DISTRIBUTED SYSTEMS SUPPORT FOR MOBILE APPLICATIONS

A. Friday and N. Davies

Introduction. Future computer environments will include mobile computers which will either be disconnected, weakly inter-connected by low speed wireless networks such as GSM, or fully inter-connected by high speed networks ranging from Ethernet to ATM. Two key characteristics of such environments are:-

(i) a heterogeneous processing environment (including relatively low-power mobile hosts) and,

(ii) rapid and massive fluctuations in the *quality of service* (QoS) provided by the underlying communications infrastructure.

The first of these issues, i.e. heterogeneity, can be addressed by exploiting emerging distributed systems standards such as the International Standards Organisation Reference Model for Open Distributed Processing (RM-ODP) [ISO,95], the Open Software Foundation's Distributed Computing Environment (OSF-DCE) [OSF,91] or the Object Management Group's Common Object Request Broker Architecture (OMG-CORBA) [OMG,91]. Such standards provide applications with uniform computational models for accessing services and enables them to operate over a variety of processor/operating system configurations. The second of these issues, QoS fluctuations, is, we believe, one of the most fundamental problems in the field of mobile computing. In this paper we consider this issue and propose extensions to emerging distributed systems standards to support mobile computing. These extensions are designed to allow the development of *adaptive or reactive applications* [Katz,94],[Davies,94] which are able to tailor their behaviour based on changes in their communications infrastructure.

Characteristics of Mobile Environments. A key characteristic of mobile computing environments is that end systems experience differing degrees of connectivity during typical operational cycles. In particular, systems are expected to function when connected to a range of networks including high-speed networks (fully connected operation) and low-speed wireless networks (weakly connected operation), and when totally disconnected (disconnected operation). A fixed network environment can provide relatively reliable communications with a bandwidth of between 1 and 100 Mbps. Such characteristics make it possible to design applications and operating systems services with little regard for optimising network traffic. Indeed, latency is often the overriding factor in determining the performance of fixed distributed systems while bandwidth is in plentiful supply.

If an end-system moves from a fixed network connection to a local-area wireless connection then the bandwidth available will fall to between 10 Kbps and 10 Mbps. In addition, the characteristics of the channel will also change. For example, the increased number of bit-errors, cell hand-offs and coverage blackspots will lead to a significant number of packets being lost. While it might be assumed that these packet losses can simply be treated as a reduction in the overall bandwidth available Cáceres demonstrated in [Cáceres94] that it is important that network protocols are tuned for networks with these packet-loss characteristics by showing that the performance of TCP was significantly reduced when it was operated over a wireless network with packet-losses due to bit-errors and cell-handoffs.

Finally, when an end-system requires wide-area wireless connectivity the available bandwidth will drop to between 0 (in coverage blackspots) and 9.6 Kbps (analogue cellular systems can usually support 1200 bps, both the GSM digital data service and CDPD support 9.6 Kbps). In all cases the latency of establishing a connection also increases by several orders of magnitude.

Furthermore, there are cost implications associated with different network types. In the case of local-area and fixed networks the costs are, in the case of most institutions, covered centrally and users are not charged according to usage. In the case of wide-area wireless communications or when mobile users must rely on dial-up lines the situation is very different. Consider the case of a user whose underlying communications is being provided by a public cellular telephone provider. The actual cost of transmitting information will depend on a vast array of factors including the tariff the user initially signed up for, the time of day, the users physical location, whether or not the user is having to exploit a roaming agreement to obtain coverage and whether the data is being sent via an explicit connection or via a short message or datagram service. Many of these factors will change dynamically and can make a

A. Friday and N. Davies
Distributed Multimedia Research Group, Department of Computing, Lancaster University, Lancaster, LA1 4YR, U.K.
E-mail: most@lancs.comp.ac.uk.

substantial difference to the cost of connectivity.

To date, the transition between modes of operation has been a heavy-weight process often involving both hardware and software re-configuration. However, the advent of technologies such as the MINT mobile internet router (The MINT router is being developed as part of the Walkstation project [Hager,93] and enables mobile computers to dynamically exploit the services offered by a number of networks by supporting seamless transitions between network types.) promise to significantly reduce the overheads associated with the transition process and lead to a more dynamic environment in which an end-system observes rapid and marked fluctuations in network characteristics. There is, therefore, an emerging requirement for a new class of distributed system service designed specifically to operate in this dynamic environment. Such services must avoid assumptions about their underlying support environment which prevents them from operating effectively across a range of networks. We term this new class of service *adaptive services.*

Distributed Systems Support For Adaptive Services. Given the requirement for adaptive services it is important to determine the role of distributed systems platforms in supporting such services (recall that distributed systems platforms are important in mobile environments to overcome the problems of heterogeneity). Research at Lancaster has shown that current distributed systems platforms are unsuitable for supporting adaptive services because they have been designed primarily as information hiders [Davies,96]; abstracting over the underlying system characteristics to provide applications with a uniform computational model which has little relationship to the supporting infrastructure. In contrast, adaptive services require explicit information regarding the underlying infrastructure in order that they can optimise their behaviour. As part of the MOST (Mobile Open Systems Technologies) project at Lancaster we have developed a prototype distributed systems platform which collects and manages QoS information for adaptive services. The implementation of our platform is based on APM Ltd.'s ANSAware software suite [APM,89]. This software suite is itself based on the ANSA architecture which has had a profound influence on the RM-ODP. Thus, the platform tackles the problem of developing applications to operate in a heterogeneous environment. The ANSA programming model is based on a location-independent object model where all interacting entities are treated uniformly as encapsulated objects. Objects are accessed through operational interfaces which define named operations together with constraints on their invocation. Objects are made available for access by exporting interfaces to a special object known as the *trader*. An object wishing to interact with this interface must then import the interface from the trader 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. At this stage, an implicit *binding* is created to the object supporting the interface, i.e. a communication path is established to the object. Invocation of operations can then proceed. To provide a platform conformant with the above programming 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, arguments and results. 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. 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) which are layered on top of a generic transport layer interface known as a *message passing service* (MPS).

The execution protocols in ANSAware are not well suited to operation over mobile networks. In our work, we have introduced a new protocol called QEX (QoS EXecution Protocol) to replace the REX protocol (we are also planning to replace GEX with a protocol G-QEX although this is still at the design stage). Currently, the REX RPC protocol 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. Hence, 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. If the parameters of the REX protocol are modified and the implementation recompiled to operate over a low speed link it performs poorly over Ethernet. To overcome these difficulties, QEX analyses the characteristics of the communications medium for each interaction and adjusts itself to make the best use of the link. The general approach in QEX is to estimate the underlying characteristics of each channel based on information obtained from round-trip times and sizes of messages. The round-trip time statistics are smoothed using a moving average calculation and fed back into the protocol to load the retry interval timers.

In addition to maintaining and using QoS information to improve its performance QEX also provides application programmers with a number of QoS related functions. These functions are available for each binding between objects and are accessed via a binding control interface which is created explicitly by the programmer. Note that the creation of explicit bindings is in contrast to the approach adopted in vanilla ANSAware where all bindings are created implicitly.

The QoS parameters associated with each binding are: the throughout, the maximum *idle times* for the client and server interfaces and a number characteristics associated with the cost of the channel (e.g. tariff structure). The idle time of an interface is the time that has elapsed since the interface last sent or received a message (the use of idle times is explained later in this section). The binding controller interface allows programmers to (among other operations) register for changes in the QoS and hence, for example, monitor the idle time of interfaces 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). Thus, through the use of our bindings, it is possible to assert that the absence of messages on a given interface for a given period of time (the elapsed idle time) is a result of there 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.

As an example of how QoS bindings can be used to tailor an application's behaviour we now consider a simple database query application developed as part of MOST. The application supports a component which provides mobile users with access to a remote database. When a mobile user issues a query the number of fields returned for each matching record is determined dynamically based on the number of matches and the network QoS. Hence, when the user is connected by a slow-speed network only information which is likely to be of use in further restricting the search is returned, while in a high-speed network where latency is more critical than throughput, the entire found-set is returned (subject to a specified threshold). The application also gives QoS feedback to users in order to allow them to adapt their behaviour to match the network characteristics. This stemmed from an original application requirement to provide a graphical monitor similar in appearance to the signal-strength meters commonly found on cellular communications equipment. In practice the display provides substantially more detailed feedback to users allowing them to see, for example, where bottlenecks are occurring in complex group based activities which are also supported by the MOST application. This enables users to adjust their patterns of work, e.g. switching between synchronous (shared white-board) and asynchronous (email) communications, to make the best use of the available network QoS.

Conclusions. Mobile end-systems will exist in environments which are subject to rapid and marked change. In order to make best use of the available resources applications and system services will have to adapt to these changes. In this paper we have described a prototype distributed systems platform developed at Lancaster which is designed to support this process of adaptation. Further details on our work on QEX and its associated platform can be found in [Davies,95],[Friday,96] or via the WWW at `http://www.comp.lancs.ac.uk/computing/most/`.

References.

[APM,89] A.P.M. Ltd. "The ANSA Reference Manual Release 01.00", APM Cambridge Limited, UK. March 1989.

[Cáceres94] Cáceres, R., and L. Iftode. "The Effects Of Mobility on Reliable Transport Protocols." *Proc. 14th International Conference on Distributed Computer Systems*, Poznan, Poland, 22-24 June, 1994. Pages 12-20.

[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.

[Davies95] Davies, N., G.S. Blair, K. Cheverst, and A. Friday. "Experiences of Using RM-ODP to Build Advanced Mobile Applications." Distributed Systems Engineering Vol. 2 No. 3, Pages 142-151.

[Davies96] Davies, N. "The Impact of Mobility on Distributed Systems Platforms." *Proc. International Conference on Distributed Platforms*, Dresden, 1996.

[Friday96] Friday, A. "Extensions to ANSAware for advanced mobile applications." *Proc. International Conference on Distributed Platforms*, Dresden, 1996.

[Hager,93] Hager, R., A. Klemets, G.Q. Maguire, M.T. Smith, and F. Reichert. "MINT - A Mobile Internet Router" *Proc. IEEE VTC'93*, Secaucus, NJ, USA. 1993.

[ISO95] ISO/IEC Draft Recommendation X.902, International Standard 10746-1, "ODP Reference Model: Overview", January 1995.

[Katz94] Katz, R.H. "Adaptation and Mobility in Wireless Information Systems." IEEE Personal Communications Vol. 1 No. 1, Pages 6-17.

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

[OSF91] OSF. "Distributed Computing Environment: An Overview", OSF, April 1991.