at their desks, but they are called to perform many different tasks in multiple departments. Trying to reach a physician on the telephone is not always feasible. Another drawback with phone communications is that the right alerts are not always forwarded to the right persons. There are cases in which parts of the information are lost or imprecise. The introduction of the HAMS offers three important advantages. First, it will make sure that an alert can reach the person who has to be notified. Second, the implementation of an urgency policy that uses concurrently multiple devices to communicate the alert can increase the probability to inform the person on time. Third, an automated alert can make sure that the information is passed accurately and completely.

The proposed system that helps improve existing scheduling techniques is very important since resources like operation theaters are few and very expensive. In an emergency situation, timely scheduling and arrangements are crucial for the patient's outcome. The HAMS that can improve the scheduling of the operations by eliminating wasted time (for example, by efficiently searching and notifying the medical staff involved, or by making sure that all the preoperational examination have been completed) will be very beneficial for saving a hospital's resources.

An additional contribution of the HAMS is that it facilitates the division of tasks. At the specification phase, the roles and the tasks can be clearly allocated beforehand. It is important that these allocations are recorded in digital forms systematically. In this way, only a few people need to be involved in the assignment of tasks and the chief resident can easily monitor and modify the current assignment and alert policy. In addition, every modification can be easily implemented through the HAMS. Moreover, the HAMS offers a log file where all the alerts and the responses are recorded, it is easy to recreate and examine the history of events. This enables the hospital to assess and adjust the procedures by identifying the inefficiencies.

One of the system features that the physicians appreciated is that it offers the capability to choose the kind of received information, reception devices, and desired time slots. For example, the chief resident receives reports for almost everything during the day but only selected urgent messages when he/she is not on duty. The rest of the residents would prefer to receive the alerts concerning their everyday tasks. The director of a department could be informed about every alert but not through every device, except for top urgencies. In every day practice, many emergency situations do not take place during "office hours." In case of an emergency operation, the necessary medical staff not on duty must be called immediately and the HAMS improves the existing practice.

There are, however, several concerns about the proposed system. Some physicians may worry that the system can be very stressing, and be afraid of receiving messages all the time if it is inappropriately configured. It is clear that the application of such a system requires the acceptance as well as the will to learn and use it by all the departments and all the staff. Generally, a major concern is how such change could be deployed and accepted. The HAMS should be fine tuned for various specific needs, and that takes time. As such, it would be necessary to keep the existing hospital information systems for a while after the introduction of the HAMS. The proposed methodology of deployment with a phased approach (cf. Section III) is vital.

In this paper, we looked into the problem of conveying alerts to the right persons at the right time using mobile devices effectively and efficiently in a healthcare environment. We have proposed a framework of an HAMS. This framework introduces a flexible alert conceptual model that allows users to specify tasks, alerts, roles, devices, and their interrelations. We have also presented our HAMS architecture and described the alert monitoring and routing mechanisms involved. The problem of finding the right person to send an alert is nontrivial and application dependent, but our approach provides administrators with a guideline to define such strategies to address this problem systematically. We have further proposed a matching algorithm based on our alert conceptual model that can be customized based on the application requirements. Moreover, we present an alert monitoring algorithm that monitors the alerts and raises their urgency accordingly while at the same time executes the action that corresponds to the according urgency level. As such, confirmed with the evaluations of physicians, an effective HAMS can be built and fine tuned to the requirements of different hospitals and departments at different times.

For ongoing and future research, we are also investigating in healthcare information integration by means of alerts instead of workflow. Besides using the HAMS in a hospital environment, we are working on the use of an alert management system as a tool for process integration within and across organization boundaries, for both medical and e-commerce environments [5]. Thus, we are also implementing an HAMS under our ME-ADOME environment [4]. On the other hand, we are investigating in the interrelations among alerts. Studies will be made on the relation of these alerts with the issues in contract enforcement. We are also interested in further issues of collaborative workforce management, especially making use of location-dependent information in assigning staff to tasks, the impact of cancellations, and other possible exceptions.

References

[1] E. Ammenwerth, A. Buchauer, B. Bludau, and R. Haux, "Mobile information and communication tools in the hospital," *Int. J. Medical Informatics*, vol. 57, no. 1, pp. 21–40, 2000.

[2] D. K. W. Chiu, S. C. Cheung, E. Kafeza, and H. F. Leung, "Three-tier view-based methodology for M-services adaptation," *IEEE Trans. Syst., Man, Cybern. A*, vol. 33, pp. 725–740, Nov. 2003.

[3] D. K. W. Chiu, Q. Li, and K. Karlapalem, "A meta modeling approach for process management system supporting exception handling," *Inform. Syst.*, vol. 24, no. 2, pp. 159–184, 1999.

[4] D. K. W. Chiu, K. Karlapalem, Q. Li, and E. Kafeza, "Process views based E-contracts in a cross-organization internet service environment," *Distributed and Parallel Databases*, vol. 12, no. 2–3, pp. 193–216, 2002.

[5] D. K. W. Chiu, B. Kwok, R. Wong, S. C. Cheung, and E. Kafeza, "Alert driven E-service management," in *Proc. HICSS37*, Big Island, HI, Jan. 2004.

[6] H. M. Deitel, P. J. Deitel, J. P. Gadzik, J. P. Gadzik, K. Lomeli, S. E. Santry, and S. Zhang, *Java Web Services for Experienced Programmers*. Englewood Cliffs, NJ: Prentice-Hall, 2003.

[7] S. Eisenstadt, M. Wagner, W. Hogan, M. Pankaskie, F.-C. Tsui, and W. Wilbright, "Mobile workers in healthcare and their information needs: are 2-way pagers the answer?," in *Proc. AMIA Annu. Fall Symp.*, 1998, pp. 135–139.

[8] A. Fano and A. Gershman, "The future of business services in the age of ubiquitous computing," *Commun. ACM*, vol. 45, no. 12, pp. 83–87, 2002.

[9] P. Grefen, J. Vonk, and P. Apers, “Global transaction support for workflow management systems: from formal specification to practical implementation,” *Very Large Database J.*, vol. 10, no. 4, pp. 316–333, Dec. 2001.

[10] I. Haimowitz, J. Farley, G. S. Fields, J. Stillman, and B. Vivier, “Temporal reasoning for automated workflow in health care enterprises,” in *Electronic Commerce: Current Reasearch Issues and Applications*. New York: Springer-Verlag, 1996, pp. 87–113.

[11] G. Hripcsak, P. Clayton, R. A. Jenders, J. J. Cimino, and S. B. Johnson, “Design of a clinical event monitor,” *Comput. Biomed. Res.*, vol. 29, pp. 194–221, 1996.

[12] P. Hung, “A capability-based activity specification and decomposition for an activity management system,” M.Phil. Thesis, Comput. Sci. Dept., Hong Kong Univ. Sci. and Technol., 1995.

[13] J. Price, *Oracle 9i JDBC Programming*. New York: McGraw-Hill, 2002.

[14] Y.-B. Lin and I. Chlamtac, *Wireless and Mobile Network Architectures*. New York: Wiley, 2000.

[15] F. Malamateniou and G. Vassilacopulos, “A workflow-based approach to virtual patient record security,” *IEEE Trans. Inform. Technol. Biomed.*, vol. 2, pp. 39–145, Sept. 1998.

[16] Unified Modeling Language (UML) Specification 1.5, Object Management Group. (2003, Mar.). [Online]. Available: http://www.omg.org/cgi-bin/doc?formal/03-03-01

[17] W. Raghupathi and J. Tan, “Strategic IT applications in health care,” *Commun. ACM*, vol. 45, no. 12, pp. 56–61, 2002.

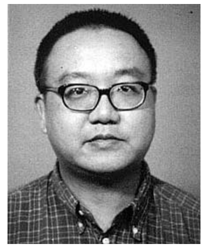
[18] D. M. Ride, C. Safran, R. S. Philips, Q. Wang, D. R. Calkins, T. L. Delbanco, H. L. Bleich, and W. V. Slack, “Effect of computer based alerts on the treatment and outcomes of hospitalized patients,” *Archives Internal Medicine*, vol. 154, pp. 1511–1517, 1994.

[19] P. Weverka. (2000) Mastering ICQ: The Official Guide. IDG Books, ICQ Press. [Online]. Available: http://www.icq.com

[20] Workflow Management Coalition. (2004) The Workflow Reference Model. [Online]. Available: http://www.wfmc.org/standards/model.htm

Eleanna Kafeza received the B.E. degree in the computer engineering and informatics from the University of Patras, Patras, Greece, in 1993, and the Ph.D. degree in computer science from the Hong Kong University of Science and Technology (HKUST), Clear Water Bay, Hong Kong, in 2000.

After graduation, she was a Visiting Assistant Lecturer at HKUST. She is now a Lecturer in the Department of Marketing and Communications at Athens University of Economics and Business, Athens, Greece. Her current research interests are in e-commerce, workflow management systems, scheduling, and temporal constraint satisfaction.



Dickson K. W. Chiu (M’89–SM’04) was born in Hong Kong in 1966. He received the B.Sc. degree in computer studies from the University of Hong Kong, Hong Kong, in 1987. He received the M.Sc. and the Ph.D. degrees in computer science from the Hong Kong University of Science and Technology, Clear Water Bay, in 1994 and 2000, respectively.

He worked as a Visiting Assistant Lecturer at after graduation at the Hong Kong University of Science and Technology. He also started his own computer company while studying part-time. From 2001 to 2003, he was an Assistant Professor at the Department of Computer Science, The Chinese University of Hong Kong. His research interest is in Information Systems Engineering for e-/m-commerce with a cross-disciplinary approach, involving Internet technologies, software engineering, agents, workflows, information system management, security, and databases. His research results have been published in more than 45 technical papers in international journals and conference proceedings.

Dr. Chiu received a best paper award in the 37th Hawaii International Conference on System Sciences, in 2004. He is a member of ACM and the Hong Kong Computer Society.

S. C. Cheung (M’03) received the B.Eng. (Hons.) degree in electrical engineering from the University of Hong Kong, Hong Kong, in 1984. He received the M.Sc. and Ph.D. degrees in computing from the Imperial College of Science, Technology and Medicine, University of London, London, U.K., in 1988 and 1994, respectively.

He is an Associate Professor of Computer Science and Associate Director of CyberSpace Center at the Hong Kong University of Science and Technology, Clear Water Bay. He has served actively on the program committees of international conferences on software engineering, distributed systems, and web technologies. His research interests include software engineering, workflows, Internet and mobile computing, multimedia, information systems, and security.

Marina Kafeza did her premedical studies at the Medical School, University of Liege, Belgium, in 1999. She completed her medical studies at the Medical School, University of Heraklion, in 2002, where she is now working toward the Ph.D. degree.

She is a Research Associate at the Department of Vascular Surgery, University Hospital of Heraklion, Crete, Greece. Her research interests focus on information systems for hospital applications and software systems for diagnostic methods on carotid atherosclerosis.