A COLLABORATIVE MULTIMEDIA APPLICATION FOR A MOBILE ENVIRONMENT

Nigel Davies, Gordon S. Blair, Keith Cheverst and Adrian Friday

Distributed Multimedia Research Group,
Department of Computing,
Lancaster University,
Lancaster,
U.K.

Fax: +44 (0)524 593608
Telephone: +44 (0)524 65201
E-mail: nigel, gordon, kc, adrian@comp.lancs.ac.uk

ABSTRACT

With recent advances in portable computers and wireless networks, it is now feasible to provide sophisticated IT support for a range of mobile workers in areas such as the police, ambulance services and the utilities industries. However, to date, such support has been limited to providing remote access to central facilities such as e-mail and databases. This paper focuses on extending this level of support to include collaborative applications requiring multimedia communications. Such applications make considerable demands on the underlying systems infrastructure. More specifically, the paper discusses the role of such collaborative multimedia applications in supporting field workers in the electricity industry. A prototype conferencing application is described which enables field engineers to jointly view and annotate network diagrams and maps produced by a Geographical Information System. Voice communications is also provided to enable discussions to take place. The underlying distributed systems platform is also presented. The application has been successfully demonstrated over GSM.

1. INTRODUCTION

The areas of Computer Supported Cooperative Work (CSCW) [Baecker93] and mobile computing [Duchamp94] have become very topical in recent years. However, to date, little research has taken place into the design of CSCW based collaborative applications which are intended to operate in a *mobile* environment. This paper describes our efforts to build such a collaborative multimedia application designed to operate using wireless communications. The application is targeted towards supporting field workers in the electricity industry and encourages more rapid response to power failures during network maintenance by providing remote (and collaborative) access to information such as geographical data and the status of the distribution network.

The paper is structured as follows. Section 2 discusses the topic of collaborative mobile applications for mobile computers and looks more closely at supporting field workers using such applications. A set of requirements are also derived. Section 3 then presents a trial conferencing application designed to improve information flow between field workers in emergency situations. A platform to support this application is then described in section 4. Finally, section 5 contains some concluding remarks.

2. COLLABORATIVE MULTIMEDIA APPLICATIONS

There are numerous examples of mobile computing applications which provide users with remote access to central facilities (e.g. email, news and financial databases). These applications typically make relatively modest demands on their communications infrastructure and have simple client-server based architectures. In this section we focus on the requirements of an example of a more demanding class of mobile application, i.e. collaborative multimedia applications. Such applications involve two or more mobile or fixed users co-operating to achieve a common goal involving the manipulation of information in a range of media types. In particular, these applications often require that *continuous media types* (e.g. audio) [Anderson90] are supported (note that providing support for continuous media types in a mobile environment is an area of increasing research activity [Keeton93, Mah93]).

2.1. An Application Scenario

As part of the MOST (Mobile Open Systems Technologies for the utilities industries) project [Davies95a] we have conducted a detailed study of the requirements of field engineers within the U.K. power distribution industry and identified a number of application areas for collaborative mobile computing. In general, these applications are based on an examination of current working practices. As an example consider the following scenario.

All work within a regional electricity company is traditionally co-ordinated by a single control centre (see figure 1). The engineer supervising a particular repair job files a schedule with the centre some days in advance. This schedule describes in detail the stages involved in carrying out the work and, in particular, the sequence of switching which must be carried out to ensure that the work can be conducted safely (i.e. the section of network being operated on is isolated and earthed) and with the minimum of disruption to users. The control centre checks the switching schedule against its central diagram of the network state (this may be held on its computer system) and approves or rejects the schedule accordingly. Expert systems may be used by both the control centre and

the field worker in the development of the switching schedule [Cross93] although at present these systems are not integrated with the centre's representation of the global network state.

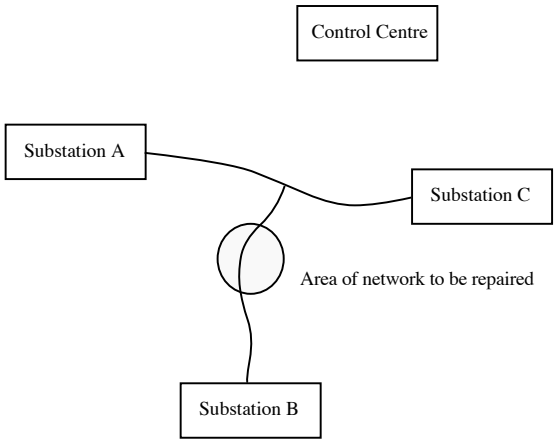

Figure 1: Maintenance of the Power Distribution Network

Once the schedule has been approved, the work may be carried out. On the day of the work, field engineers are dispatched to the appropriate switching points (substations A, B and C). The control centre then uses a voice-oriented private mobile radio system to instruct the staff as to which switches to operate. The use of a central co-ordinator helps to ensure the work is carried out in the correct order and allows the centre to maintain an up-to-date picture of the network's state. Once the work has been carried out the engineer must wait until returning to the office before completing the associated paper work.

2.2. Introducing Mobile Communications

There are clearly a number of disadvantages with the approach described above, in particular the lack of availability of global network state for the engineers in the field and the reduction in efficiency caused by the bottle-neck of a central point of control. The latter of these points becomes particularly important when faults occur requiring multiple unscheduled work items to be carried out. These disadvantages could be overcome by providing field engineers with a shared up-to-date picture of the network state. Providing such a picture would, of course, require a high degree of real-time synchronisation between field engineers operating on the network to ensure consistency between views of the network state.

Our aim is exploit the capabilities of emerging mobile networks such as GSM to provide the means for field engineers to have at hand the information they require in the field and to enable them to collaborate with one another to discuss relevant aspects of this information. For example, we aim to allow field engineers who are not physically co-located to view and manipulate shared diagrams and information and to communicate using voice. This would ease congestion at the control centre which, in addition to supplying individual engineers with information, is also responsible for coordinating collaboration between engineers. Hence, the application is novel since much of the communications involved is of a time critical, peer-to-peer nature (i.e. mobile field engineer to mobile field engineer) rather than conforming to the client-server architecture more commonly found in mobile applications.

2.3. Supporting Such Applications

In considering the development of a collaborative application to assist field engineers we have identified the following requirements:

i) Support for Geographical Information

Field engineers make extensive use of geographic information. Examples include maps showing the position of underground cables, circuit diagrams and reports of faults whose ordering is based on geographic location. An application to support field engineers must offer support for the display and manipulation of data of this type. Note that this does not equate to a requirement for a fully functioning GIS system; many of the features found in such systems are not required by field engineers. However, some GIS functionality is required since the application must be able to interpret and cross-reference information based on geographic co-ordinates (e.g. to allow diagrams showing cable layout to be superimposed on a physical map background).

ii) Support for Collaboration

Collaboration is an essential part of a field engineer's work, particularly in situations such as fault diagnosis. Moreover, as also highlighted, providing field engineers with increased collaboration support would help minimise the effect of the bottle-neck at the control-centre. Hence, there is a requirement to support groups of engineers collaborating using geographic information, e.g. a diagram of the current network state. Note that these groups are typically small with only two or three engineers collaborating at any one time.

iii) Support for Audio Communications

Audio communication is an essential tool for collaboration. In addition, many of the safety procedures used by field engineers rely on voice communications. These procedures could not be replaced by other media types in the foreseeable future and hence voice communications must be included in any application to support field engineers.

iv) Operation in Heterogeneous Processing Environment

In common with most large organisations electricity companies have a wide range of hardware architecture/operating system

configurations currently operational. Field engineers often require access to multiple resources both within the company and (increasingly) external to the company and hence must interact with heterogeneous service providers. The issue of heterogeneity is particularly significant in a mobile environment since mobile computers will clearly be required to interact with a wide range of service providers the selection of which must, in part, be determined by their physical location. Thus, while standardisation on communications systems such as GSM is important it is, as has become apparent in static environments, equally important that higher level distributed processing standards are employed to ensure an open system.

v) Operation in Heterogeneous Networked Environment

Field engineers require applications which can operate over a wide-area wireless network. In addition however, since a significant percentage of their time may be spent in an office or control-room environment, support is also required for applications to operate over and *exploit the benefits* accruing from the use of high-speed fixed networks. In other words, there is a requirement for applications to operate over networks offering a wide variety of QoS characteristics.

The following sections describe a prototype application based on the scenario identified above and a distributed systems platform which provides support for writing applications with these characteristics.

3. THE COLLABORATIVE APPLICATION

The prototype application we have developed allows field engineers to view geographical information (e.g. network diagrams) using a public-domain Geographical Information System (GIS) called GRASS. Support for collaboration with other field engineers is provided in two ways. Firstly, field engineers are able to synchronise the operations of their GIS with those of one or more remote engineers. In this way a common shared view of a particular piece of information can be maintained. Once a shared view has been established the application allows engineers to annotate the display with highlight marks which are subsequently propagated to all relevant remote engineers. Secondly, the application supports communication between field engineers via audio communications.

An example screen shot of the application's user interface is shown in figure 2. The conference manager or group co-ordinator component of the application is shown in the top left of the diagram. This component makes extensive use of icons [Gittins86] to represent users and their applications. On the left hand side of the group co-ordinator component are icons representing the sub-applications or *modules* which are available to the user (the telephone represents audio communications and the globe GIS services) and on the right hand side are icons representing users that can participate in conferences. In the centre is a list of the current conference participants. Under each participantÕs icon is a column of further icons which represent the modules which that user is currently running in conference mode (as opposed to stand-alone mode). By selecting these icons in rows the user is able to easily direct messages to a subset of the conference members. The user is able to select (by clicking on a single button) whether or not an operation (for example drawing a line in the GIS module) is propagated to those group members whose module icons have been selected.

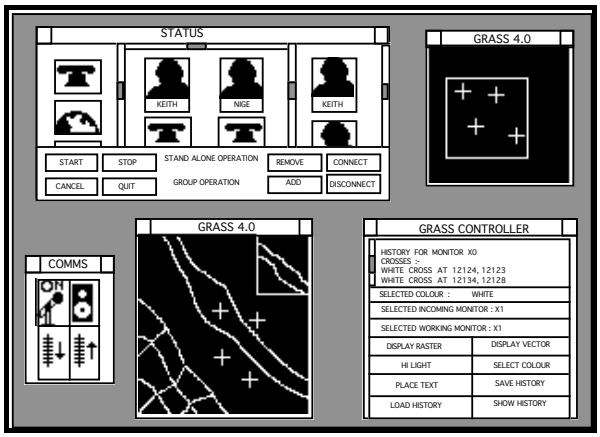

Figure 2:The user interface

Considerable emphasis has been placed on reporting to the user the state of the communications links between local modules and remote users' modules. In our prototype this is achieved by colouring the module icons under each user in the central portion of the display to reflect connectivity. If a module icon is visible under a user icon then that user is actually running that module in group mode. The module icon's background colour indicates whether or not the user's module is currently reachable. So for example, if a communications failure occurred this would be indicated by a red background.

Users are assisted when performing operations within the group module by the appropriate highlighting of both user and module icons. As an example of this, before an operation button has been pressed (e.g. add user) all such operation buttons would be highlighted whereas all user icons would be greyed. This use of greying is to warn the user that currently clicking on a user icon would have no effect. However, on clicking an operation button the operation buttons would become greyed and the user icons (and cancel button) would become highlighted.

Figure 2 also shows the user interfaces to the audio conferencing module (bottom left) and the GIS services (bottom middle, bottom right and top right).

4. THE UNDERLYING PLATFORM

A distributed systems platform has been developed to support the trial application described above. This platform is based on emerging distributed systems standards to enable operation in a

heterogeneous environment. This area of standardisation has seen intense activity in recent years as witnessed by efforts such as the Object Management Group's Common Object Request Broker [OMG91], the Open Software Foundation's Distributed Computing Environment [OSF92] and the ISO's ODP standard [ISO95a, ISO95b]. Our platform is based on the latter standard. More specifically, the implementation of our platform is based on the ANSAware software suite [APM93]. This software suite is itself based on the ANSA architecture which has had a profound influence on the RM-ODP. Thus, the platform realises many of the concepts contained within the RM-ODP while making a number of simplifying assumptions to allow efficient implementation.

4.1. The ANSAware Distributed System

4.1.1. The ANSAware Object Model

ANSAware is based on a location independent *object-based* model of distributed systems (its computational model). In this model, interacting entities are treated uniformly as *objects*, i.e. encapsulations of state and behaviour. Objects are accessed through *operational interfaces* which define named operations together with constraints on their invocations. Operations are accessed through a binding; at present in ANSAware, such bindings are established implicitly before first access (see below).

Objects offering services are made available for access by *exporting* interfaces to a database of service interfaces known as a *trader*. An object wishing to interact with a service interface must *import* the interface by specifying a set of requirements in terms of a interface type and attribute values. This will be matched against the available services and a suitable candidate selected. Any number of traders can exist and these may be linked or federated to allow access to services in different administrative domains.

Also central to the ANSAware object model is the notion of *transparency* whereby selected aspects of systems can be made invisible to applications. This is achieved by means of notional *transparency functions* interposed between the application and the support layers (it should be noted that the application of transparency functions is under user control and hence transparencies are *selective)*. An important example of a transparency is *group transparency* which allows multiple services to be invoked via a single interface. Other transparencies include location, access, transaction, replication, and migration transparencies.

4.1.2. Engineering of ANSAware

To provide a platform conformant with the object model, the ANSAware suite augments a general purpose programming language (usually C) with two additional languages. The first of these is IDL (Interface Definition Language), which allows interfaces to be precisely defined in terms of operations as required by the computational model. The second language, dpl (distributed processing language) is embedded in a host language, such as C, and allows interactions to be specified between programs which implement the behaviour defined by these interfaces. Specifically, dpl statements allow the programmer to *import* and *export* interfaces, and to *invoke* operations in those interfaces. A number of system services are supplied which include a trader service and a factory service for creating new objects.

In the engineering infrastructure, the binding necessary for invocations is provided by a remote procedure call protocol known as REX (Remote EXecution protocol) or a group execution protocol know as GEX (Group EXecution Protocol). This is layered on top of a generic transport layer interface known as a *message passing service* (MPS). A number of additional protocols may be included at both the MPS and the execution protocol levels and these may be combined in a number of different configurations. The infrastructure also supports lightweight threads within objects allowing multiple concurrent invocations.

All the above engineering functionality is collected into a single library, and an instance of this library is linked with application code to form a *capsule*. Each capsule may implement one or more computational objects. In the UNIX operating system, a capsule corresponds to a single UNIX process. Computational objects always communicate via invocation at the conceptual level but, as may be expected, invocation between objects in the same capsule is actually implemented by straightforward procedure calls rather than by execution protocols. ANSAware currently runs on a variety of operating systems platforms including various flavours of UNIX, VMS and MS-DOS/Windows.

4.3. Extensions for Multimedia and Mobility

We have extended the basic ANSAware platform to support the transmission of continuous media and also to operate more efficiently in a mobile environment. The key changes are discussed below.

4.3.1. Explicit QoS-Managed Bindings

The first major change has been to extend the support for bindings in ANSAware. A useful side-effect of this work is that our version of the ANSA platform is now aligned closer to the current RM-ODP standard. To meet the requirement for an abstraction of real-time data flow over time, we have added the concept of *explicit stream bindings* (as described in the RM-ODP) to our ANSA based platform. Stream bindings provide an end-to-end abstraction over continuous media communication and support arbitrary m:n connections, i.e. they allow m sources to be connected to n sinks.

Within our platform stream bindings are established using an explicit bind operation which takes as parameters the source and sink interfaces to be bound and a further set of desired QoS parameters. As with stream interfaces, these parameters can include a specification of the desired throughput, latency and jitter associated with the binding (more details of the specification of QoS parameters for continuous media bindings can be found in [Coulson95]). Clients are returned a binding control

interface as a result of an explicit bind operation. To control the QoS of the stream once the binding has been established the control interface includes a pair of operations *setQoS()* and *getQoS()*. These operations take as arguments a set of QoS parameters which can then be passed by the stream binding to the underlying transport protocol. A call-back mechanism is also provided to inform client objects of QoS degradations reported by the underlying transport service.

We have also added a new class of explicit binding for use with operational interfaces. These bindings are established using the binder$Bind operation as above but take as arguments operational interfaces. The resulting binder control interface is identical to that used for stream bindings except that clients are allowed to specify and monitor a different set of QoS parameters associated with the binding. This enables, for example, a client to ask to be informed when the QoS service supplied by the binding falls below a specified threshold. Of particular relevance to mobile applications is the ability to monitor the possibility of sending or receiving messages via a specified binding without having to explicitly send application level test messages, i.e. applications can delegate responsibility for guaranteeing QoS assertions to the system. This is of significance since it allows mobile applications to be structured in an event based fashion (c.f. polling). For example, through the use of our bindings it is possible to assert that the absence of messages on a given interface is a result of their being no traffic intended for the specified interface rather than a result of communications failure. In addition, QoS driven bindings allow the system to optimise the use of test messages which might otherwise be duplicated if left to individual applications, e.g. if multiple applications wished to test QoS assertions between the same pair of objects. Further details of the structural changes in applications and system services possible as the result of the introduction of QoS driven bindings can be found in [Davies94]).

4.3.2. Underlying Protocols

The current execution protocols in ANSAware (i.e. REX and GEX) are not well suited to operate over mobile networks. The problem with the REX protocol is that it takes no account of the characteristics of the underlying network. More specifically, parameters such as the number of retry attempts and the interval between these attempts are fixed at installation time. Thus, when a system configured to operate over an Ethernet is run over a low-speed network the absence of congestion control within REX means that almost no data is actually communicated between user processes. Instead, the network becomes overloaded with REX control messages.

The GEX protocol is wholly unsuitable for use over many wireless networks. The protocol has been designed to avoid a central point of failure for group communication. Instead, group events such as membership changes and message arrivals are co-ordinated by all group members exchanging state information using an internal token-based protocol. This generates multiple redundant messages if hardware broadcast is supported and involves the establishment of multiple point-to-point connections between group members if hardware broadcast is not supported.

Consequently we are in the process of extending the ANSAware protocol suite with a pair of new execution protocols called QEX (Quality-of-service driven remote EXecution) and G-QEX (Group-Quality-of-service driven remote EXecution). These protocols will gather information on specified bindings based on (for example) the number of retries and average delay they experience over a given channel and be able to pass this information on to interested clients. This will allow our new protocols to implement congestion control *and* to propagate this information to applications in order that they can 'back-off' to further reduce network traffic. In the case of G-QEX we hope to be able to adjust the protocol's approach to supporting groups (i.e. centralised or distributed) based on network and user QoS information. Where necessary QEX and G-QEX will periodically test bindings in order to be able to guarantee QoS assertions. A first version of QEX has now been implemented and is used to support the trial application.

4.3.3. Enhanced Trading Functionality

Finally, the implementation of the trading function in ANSAware assumes a hierarchical patterns of federation. In more detail, ANSAware only allows traders to be arranged into a one level hierarchy with a single master trader at the top of the hierarchy and a number of sub-traders as the leaf nodes. Requests for services which can not be matched by a sub-trader are passed on to the master trader but there is currently no easy way for the master trader to propagate such requests back down to additional sub-traders. This hierarchical arrangement while adequate for fixed networks is inappropriate for mobile systems where each portable computer is likely to have its own trader in order that it can continue operation during periods of disconnection. Moreover, given the nature of the communications links a federation scheme which requires traversal up and down a hierarchy (possibly requiring multiple dial-up connections to be established) is clearly unsuitable.

We have addressed this issue by introducing a mechanism to enable peer-to-peer linking of traders. The links thus established can have constraints imposed on them ensuring that they are traversed only when there is a strong likelihood of the remote trader being able to satisfy the request. For example, we can specify that services owned by a particular user are always found on a specified machine and traverse the link to that machine only when looking for that user's services.

5. CONCLUDING REMARKS

To date there has been little work on the development of collaborative multimedia applications to operate in mobile environments. This paper has described experimentation with such applications to support field engineers in the electricity industry. A distributed systems platform has also been developed

to support such applications. This platform features quality of service managed bindings which offer feedback on the quality of service of the underlying network, remote procedure call protocols which adapt to the quality of service obtained, and a system of federating traders better suited to peer to peer communication in a mobile environment.

The application and associated platform has been demonstrated running in a heterogeneous environment consisting of SUN workstations, desktop PCs and portable PCs. Mobile communications is provided by either analogue cellular phones, operating at 2,400 bits/sec and GSM at 9,600 bits/sec. Alternatively, a network emulator can be used allowing us to simulate the varying degrees of connectivity likely to be experienced by field engineers during a typical operational cycle [Davies95].

ACKNOWLEDGEMENTS

This work is carried out as part of the MOST project, involving Lancaster University and EA Technology, under the DTI/ EPSRC Open Distributed Systems Architecture (ODSA) Programme (Grant Reference GR/H87704).

REFERENCES

[Anderson90] Anderson, D.P., S. Tzou, R. Wahbe, R. Govindan, and M. Andrews, “Support for Continuous Media in the Dash System”, Proc. 10th International Conference on Distributed Computing Systems, Paris, May 1990.

[APM93] A.P.M. Ltd. “ANSAware 4.1 Application Programming in ANSAware”, Document RM.102.02, A.P.M. Cambridge Limited, Poseidon House, Castle Park, Cambridge CB3 0RD, UK, February 1993.

[Baecker93] Baecker, R., “Readings in Groupware and Computer Supported Cooperative Work”, Morgan Kaufmann, San Mateo CA, ISBN 1-55860-241-0, 1993.

[Coulson95] Coulson, G., G.S. Blair, F. Horn, L. Hazard, and J.B. Stefani, "Supporting the Real-Time Requirements of Continuous Media in Open Distributed Processing", To appear in Computer Networks and ISDN Systems, Special Issue in Open Distributed Processing, 1995.

[Cross93] Cross, A.D., J.R. Brailsford, and A.T. Brint, “Expert Systems to Support Network Switching”, Proc. 12th International Conference on Electricity Distribution, CIRED 1993, Birmingham, U.K., pp 17-21 May, 1993.

[Davies94] Davies, N., S. Pink, and G.S. Blair, "Services to Support Distributed Applications in a Mobile Environment", Proc. 1st International Workshop on Services in Distributed and Networked Environments, Prague, Czech Republic, June 1994.

[Davies95a] Davies, N., G.S. Blair, A. Friday, A.D. Cross, and P.F. Raven, "Mobile Open Systems Technologies For The Utilities Industries”, CSCW Issues for Mobile and Tele-Workers, Dix, A. (ed), Springer-Verlag, 1995.

[Davies95b] Davies, N., G. S. Blair, K. Cheverst, and A. Friday, "A Network Emulator to Support the Development of Adaptive Applications”, 2nd USENIX Symposium on Mobile and Location Independent Computing, Ann Arbor, Michigan, April 1995.

[Duchamp92] Duchamp, D., "Issues in Wireless Mobile Computing”, Proc. Third Workshop on Workstation Operating Systems, Key Biscayne, Florida, U.S.A., pp 2-10, 1992.

[Gittins86] Gittins, D., "Icon-based human computer interaction", International Journal for Man-Machine Studies, Vol. 24, pp 519-543, 1986.

[ISO95a] ISO/IEC Recommendation X.902, International Standard 10746-2, “ODP Reference Model: Descriptive Model”, January 1995.

[ISO95b] ISO/IEC Recommendation X.903, International Standard 10746-3, “ODP Reference Model: Prescriptive Model”, January 1995.

[Keeton93] Keeton, K., B.A. Mah, S. Seshan, R.H. Katz, and D. Ferrari, “Providing Connection-Oriented Network Services to Mobile Hosts”, Proc. USENIX Symposium on Mobile and Location Independent Computing, Cambridge, Massachusetts, U.S.A., pp 83-102, August 1993.

[Mah93] Mah, B., S. Seshan, K. Keeton, R. Katz, and D. Ferrari, "Providing Network Video Service to Mobile Clients”, Proc. 4th Workshop on Workstation Operating Systems (WWOS-IV), Napa, U.S.A., October 1993.

[OMG91] OMG, "The Common Object Request Broker: Architecture and Specification (CORBA)", Report 91.12.1, The Object Management Group, 1991.

[OSF92] The Open Software Foundation, “Distributed Computing Environment”, 11 Cambridge Center, Cambridge, MA 02142, USA.