# Context-Aware Multimedia Computing in the Intelligent Hospital

Scott Mitchell, Mark D. Spiteri, John Bates, George Coulouris
Laboratory for Communications Engineering
University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, UK
{*rsm31, mds24, jb141, gfc22*}*@cam.ac.uk*

This paper describes an application middleware that addresses the requirement for immediate high-quality multimedia communications in environments where users' work practices exhibit a large degree of physical mobility. A modern hospital is one such environment, with diverse, often mission-critical, communication needs that are not addressed adequately by existing systems. By integrating a multimedia framework with an event-based notification system, we are developing **QoS DREAM**, a platform that can provide seamless, context-sensitive communications, which can adapt to users' location and even follow them around. This paper describes how we are using QoS DREAM to design what we term our concept of the "Intelligent Hospital". We also introduce a usability study that we are currently engaged in. This is allowing us to determine a realistic set of communication and information access requirements within a busy hospital environment. We conclude by illustrating a prototype of the system.

## 1. Introduction

The theses of this paper are that, (1) in a modern organisation like a hospital where staff move around while carrying out their clinical duties, there is a requirement for immediate high-quality multimedia communication sessions that do not require staff to know the present physical location of their colleagues. Furthermore, (2) staff often require an awareness of the location—and other context—of their colleagues, their patients, and flexible access to information like past and current patient records, medical references and drug databases. Rather than have to periodically check for information that might still be unavailable, they require a mechanism where they are notified of interesting occurrences when they occur, regardless of their present location. These points are validated by a study on communication issues in clinical settings documented in [Coi96].

Many of the clinical, administrative and support roles present in a hospital entail a high degree of mobility, particularly in a department such as "Accident and Emergency", where clinicians must respond to incoming cases as they occur and may need to call upon the services of outside specialist teams at any time. Much of the communication is asynchronous; e.g. waiting for notification that test results are available, or paging a colleague and waiting for him to return the call. Existing solutions, like telephones and pagers, are lacking for various reasons: busy clinicians typically cannot wait by the phone for a call, and may not be able to answer a page immediately for any number of reasons. The existing technology lacks context awareness: it is pointless and inappropriate to page a doctor for a minor consultation when they are engaged in a critical procedure.

We propose to replace existing intra-hospital communications with a system of digital audio and video streams coupled to an event-based programming mechanism for control and notification. The "Intelligent Hospital" will be wired with sensors to enable tracking the location of staff, patients and equipment. Terminals will deployed throughout the hospital providing access to audio/video conferencing and other information systems.

The Intelligent Hospital is a location- and context-aware application; that is, the system knows the location of each of its users and has some idea of what they are currently doing. This enables a user to, for example, request a call to 'Doctor Alice' and have the call connected *at her current location*, regardless of where each has moved to in the meantime. Similarly, clinicians will be able to view test results as they become available, without polling, and receive notification of adverse changes to a patient's vital signs.

Section 2 gives an overview of the QoS DREAM middleware framework. Section 3 discusses the specific requirements of the Intelligent Hospital in more detail and shows how these are addressed by QoS DREAM. Section 4 briefly examines our prototype implementation while Section 5 presents conclusions and a summary of selected related work.

## 2. The QoS DREAM framework

The Intelligent Hospital software is being built as a demonstrator application of the QoS DREAM (*Quality of Service for Dynamically Reconfigurable Adaptive Multimedia*) middleware platform. QoS DREAM integrates and extends existing technologies for reconfigurable multimedia streaming and event-based programming to provide a platform supporting context-aware, event-

driven multimedia applications. We are particularly interested in applications exhibiting a high degree of *user* mobility, but not necessarily requiring the use of mobile *devices*. The platform has four main conceptual components (Figure 1):

1. An operating system support layer, offering resource management and admission control functionality for audio and video streaming.
2. Dynamic multimedia streaming, based on the **DJINN** platform [NKMC98, MNCK99] developed at QMW College, London. The platform must be able to reliably and efficiently re-route and reconfigure streams to account to the movement of participants and changes in system load.
3. An event-based programming paradigm utilising the **HERALD** architecture from Cambridge [Spi00]. Event notification is a powerful mechanism for rapid construction of highly dynamic and reactive dynamic applications, and for integrating existing legacy systems.
4. A set of integrated APIs for building applications using the underlying multimedia and event technologies.

A key feature of the architecture is our use of a *dynamic runtime model* of the application. The model is built from hierarchically composed components, where the leaves of the component tree map directly onto the *active* components of the underlying application. The interior nodes of the tree are *composite* components; these add additional high-level behaviour to applications by providing operations to manipulate their sub-components. Each model component encapsulates a description of the reconfiguration and QoS properties of its sub-component tree and the associated active components, as well as defining the patterns of event registration and starting event sources. Figure 2 illustrates this, showing a media stream flowing between active components on two different hosts. The application model encapsulates the connectivity between these components and the rest of the system. The model controls the operation of the active components and receives notification of state changes in the active layer.

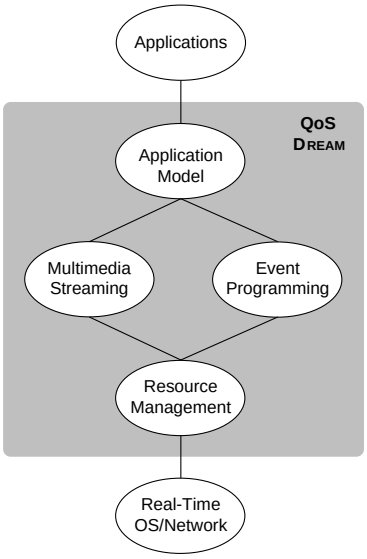


**figure 1**

Our primary motivation for the use of an application model is to separate the design of an application from its realisation at run-time. The model enables programmers to build and evolve applications at a high level of abstraction while remaining unaware of the distinction between model and active components.

## 2.1. Multimedia streaming & reconfiguration

The DJINN framework supports the construction and execution of highly dynamic, QoS-aware distributed multimedia systems.

Access to resources is controlled by a distributed admission-control mechanism. The application model specifies the resource requirements for each of its components and updates this specification whenever the application is reconfigured. A per-application admission-control agent distributes resource requests to the hosts controlling the required resources. The agent enters into a negotiation with the various resource managers to determine a suitable allocation of physical resources. The outcome of this negotiation may be influenced by the priority or importance of the application, with higher priority applications permitted to "steal" resources from their low priority companions. In the medical context this allows, for example, personal video calls to be suspended temporarily in favour of an emergency consultation.

It should be noted that the admission control mechanism does not aim to entirely eliminate the need for short-term adaptation of application performance to deal with transient changes in resource availability. Rather, it seeks to minimise the amount of adaptation required by converging quickly on a near-optimal operating point for the application, and providing some minimum guaranteed level of service.

DJINN implements a transactional reconfiguration mechanism to ensure that applications will always be left in a consistent state after a reconfiguration, and to minimise "glitches" and loss of temporal integrity that can occur during the reconfiguration of live, running media streams.

Reconfigurations are initially carried out on the model components only. All changes to the model are enclosed within an *atomic action*, a transactional construct that allows the application to be "rolled back" to its

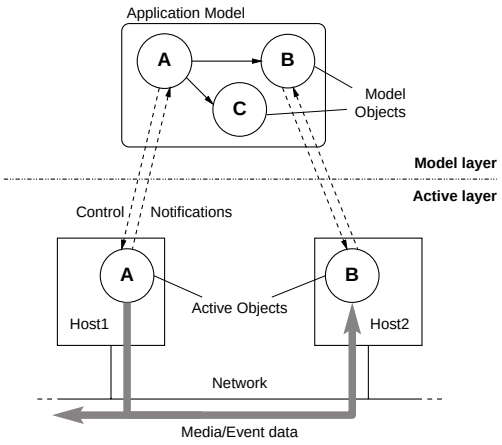


**figure 2**

previous stable state if the reconfiguration cannot be completed successfully for some reason. As with a conventional database transaction, once all of the necessary changes have been made to the model, the atomic action can be committed. The commit process checks that the new configuration of the model is *consistent* (under the application's definition of the term) and that it passes the admission-control test. Once these checks have been performed, and the necessary resources reserved, there is nothing – barring equipment failure – that will prevent the active layer of the application moving safely, and atomically, to the new configuration.

However, the usual notion of atomicity must be revised somewhat when a reconfiguration is performed on live media streams. If the ordering of updates to the active layer objects is not carefully managed, media data may be lost or corrupted *during the transition* from the initial to final configuration. In order to minimise these undesirable effects, DJINN computes a real-time schedule for the active layer updates; the schedule expresses a trade-off between the *smoothness* of the transition, the resources used and the time taken to complete the reconfiguration.

The desired smoothness of a reconfiguration is an application-defined quantity. For example, in a "follow-me video" scenario, an instantaneous handover between displays is probably only necessary if the viewer can see both displays simultaneously and move quickly between them; if the displays are in different rooms a more conservative, resource-friendly handover procedure could be used. On the other hand, if the goal of an application is to track an individual using multiple cameras, the transitions between cameras should be as seamless as possible.

### 2.2. Event notification

The QoS DREAM framework employs the HERALD event transport [Spi00] to propagate event notifications to its distributed components.

HERALD is based on the revised version of the Cambridge Event Architecture described in [BBMS98, BMBM+00], and endorses the *publish-register-notify* programming paradigm. In order to support loose coupling, HERALD uses a purely reactive interactive model, where components discover and learn about each other's capabilities and events at runtime as required. In a distributed system built using HERALD, each event-aware application unit is known as an event component, and can be an *event source* or an *event client*. Events are structured objects, and flow from event sources to consuming event clients. *Sources* can be wrappers around an application or device, and based upon the monitoring of some property or activity within that application/device, generate events pertaining to it. *Event clients* register interest in events directly with *event sources* and are sent *event notifications* when events matching their *registrations* occur at that source. In practice, the functionality represented by either component type is provided through a package of class libraries, two of which embody the functionality of an event client and an event source respectively. These classes abstract away most of the detail pertaining to event registration, filtering, event queuing, communication, fault tolerance, event persistence, reliability and security. Application writers can wrap event functionality around their application units or legacy interfaces very rapidly without concerning themselves with the underlying middleware intricacies. A hardware monitor can therefore have an event source wrapped around it that internally communicates with the proprietary device interface, but only 'exports' events to the outside world.

A component can be both a source and client of events concurrently with respect to different components. By integrating both source and client functionality, it is possible to built specialised transforming components like *federators*, *gateways*, *storage modules*, and *event brokers*. Clients register interest with event sources by submitting content-based filtering templates that are then used by a source to determine whether a client is interested in a particular occurrence of an event. Policies reflecting notification priority and delivery deadlines, frequency, expiry, and event storage can be attached to an event registration.

## 3. The Intelligent Hospital

There is a danger when developing applications based on a new and unproven platform, that the system will end up merely reflecting the features of the technology rather than the real-world requirements of the application domain. To avoid falling into this trap, we are carrying out a usability study at the Accident & Emergency (A&E) department of the Royal London

Hospital. The aim of the study is to acquire an understanding of the work practices of staff in this environment, in particular the critical factors in their communication activity. This knowledge will allow us to build an application that assists staff in carrying out their duties without introducing new intrusive technologies that impinge upon their already very busy schedules.

Most communication between staff *within* the A&E department takes place on a face-to-face basis. However, urgent clinical duties coupled with the sheer size of the department mean that it is often impossible to firstly locate a given staff member then move to their current location in order to speak with them. In addition, A&E clinicians must communicate with people outside the department, ranging from consultations with colleagues in other specialisations to answering queries from patients' relatives. These interactions are conducted through traditional telephone and pager networks. The paging system is particularly disliked, as it encourages interruption for trivial reasons and frequently results in long delays waiting for a paged individual to make contact.

### 3.1. Proposed infrastructure

In order to address the communication problems indicated by our study, we propose the installation of a high-bandwidth network dedicated to multimedia communication, with a number of flat-screen touch-sensitive terminals ubiquitously deployed throughout the hospital. The use of a dedicated network allows us to provide real-time guarantees for audio and video streams and for event notifications. These guarantees span both the operating system and the network. Attached to each of these terminals are a digital video camera, a microphone, and a speaker. Potentially, the software can be speech-driven in addition to being controlled by touch. Clinicians can walk to a terminal, authenticate themselves, and request a video call to some other individual (or the current occupant of a specific clinical role) within the hospital. They do not have to know where the person they wish to talk to is currently located, since the system is aware of the location of all members of staff. The call is put through to the terminal closest to the individual being called, who can then choose whether they want to take the call or not. If the call is answered, a videoconference is set up between the two individuals. For audio-only communication with mobile clinicians, we are considering the use of wireless headsets using short-range radio technology such as Bluetooth [Blu99].

In order to acquire and retain knowledge of the locations of people and other mobile entities we are evaluating the use of various tracking technologies. Active Badges [HH94] meet some of the requirements of a tracking system in this application and have the advantage of being a mature and stable technology; thus it is likely that these will be deployed in our initial prototype. An Active Badge system consists of a network of infrared sensors deployed around a building that can detect badges worn by people or attached to mobile equipment. The badges are small battery-powered tags that can also carry the name and picture of an individual, thus doubling as identification tags. The badges can also receive signals from the sensors, thus acting as pagers or simply as a means to attract its wearer's attention.

Unfortunately several features of the Active Badge technology limit its usefulness as a tracking system. Firstly, badge sightings give information on *containment* rather than absolute *location*; that is, a sighting only indicates that a badge is somewhere within the space visible to a given sensor, not the position of the badge within that space. Similarly, reflection and transmission of badge signals off and through walls, windows and doorways increases the space "visible" to each sensor and reduces the accuracy of sightings. Although careful positioning of sensors—and the use of multiple sensors covering the same space—can eliminate some of these effects, it is not in general possible to resolve badge sightings more precisely than the level of a single room. This is particularly unsatisfactory when the environment includes large open areas—such as hospital wards.

We are investigating a number of other candidate sensor technologies. Active Bats [HH99] provide very fine-grained 3-D position and orientation data, but the necessary sensor infrastructure remains prohibitively expensive. More practical technologies for the hospital environment include passive radio tags with sensors positioned across doorways or along artificial boundaries in larger spaces, or *proximity* detection using low-power radio systems such as Bluetooth [Blu99] or PEN [BC97].

### 3.2. Application scenarios

The following scenarios illustrate typical usage of the application we are building on top of QoS DREAM within a hospital environment. These scenarios are derived from our workplace study.

***Remote Consultation.*** Doctors in the A&E must frequently consult with their colleagues in other departments, to determine whether patients should be admitted for further, more specialised, treatment. Consider a patient who *may* need to be admitted for surgery: typically the A&E clinician will make this decision in consultation with an on-duty surgeon. Since highly trained surgeons do not sit around idly waiting

for calls from the A&E, there is a reasonable probability of the surgeon being in the operating theatre. Currently, the surgeon will be paged and messages relayed to the A&E by a nurse, assuming one is available. Surgeons have told us that they would benefit from the ability to converse directly with their A&E colleagues, even while operating on another patient. The caller would simply request a connection to the on-call surgeon; the call could then be connected directly to a speakerphone or wireless headset in the operating room. The surgeon is able to make a more *informed* decision on the treatment of the A&E patient, and to make it *quickly*. In the case of other, more mobile specialists, remote consultations might utilise video as well as audio and if necessary "follow" the doctor as she moved around the wards.

***Tracking of patients and equipment.*** While A&E staff generally have excellent awareness of the patients under their individual care, they cannot be expected to know the names, locations and status of *every* patient in the department. However, this information is often requested—by relatives calling from outside the department, by consultants from elsewhere in the hospital, or by support staff transferring the patient to another department. Our study has indicated that any system requiring manual input of patient movements would not be updated, so we are proposing the use of an automatic patient tracking system, using temporary or disposable tags issued when patients arrive in the A&E. Basic patient data would be available to any staff member from any computer terminal and perhaps replicated on wall-screens replacing the single fixed whiteboard used currently. We also plan to fit tracking tags to the various pieces of mobile equipment used in the A&E—these are frequently left in cubicles or transferred along with patients and cannot be found quickly when they are needed.

***Notification of awareness and patient data.*** Several tests are carried out on patients after admission into the A&E department. The doctor responsible for the patient then has to keep calling the laboratory to check whether the results are available. This is a waste of time from the doctor's perspective and may result in the patient being delayed unnecessarily. Using event-based notification, data (such as patient test results) can be denoted as being 'interesting' to any software entity (like the Doctor's management module), and be notified to them when it occurs. Conditions can be attached to the 'registration of interest', and the data can be forwarded to the Doctor's current location for him to retrieve on a terminal at hand. Likewise, senior nurses responsible for monitoring the deployment of human resources throughout the sections of a department can be alerted in real time when a ward becomes too busy allowing them to re-distribute staff as necessary.

## 4. Experimental prototypes

In order to verify the generic capabilities of the QoS DREAM middleware as well as the feasibility of our Intelligent Hospital application, we have deployed a prototype 'follow-me' video application (see Figure 3). This implementation demonstrated how a clinician could request a video call to another clinician without being aware of his location, have the call put through, establish an audio-video conference with variable compression/de-compression codecs according to the existing bandwidth, and have the conference smoothly reconfigure and move to follow whenever one of the clinicians changes location. The clinician being called is notified of the call by a signal sent to his Active Badge, and can acknowledge the call by clicking one of the buttons on his badge.

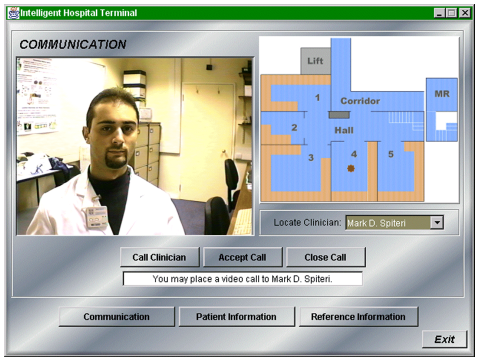


**figure 3**

We are currently implementing a second Intelligent Hospital prototype that will take into account the results of our user requirements study

The active layer of a QoS DREAM application requires an environment offering soft real-time access to system resources such as processor and network bandwidth. We are implementing the prototype platform on top of the Linux-SRT [Ing00] real-time kernel and investigating the use of a token-based Ethernet variant such as RETHER [VC95] or BEAT [Kos98] to provide bandwidth guarantees over our existing network infrastructure.

## 5. Related work

The component architecture of QoS DREAM is similar to other multimedia middleware systems such as Sumo-ORB [BS98] and CINEMA [Bar96]. CINEMA also uses a model-based approach to reconfiguration; however QoS DREAM is novel in its use of an *active* runtime model encapsulating QoS and event manage-

ment, as well as reconfiguration. In addition, the atomic action mechanism encourages the construction of reliable, yet highly dynamic, applications. Researchers at Lancaster University have been investigating mobile multimedia support for emergency services [DF98], such as equipping ambulances with videoconferencing systems.

Our approach to event notification differs from other event mechanisms like Elvin [FMK+99], Jedi [CDF98] and READY [GKP98] in that it promotes de-coupling of components through its use of reflective interfaces, while still enabling overall orchestration if required. A comprehensive registration mechanism using content-based registration templates and policies is provided.

## 6. Conclusion

The QoS DREAM multimedia framework integrates two novel technologies, and gives a distributed application developer the flexibility to build an application tailored to the needs of his environment. Based on the insight we are gaining from our usability study, we are developing the Intelligent Hospital system to validate QoS DREAM.

We acknowledge the assistance of Dr Tim Coats and the staff at the A&E Department of the Royal London Hospital. The design of the user requirements study benefited greatly from the advice and guidance of William Newman at XRCE (Cambridge). This research is funded by EPSRC, the UK Engineering and Physical Sciences Research Council.

## 7. References

[Bar96] I. Barth, "Configuring Distributed Multimedia Applications Using CINEMA", *Proc. IEEE MMSD'96*, Berlin, Germany, Mar 1996

[BBMS98] J. Bates, J. Bacon, K. Moody and M.D. Spiteri, "Using Events for the Scalable Federation of Heterogeneous Components", *Proc. ACM SIGOPS EW'98*, Sintra, Portugal, Sep 1998

[BC97] F. Bennett, D. Clarke, J. B. Evans, A. Hopper, A. Jones and D. Leask, "Piconet - Embedded Mobile Networking", *IEEE Personal Communications* 4:5, Oct 1997

[Blu99] Bluetooth Special Interest Group, "Bluetooth Specification v1.0B", Dec 1999. http://www.bluetooth.com/developer/specification/

[BMBM+00] J. Bacon, K. Moody, J. Bates, C. Ma, A. McNeil, O. Seidel and M.D. Spiteri, "Generic Support for Asynchronous, Secure Distributed Applications", *IEEE Computing*, Mar 2000

[BS98] Gordon Blair and Jean-Bernard Stefani, *Open Distributed Processing and Multimedia*, Addison-Wesley, Harlow, England, 1998.

[CDF98] G. Cugola, E. DiNitto, and A. Fugetta, "Exploiting an event-based infrastructure to develop complex distributed systems", *Proc. ICSE'98*, pages 261-270, 1998.

[Coi96] E. Coiera, *Clinical Communication – A New Informatics Paradigm*, Technical Report HPL-96-64, Hewlett-Packard Laboratories, May 1996. http://www.hpl.hp.com/techreports/96/HPL-96-64.html

[DF98] N. Davies and A. Friday, "Applications of Video in Mobile Environments", *IEEE Communications*, Jun 1998

[FMK+99] G. fitzpatrick, T. Mansfield, S. Kaplan, D. Arnold, T. Phelps, and B. Segall, "Instrumenting and augmenting the workaday world with a generic notification service called Elvin", *Proc. ECSCW'99*, Copenhagen, Denmark, Sep 1999

[GKP98] R.E. Gruber, B. Krishnamurthy, and E. Panagos, "High-level constructs in the READY notification system", *Proc. ACM SIGOPS EW'98*, Sintra, Portugal, Sep 1998

[HH94] A. Harter and A. Hopper, "A distributed location system for the Active office", *IEEE Networking*, 8(1), Jan 1994

[HH99]. A. Harter, A. Hopper, P. Steggles, A. Ward and P. Webster. "The Anatomy of a Context-Aware Application", *Proc. ACM/IEEE MOBICOM'99*, Seattle, Washington, USA, Aug 1999

[Ing00] D. Ingram, *Integrated Quality of Service Management*, Ph.D. Thesis, University of Cambridge, Jan 2000

[Kos98] R. Koster, *Design of a Real-Time Communication Service for Local Area Networks*, Master's Thesis, Department of Computer Science, University of Kaiserslautern, Germany, May 1998

[MNCK99] S. Mitchell, H. Naguib, G. Coulouris and T. Kindberg, "A QoS Support Framework for Dynamically Reconfigurable Multimedia Applications", in L. Kutvonen, H. König and M. Tienari (eds), *Distributed Applications and Interoperable Systems II*, Kluwer Academic, Boston, 1999

[NKMC98] H. Naguib, T. Kindberg, S. Mitchell, and G. Coulouris, "Modelling QoS Characteristics of Multimedia Applications", *Proc. IEEE RTSS 98,* Madrid, Spain, Dec 1998

[Spi00] M.D. Spiteri, *An Architecture for the Notification, Storage and Retrieval of Events*, Ph.D. Thesis, University of Cambridge, Jan 2000

[VC95] C. Venkatramani and T. Chiueh, "Design, Implementation and Evaluation of a Software-based Real-Time Ethernet Protocol", *Proc. ACM SIGCOMM'95*, Cambridge MA, USA, Aug-Sep 1995