Adaptive Middleware for Heterogeneous Defence Networks - an Exploratory Simulation Study

B.McClure,†§ J. Indulska,§ T.A. Au†
†*Defence Science and Technology Organisation, PO Box 1500, Salisbury, SA 5108, Australia.*
§*Department of Computer Science and Electrical Engineering, The University of Queensland, Brisbane, Qld 4072, Australia.*
email: Mcclures@ozemail.com.au, Jaga@csee.uq.edu.au, Andrew.Au@dsto.defence.gov.au

Abstract

This paper presents the design and evaluation through a discrete event simulation of an ODP-based Adaptive Computing Architecture which manages network resources in large-scale heterogeneous error-prone networks. The emphasis is given to network (communication) adaptation of this architecture simulated for an exemplar defence network. The results show that, for this network, the architecture provides significant improvement in terms of higher priority requests meeting their QoS requirements and adaptation to link failure under heavy link utilisation. In addition, link utilisation is lower with the architecture active.

1. Introduction

Today, to fulfil the operational requirements of modern battlefield interoperability, defence organisations require the deployment of distributed applications such as video on the Web, situational awareness and collaborative planning. Existing system and application architectures, which typically assume LAN quality networking, do not readily accommodate these new technologies. Further problems emerge when this situation is combined with the mix of heterogeneous networking technologies and WAN links used by many large organisations [1, 2].

Defence networks comprise communication technologies ranging from Giga-bit fibre-optic links to satellite channels and HF radio links. Network communications protocols used include ATM, IP, X.25 and various military "link" protocols. There are some special requirements that impact on defence networking. First the network must provide a minimum level of service at all times — the network needs to be resilient to intentional damage; e.g. physical or electronic attacks on communications links or nodes should not bring down the entire network. Second, the network should degrade gracefully when it is damaged or overloaded. In particular, the network should ensure that at least the most critical communications traffic is carried. Secondary requirements include ensuring that the network is secure, coping with intrinsically "poor" communications links and accommodating enforced communications silence. The diverse nature of defence networks and the increasing use of communications intensive applications for critical tasks present significant problems. In essence, there is a real need to be able to manage the use of network and communications resources so that it is possible to dynamically adapt to changes in service requirements and network conditions.

There has been extensive research in the area of Quality of Service (QoS) management in computer networks [3, 4, 5] and in particular Schill *et al*. [6] describe a network topology model that incorporates round-trip delay estimates and link costs and specifies the data-link protocols used. However, none of the above work adaquately addresses the full scope of the problem in the defence environment. Supporting distributed applications in a defence network requires a comprehensive architecture for resource management, capable of selectively providing guaranteed service according to an over-arching policy. A piece-meal solution relying on applications and services working cooperatively is not sufficient. Policy based resource management needs to form an integral part of the distributed computing infrastructure.

We propose an Adaptive Computing Architecture (ACA) to manage enterprise-wide network resources according to predefined adaptive policies. These policies support adaptation of client – server bindings and use of network resources to address problems that arise from changes in network availability and demand. The ACA is very flexible in that it supports changes to policies and the representation of the enterprise network at runtime and has an adaptive mechanism that can be tailored to particular needs. A limited "proof of concept" prototype has been developed [7], which implements some of the ACA functions on the CORBA distributed platform.

The focus of this paper is on the ability of large heterogeneous networks to meet QoS requirements of high priority requests. In the absence of appropriate adaptability strategies, a system's response to increased load or reduced capacity will be unacceptable application performance, rejection of critical service requests, or even total network collapse; e.g. Ethernet "meltdown". Adaptation in the ACA can be triggered in two ways; by changes in availability of network resources, or by changes in demands on network resources from the objects that use them. The mechanisms for adaptation can be grouped into three broad classes: network adaptation, binding adaptation and application adaptation.

Network adaptation covers the allocation of communications resources to application objects and their dynamic adjustment. The infrastructure initially selects communications paths, and allocates resources to a binding between objects to satisfy their QoS requirements. Later, it may adjust the route and resource allocations, possibly by preemptively reallocating resources to higher priority tasks.

Client – server binding adaptation covers a variety of mechanisms that mediate the objects' communication needs, statically or dynamically. When a client object requests a service from "the network", the request can typically be satisfied using objects available at various locations. The ACA can manage network load and requested QoS by selecting the most favourably located service object. In some cases, filter or adaptor objects can be inserted between the client and service to adjust network traffic or cope with interface mismatches.

Application adaptation covers cases where the application objects themselves need to be involved in the adaptation process. For example, when a client object makes an initial request for service, it may need to negotiate QoS measures with the networking infrastructure. Subsequently, the ACA may notify the client that the QoS measures have changed, offering it the chance to adapt to the new measures or terminate.

The ACA directly supports network adaptation and binding adaptation and provides the hooks for application adaptation. Our approach relies on having knowledge of the network and the QoS that it is capable of providing at any given time. For this purpose, a method for modelling network topology and its static QoS measures (potential QoS) has been defined. To deal with the scalability problem, the method allows modelling at various levels of abstraction. This network topology and QoS information is augmented with dynamic network configuration and load information to achieve a description of the current network state.

In order to validate network adaptation of the ACA it was necessary to test it on a larger network than that used by our inital prototype. Defence Science and Technology Organisation (DSTO) in Australia is currently building an Experimental Command, Control Communications and Intelligence (C3I) Technology Environment (EXC3ITE). This environment is an interstate heterogeneous network infrastructure including satellite links. The EXC3ITE network will support a number of emerging CORBA based applications. Potential users of the EXC3ITE network require methods for moderating the use of network resources in this environment. Therefore, the EXC3ITE network has been modelled as a realistic exemplar heterogeneous defence network. A simulation model has been built to test complex functionality of the network adaptability before a full ACA implementation is considered for EXC3ITE.

This paper presents a qualitative assessment of the performance of the ACA over the EXC3ITE network compared to the performance without the architecture. In particular, the ACA is assessed with regard to: i) overall satisfaction of application QoS requirements taking priority into account, ii) benefits to the individual users competing for network resources subject to i), iii) bandwidth utilisation of the network and iv) validity of our model of topology and QoS. Our simulation results show that i) and ii) are better for high priority requests when the architecture is active, for medium to heavy network loads. Secondly, we show that the network utilisation will be reduced when the ACA is active. This is a result of connection requests being denied, delayed or reduced in response to network overload and is one cost of providing better performance to higher priority requests.

2. Overview of the Adaptive Computing Architecture

The ACA is based on the Open Distributed Processing (ODP) framework for building distributed applications. It therefore assumes that supporting services are available in the infrastructure such as the Trader and Type Manager. These ODP functions provide a broker service, and management of interface type descriptions and relationships (such as runtime type compatibility checking). An adaptive network resource management framework needs to maintain three key kinds of information about the system. First, management policies need to be modelled to allow adjustments of policy at runtime. Second, the interface specifications and QoS requirements of applications and services need to be modelled so that the management system can support a dynamic environment of communicating objects. Finally, network topology and QoS measures need to be modelled so that the management system can make appropriate decisions based on network conditions.

Figure 1 is a high-level logical depiction of our resource management architecture whose functionality is provided as a distributed set of services. At the centre of the architec-

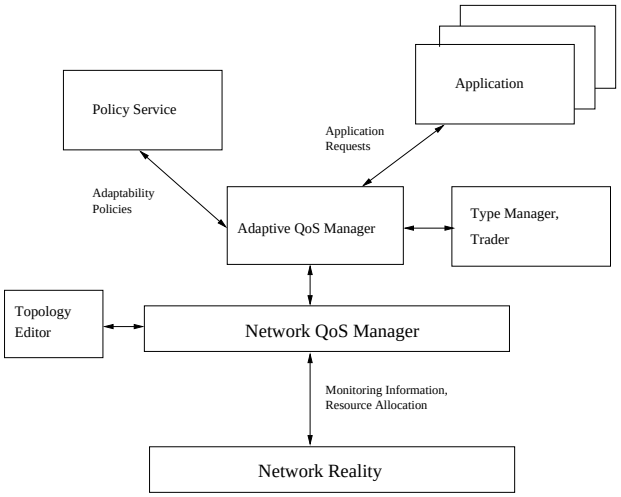

Figure 1: **Adaptive Computing Architecture.**

ture is the Adaptive QoS Manager (AQM) that manages resources to satisfy application requirements according to current adaptation policies and network conditions. The Policy Service stores the current adaptation policies. The Network QoS Manager (NQM) maintains the system's model of the network topology, QoS and the current state of the network. The Local Monitor performs monitoring of QoS delivered to applications. The Type Manager is an ODP service that is used here to store interface and QoS descriptions for applications and services. The Trader is another ODP function that manages current offers of service, and matches them against requests from clients via the AQM.

2.1 Adaptive QoS Manager

The function of the Adaptive QoS Manager (AQM) is to establish and manage bindings between application and service objects. While these bindings are made at the request of individual client objects, the AQM is responsible for managing enterprise resource usage according to the current policies. This is done by mediating new requests for service and actively managing resource usage by existing bindings. If network resources are not available (and policy dictates), the AQM may take action to manage resource usage. In the worst case, it can pre-emptively shut down an existing binding from another application to free up resources. The AQM can also adjust an existing binding to optimise its resource usage. Priority is an important concept here because it is used to determine which requests are pre-empted or rejected first when resources are insufficient. In this work, priority is being associated with a binding request and priorities are restricted to three levels: high, medium and low.

2.2 Policy Service

The Policy Service stores the adaptation policies that specify *how* and *when* the Adaptive QoS Manager should adapt to changing resource availability. Our model for adaptation policies is based on Sloman's policy language [8].

2.3 Network QoS Manager

The Network QoS Manager is the component in the adaptive resource management architecture that holds knowledge of the state of the network. This knowledge consists of two distinct parts. Firstly, the nominal network topology and its associated QoS measures (e.g. link capacities) are expressed using the Network QoS Specification Language (NQSL). Secondly, the NQM must be informed of the current network load by management interfaces on critical links and gateways. The NQM may also receive monitoring information from Local Monitors (LM) regarding the QoS delivered to established bindings. The other functions of the NQM include using the information that it gathers to perform route selections and preemption on the basis of required QoS parameters. This function is used by the AQM to allow it to make optimal service selections if there are several servers available.

2.4 The Model For Network Topology and QoS

The ACA depends on having detailed information about the topology of the network and its QoS and functional characteristics in order to make adaptive decisions. The networks to be represented may be large and heterogeneous. For these reasons, a Network QoS Specification Language (NQSL) has been defined to represent network topology and QoS at various levels of abstraction. NQSL describes networks in terms of hierarchical QoS domains, logical links, elements and gateways [9].

3. Simulation Model

The EXC3ITE network being modelled consists of 8 LANs connected by a mixture of satellite and ATM links depicted in Figure 2. The satellite links are those links eminating from truck-mounted nodes 1 and 2.

Each LAN has a number of workstations connected to an Ethernet bus. There are several unique characteristics of the EXC3ITE network which result in the effective ATM throughput being limited to 512 Kbps and reduce the routing ability of the network. We assume that all traffic on the LANs and WANs originate from the applications that are using the ACA. The IP traffic is carried through the network using Classic IP. Two of the LANs are connected to mobile trucks with satellite dishes. These trucks are packed up, moved and re-established at a new location from time to time. Two other LANs are "relocatable", able to be packed up in boxes and connected to different points in the commercial ATM networks. Neither of these forms of mobility result in dynamic address changes. However, the mobility of the trucks may result in changes to the available QoS.

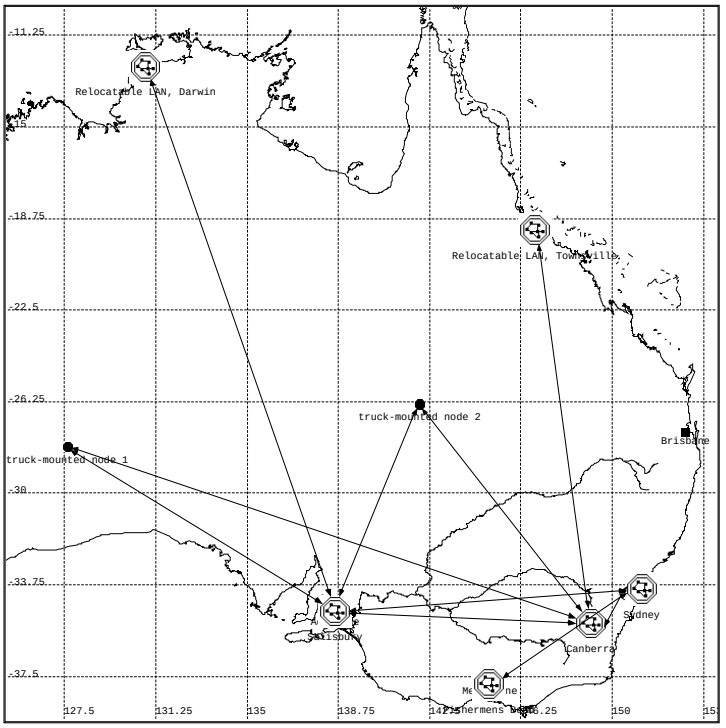

Figure 2: **EXC3ITE Network Layout.**

The OPNET commercial network and distributed system simulation package has been used to support the simulation. In particular, the example TCP, IP and Ethernet models supplied with OPNET have been used as the basis of the network simulation (to simulate communication between both applications and ACA servers). These example models incorporate most functionality of the relevant standards for each of these communication protocols [10].

3.1 Representation of the AQM in the simulation model

The simulation model of the ACA represents an *implementation* of the architecture. This includes some design decisions influenced by the network, applications and policies represented that are not inherent in the architecture itself. The AQM is implemented as two distributed components. The first part is a local proxy, which together with the policy service is present on all workstations. This part of the AQM implements policies local to the workstation and provides an interface between applications and the architecture. Second, aspects of the AQM which have wider impact form part of a LAN proxy. For example, policies that pre-empt bindings from other workstations on LANs have a wider impact. There is one LAN proxy on every LAN.

3.2 Representation of the NQM in the simulation model

The NQM is also implemented in a distributed manner and is present on one workstation in every LAN (ie the LAN proxy). The NQM stores the topology and QoS information based on the NQSL representation of the EXC3ITE network and updates from the monitoring services. The NQM obtains dynamic information about the state of the network from two sources: network management and the Local Monitoring function. Network management provides information about changes to network topology (eg: links going down) and QoS available on the links. Local Monitoring of connections provides information about whether applications are receiving the requested QoS and can be used to determine QoS available on outgoing links from a LAN. Information regarding the traffic load of non-adjacent links is obtained from all the other NQM components.

The NQM receives connection requests from the AQM, with QoS requirements and priority. The NQM creates these connections and can implement source routing and pre-emption if policy dictates. LAN proxies exchange local information such as bandwidth, delay, priority and error information when significant changes occur in the network state. From this information each NQM instance builds up a QoS picture, which in this particular simulation, only concerns the WAN links. When the NQM is requested to create a connection, it uses its QoS and topology information to

determine a high-level route and forwards a request through each LAN proxy along the path. This request may specify pre-emption if appropriate. The request is passed to the destination and returned to the source. If the selected path is blocked, the rejecting proxy may attempt an alternate path, which may trigger further policy action within the AQM. If pre-emption is required, the relevant LAN proxy will be consulted and (according to policy) it will select the best connection(s) for pre-emption. The process of pre-emption itself may require communications back to the LAN proxy originating each connection to be pre-empted. If the AQM/NQM cannot find an alternate link or other solution to satisfy the application QoS requirements, the pre-empted binding will be terminated.

3.3 Adapting to link failures

Link failure has the potential to undermine the reservation and moderation that the ACA performs. Such failure can be detected by normal network management platforms which are able to notify the ACA on each side of the network partition. The response of the ACA to link failure is to see whether some of the connections that previously used the failed link can now be re-routed. This is particularly important for high-priority connections. Connections that cannot be handled this way are terminated. If necessary, connections on other links will be preempted to make room for the high-priority connections. Note that connection re-routing is performed without informing the application and without tearing down the connection.

Some of the functionality of this protocol duplicates existing network protocol capabilities such as resource reservation of ATM and RSVP, and priorities of IPv6. In networks where ATM and RSVP are ubiquitous, the role of the NQM could be reduced. In particular, source routing may not always be available or the priority based queueing available in some routers may be required. However, even ATM and RSVP do not provide the binding pre-emption that is required. In an environment where source routing is not available, high-level routing can be performed in software (at a cost in processing power and with limits on throughput). The Defence networks that are considered here are heterogeneous and therefore require the higher level management of bindings present in the ACA. The ACA has been designed to allow the policy implementation to determine how source routing is applied. Also, source routing is serving two functions in the simulated architecture: supporting control of QoS delivered and enabling a simple algorithm to determine which bindings to preempt. For the simulation it is assumed that all bindings are being source routed.

3.4 Representation of applications and policies in the simulation model

The applications generate requests for CORBA client – server bindings with QoS associated requirements and priority according to an assigned probability distribution function. CORBA RPCs are represented as a series of TCP segments transmitted from one application to another, processed for some time and then returned as another series of TCP segments. The requests are judged successful if the TCP connection completes the transmission in the requested time. The requests are handled by the AQM as described above. Three applications were modelled: i) map/image retrieval, ii) messaging and iii) track updates for collaborative situation display. These applications are represented on each client workstation in the simulated network. In addition, there is one LAN proxy workstation in each LAN and most LANs have a server workstation. The server workstation simulates the server component of each application. The applications generate application invocations according to their assigned Probability Distribution Functions (PDF) that determine the time between generations of CORBA RPCs.

There are a number of policies in the ACA that affect the QoS received by applications. In an implementation of the ACA, these policies would be represented using the notation of [8]. The policies used in the simulation are presented here in a high-level form:

i) Perform IP source routing on all bindings in order to meet the QoS requirements, ii) Select the best available server given the QoS requirements and topology information, ii) Ensure that each track update is transferred within the specified time constraints or abandon the binding and return an error, iii) Ensure that high priority requests have precedence over lower priority requests, iv) If needed to meet the QoS requirements of a high priority binding, pre-empt a lower priority binding.

3.5 Other modelling issues

The node model of each client workstation in the simulation is depicted in Figure 3. As described above, this includes the local AQM, Local Monitoring and Policy Service which cooperates with the AQM, NQM and policy service components on the LAN proxy.

Delays introduced by processing of packets in the CORBA layer are not modelled. This is reasonable because simulations with and without the ACA will be subject to the same assumption and because communications delays in general are expected to be more significant than processing delays. However, based on the computational complexity of the routing algorithm [11] and the size of the network, the estimated route calculation time is 20 μ seconds. The Trader(Tr) and Type Manager (Tm) are modelled in a simplified manner — interactions between components of these services are not modelled as this will be the same for simulations with and without the architecture and occurs only

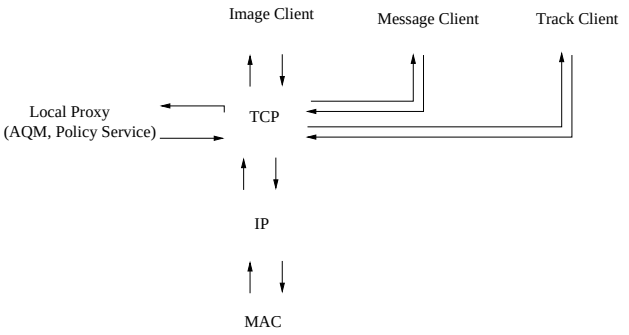

Figure 3: **Workstation node model.**

once per CORBA RPC or less.

It is assumed that in general, only a small proportion of the total traffic is of high priority and that all applications are well behaved. That is, no applications consume more bandwidth than they have been allocated. This is ensured by inserting appropriate delays between TCP packet transmissions.

4. Simulation Tests and Results

Simulation runs were performed to assess the performance of the ACA over the EXC3ITE network, where the satellite links were assumed to have an error rate of 10^{-7} and a delay of 250 msec. The required QoS and priority level are specified in each connection request. Each request was then sent to one of the servers on remote LANs selected at random. Three applications were modeled, including imagery, messaging, and track update applications. Table 1 depicts the parameters of these applications, of which the priority level is uniformly distributed with 1 percent high, 10 percent medium and 89 percent low. Note that each application waits for a request to complete before starting a new request. Therefore, "time between calls" is the time from when one request completes to when the next request is issued.

	Image	Message	Track Update
Size (KB)	1000-5000	10-125	10-150
Time between calls (Seconds)	Mean: 26.9 Variance: 64 Distribution: Normal	Mean: 26.9 Variance: 64 Distribution: Normal	Fixed at 30
Delivery time (Seconds)	50-500	300 (low) 150 (med.) 100 (high)	Fixed at 30

Table 1: Parameters of Applications in the Simulation.

Each simulation ran for 60 minutes of network time, which took up to 54 hours to run on an UltraSparc 5. The reason these simulations took so long is that the full functionality of TCP, IP, ARP and MAC layer protocols are all being simulated using the OPNET models of these protocols. Normal operations in the EXC3ITE network were simulated with and without the ACA.

The simulation results are shown in Figure 6, where the success rates of connection requests are plotted against the run count in a set of simulation runs. Five simulations were run with identical parameter settings but different seed value for the sake of randomness. The non-ACA results for the different priorities have been combined in Figure 4 since without the ACA each priority level is treated the same.

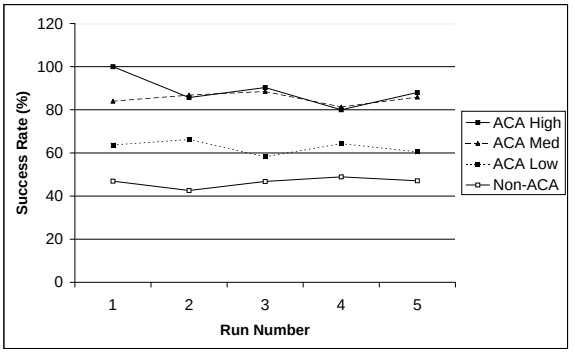

Figure 4: **Percentage of Successful Connection Requests.**

The results show that the QoS requirements of high and medium priority connections were more likely satisfied when the ACA was active. In the absence of the ACA, the bandwidth was shared by all incoming requests, and the QoS requirements of about half of these connections were met. The low priority connections were more likely to have their QoS requirements met, even when the ACA was active because some of the low priority requests were rejected by admission control or preemption, resulting in less traffic and lower link utilisation.

Figure 5 depicts the average link utilisations for the simulation runs with and without the ACA, demonstrating the load balancing effect of the ACA. In the presence of the ACA, the traffic was more evenly distributed over the network. The simulation results show that the ACA diverted more traffic to otherwise under-utilised satellite links, while reducing possible overload at non-satellite links.

Simulations were also run with failure of a selected link for a period of 1000 seconds to test the operational resilience in the presence of the ACA. Table 2 depicts the re-

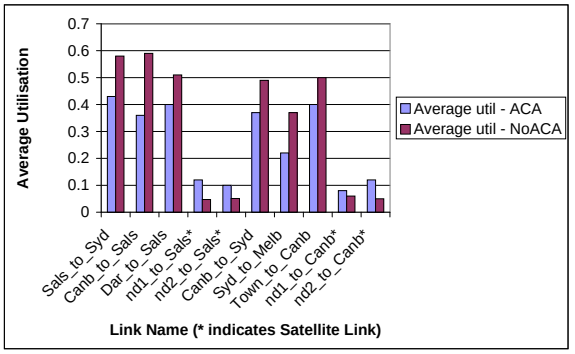

Figure 5: **Average Link Utilisation.**

sults of this run and a similar run, without the ACA being active.

Priority	With ACA	Without ACA
High	96 %	21.7 %
Medium	85.1 %	40.7 %
Low	59.8 %	39.1 %

Table 2: Connection Success Rates with Link Failures.

These results show that the ACA has achieved similar results in the presence of links failures albeit this is inferred from a small data sample.

5. Conclusions

Today, to fulfil the operational requirements of modern battlefield interoperability, defence organisations require the deployment of distributed applications. Existing system and application architectures, which typically assume LAN quality networking, do not readily accommodate these new technologies. Further problems emerge when this situation is combined with the mix of heterogeneous networking technologies and WAN links used by many large organisations. This paper has presented an Adaptive Computing Architecture which enables policy driven adaptation to changes in resource availability in heterogeneous defence networks. The design and results of a discrete event simulation of this architecture in the DSTO EXC3ITE network have been presented, focussing on network adaptation.

The results show that, for the EXC3ITE network, the architecture provides significant improvement in terms of higher priority requests meeting their QoS requirements and adaptation to link failure under heavy link utilisation. In addition, the presented architecture leads to lower link utilisation.

References

[1] J. Zinky, D. Bakken, and R. Schantz, "Overview of quality of service for distributed objects," in *IEEE Dual Use Technologies Conference*, May 1995.

[2] M. Davis and A. Downing, "Adaptable system resource management for soft real-time systems," in *Symposium on Command and Control Research and Decision Aids*, June 1994.

[3] V. Bharghavan, "Challenges and solutions to adaptive computing and seamless mobility over heterogeneous wireless networks," *International Journal on Wireless Personal Communications: Special Issue on Mobile and Wireless Networking*, March 1997.

[4] N. Yeadon, A. Mauthe, F. Garcia, and D. Hutchison, "QoS Filters: Addressing the heterogeneity gap," in *European Workshop on Interactive Distributed Multimedia Systems and Services, (IDMS 96)*, March 1996.

[5] M. Satyanarayanan, "Mobile information access," *IEEE Personal Communications*, vol. 3, February 1996.

[6] A. Schill and S. Kummel, "Design and implementation of a support platform for distributed mobile computing," *Distributed Systems Engineering Journal*, vol. 2, pp. 128–141, March 1995.

[7] S. Crawley, J. Indulska, and B. McClure, "ODP-based adaptive management of network resources in heterogeneous defence networks," in *IEEE Ninth International Workshop on Distributed Systems Operations and Management*, pp. 25–38, October 1998.

[8] M. Sloman, "Management issues for distributed services," in *IEEE Second International Workshop on Services in Distributed and Networked Environements*, pp. 52–59, 1995.

[9] B. McClure, J. Indulska, and S. Crawley, *Adaptive Computing Architecture for Heterogeneous Defence Networks*. University of Queensland, Internal Report UQ-TR-429, February 1998.

[10] K. Archie, G. Campbell, G. Cathey, A. Cohen, and R. Finn, *OPNET Modeller Example Models Manual: Protocol Models*. Mil3 Inc, 1994.

[11] J. Van Wyk, *Data structures and C programs*. Addison Wesley, 1988.