PROPOSAL

Section I.

Reference: Office of Science, Notice 99-10

Technical Topic Areas:
Management of High-speed Autonomous Networks, Application Steering, Visualization, and Control

NMVC++: Middleware for Monitoring, Visualizing, and Controlling High-speed Autonomous Networks

Prepared for:

U.S. Department of Energy
Office of Science
Grants and Contracts Division,
SC-64, 19901 Germantown Road,
Germantown, MD 20874-1290, Attention: Program Notice 99-10

Prepared by:

Douglas C. Schmidt, Associate Professor, Washington University
Jonathan S. Turner, Professor, Washington University
Eileen Kraemer, Assistant Professor, University of Georgia
Ken Wong, Senior Research Associate, Washington University
Fred Kuhns, Senior Research Associate, Washington University
David Levine, Senior Research Associate, Washington University

Points of Contact:

Technical Matters:
Douglas C. Schmidt, Director, Center for Distributed Object Computing
Dept of Computer Science, Washington University in St. Louis
(314) 935-7538, FAX (314) 935-7302, EMAIL schmidt@schmidt.wustl.edu

Administrative Matters:
Jonathan S. Turner, Director, Applied Research Laboratory
Dept of Computer Science, Washington University in St. Louis
(314) 935-6132, FAX (314) 935-7302, EMAIL jst@cs.wustl.edu

Proposal Abstract

This proposal is a three year effort aimed at the design, implementation, and demonstration of an advanced, highly scalable *differentiated services* (DiffServ) testbed and its associated network management middleware. The foundation of this testbed rests on two technological components that will be developed in this proposal: (1) a scalable, reprogrammable gigarate multiservice switching network (MSSN); and (2) an embedded, real-time, CORBA-compliant ORB (embedded TAO). The architecture of the MSSN provides hardware and software support for handling multiple signalling protocols and per flow processing through its unprecedented reprogrammable components and through its ability to steer flows through processing functions. The architecture of embedded TAO provides support for end-to-end real-time processing in a small footprint, making it ideal for constructing lightweight network management services which can be accessed through a uniform interface. This combination will lead to a highly accessible, state-of-the-art testbed that is open to a high degree of modification, extension, and experimentation.

Four other components will rest on this foundation: (1) TAO network services, a set of services for network management and resource provisioning, (2) DiSp modules, a set of modules that can be used to construct varieties of differentiated services, (3) AFS, an advanced network management agent filtering system that will provide highly scalable multi-attribute event filtering, and (4) POLKA++, a distributed visualization system that supports rapid, configurable, scalable network management displays. The integration of these components will allow users and administrators to monitor, visualize, and control the performance of their applications' QoS across multiple autonomous networks (AN)s by integrating (1) advanced networking hardware, (2) QoS-enabled middleware, and (3) scalable visualization algorithms to support differentiated services across ANs.

Moreover, our testbed will integrate and enhance technologies we have developed under previous and ongoing sponsorship. These technologies include WUGS (a scalable, gigarate ATM core), TAO (a real-time, CORBA-compliant ORB), APIC (an ATM port interconnect controller), and ANPE (an active networking processing element) – all products of the Center for Distributed Object Computing (DOC) and the Applied Research Laboratory (ARL) at Washington University in St. Louis.

1 Introduction

Next Generation Internet (NGI) applications will require advanced networks and middleware to support a wide mix of dynamically changing multimedia streams. These streams result from activities like high-bandwidth data acquisition; transparent, across-the-Internet data cache updates; and remote collaboration through interactive, multimedia teleconferencing. For instance, BaBar at SLAC [1] and CMS at CERN [2] are high-energy physics experiments that generate petabytes of testbeam data that must be filtered and stored in large-scale databases. Groups of geographically distributed scientists must collaborate to view, analyze, and interpret portions of this data and to design new experiments. The quality, and sometimes the feasibility, of these activities are directly related to the ability of the networks and middleware to provide these applications with their necessary end-to-end quality of service (QoS).

However, conventional internets that span multiple autonomous networks (AN)s cannot deliver the end-to-end QoS to NGI applications like those described above. Moreover, these applications will only be able to use the NGI effectively if middleware tools, technologies, and frameworks can be developed and deployed to *program, provision, monitor, and control* applications and services that span multiple ANs. Although researchers and commercial vendors have produced network elements that offer both high performance and have some QoS features, capabilities for programming, provisioning, monitoring, and controlling end-to-end QoS across multiple ANs are only beginning to be addressed.

It is crucial, therefore, that next-generation networks and middleware support: (1) a diverse set of end-to-end QoS requirements, (2) dynamic functionality and (re)configurability, and (3) a uniform resource, service, and control access interface to heterogeneous hosts and autonomous networks. Only such networks and middleware can be *general* enough to support stringent NGI application bandwidth, delay, and jitter requirements; *adaptable* enough to react to changing user demands and technological evolution; and *stable* enough for the development of significant amounts of advanced network and middleware services.

This proposal is a three year effort aimed at the design, implementation, and demonstration of a highly scalable *differentiated services* (DiffServ) testbed and its associated network management middleware. This integrated system will allow users and administrators to monitor, visualize, and control the performance of their applications' QoS across multiple autonomous networks (AN)s by integrating (1) advanced networking hardware, (2) QoS-enabled middleware, and

(3) scalable visualization algorithms to support differentiated services across ANs.

Two key goals of our *network monitoring, visualization, and control* (NMVC++) system are to (1) ensure adequate end-to-end QoS to NGI applications while (2) maintaining high levels of network and endsystem resource utilization. In addition, NMVC++ will allow network administrators to calibrate and fine-tune network and application parameters in real-time according to observed traffic patterns, thereby ensuring that network elements and middleware services adapt efficiently to dynamically changing conditions. Figure 1 illustrates the key components in our proposed NMVC++ architecture, which are described in Section 2.

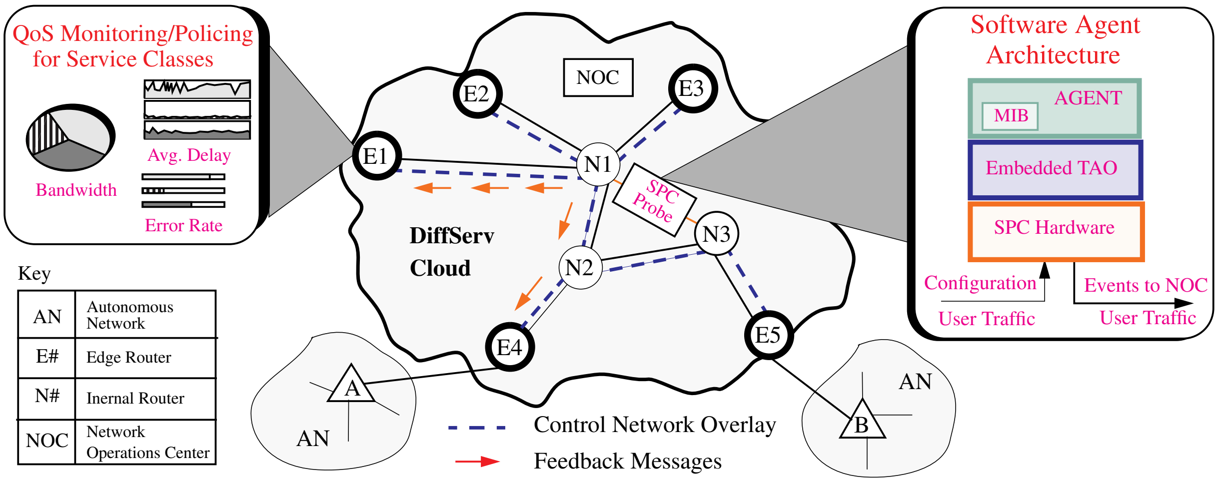

Figure 1: Components in the NMVC++ Architecture

2 Method of Accomplishment and Expected Results

This section outlines the method by which we will leverage and significantly enhance our existing technologies to produce an integrated NMVC++ system that will have a significant impact on the DoE community. Our project deliverables will include a tested, useable NMVC++ system prototype, demonstration applications, along with technical papers and documentation. Our team has a long history of successfully transferring our state-of-the-art research into tools and products that are used widely in industrial, governmental, and academic projects, such as high-energy physics experiments like BaBar at SLAC [1] and CMS at CERN [2], the HLA/RTI distributed interactive simulation middleware, and avionics mission computing [3], satellite control systems [4], and electronic medical imaging systems [5, 6]. We will leverage our extensive user-base [7] and augment it with a development and deployment process that emphasizes rapid prototyping, periodic demonstrations and course correction evaluations, and coordination with other parts of NGI and other key DoE project participants.

2.1 Leveraged Technology

To ensure the success of the NMVC++ project, we will leverage our existing expertise and ongoing research results in high-speed networking [8, 9, 10], middleware [11, 12, 13], and network management [14, 4] to produce a cost-effective, high-payoff testbed built upon the following components developed at Washington University (WU)[1] to support network monitoring, visualization, and control:

NMVC (a testbed for network monitoring, visualization, and control of gigabit networks): This testbed is a prototype implementation of a highly scalable network monitoring, visualization, and control system within a single AN [14]. At the hardware level are gigabit IP routers constructed from the Washington University Gigabit Switching (WUGS) [15]

[1]Work on these technologies is supported by the government (NSF and DARPA) and industry (Bellcore, Boeing, Hughes, Intel, Lucent, Microsoft, Motorola, SAIC, Siemens, and Sprint).

technology [16]. The testbed uses a Smart Port Card (SPC)-based network probe to monitor traffic at gigabit rates and to process IP packets [17, 9].

TAO (a CORBA-compliant Object Request Broker (ORB)): TAO is an open-source, standards-based, high-performance, real-time ORB endsystem designed to address the QoS requirements of NGI applications [11]. It is the first ORB to support end-to-end QoS guarantees over ATM/IP networks [16, 13]. TAO contains many optimizations for key sources of overhead like context switching, dynamic allocation, data copying, demultiplexing, and pluggable protocols that make it suitable as a middleware interface to QoS services [18, 12, 19, 13].

ANN (an active networking programming framework): The **ANN** framework provides the flexibility of Active Networking technology for automatic, rapid protocol deployment and application-specific data processing and forwarding at gigabit rates. The ANN environment allows administrators and applications to control networking nodes and packet processing and forwarding [20].

Section 2.2 describes the innovations that we are developing to significantly enhance the existing components and capabilities described above.

2.2 System Overview

Our proposed NMVC++ system consists of four components: (1) Embedded TAO network services; (2) DiSp, an architecture for differentiated services; (3) AFS, an advanced network management agent filtering system; and (4) POLKA++, a distributed visualization system that supports rapid, configurable, scalable network management displays. These components will be implemented using two enabling technologies: (1) MSSN (Multiservice Switching Network) and (2) embedded TAO (The ACE ORB). As described below, the MSSN is a reprogrammable gigarate multiservice switching network, and Embedded TAO is a real-time, CORBA-compliant ORB that provides support for end-to-end, real-time processing in a small footprint.

Figure 2 shows the main hardware components of the MSSN. The MSSN supports communication using a variety of

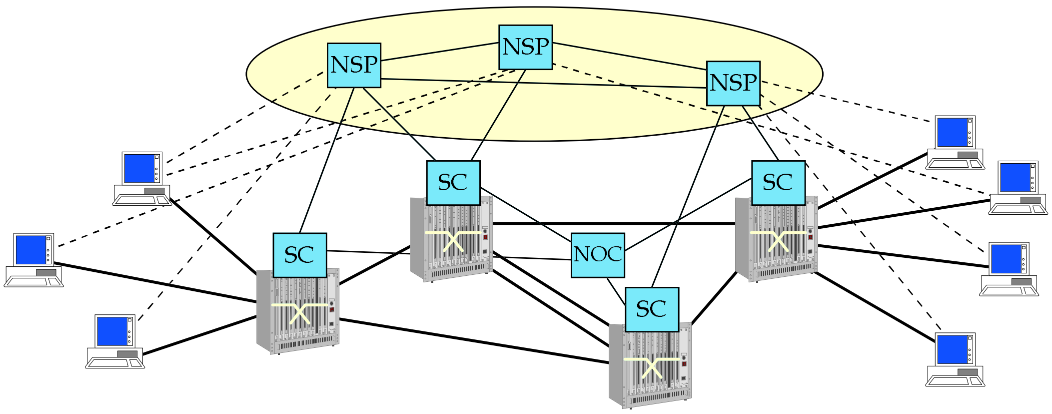

Figure 2: Multiservices Network

network services (*e.g.*, IPv4, IPv6, ATM, Active IP). Hosts may simply send conventional IP packets and/or request special services (bandwidth reservation, QoS routing, encryption) from Network Service Processors (NSPs). NSPs request services from particular routing switches through the Switches' Switch Controllers (SC). These requests are made using a generic, hardware-independent control interface. Network Operations Centers (NOCs) enable human managers to monitor systems and deal with exceptional conditions.

2.2.1 Embedded TAO

TAO is an open-source, standards-based, high-performance, real-time ORB endsystem that supports applications with deterministic and statistical QoS requirements, as well as "best-effort" requirements, and is the first ORB to support end-to-end QoS guarantees over ATM/IP networks [16]. TAO's features and optimizations include an ORB Core that minimizes context switching, synchronization, dynamic memory allocation, and data movement [18]; a highly-scalable Object Adapter that demultiplexes requests in constant-time [12]; an optimizing IDL compiler [19]; real-time I/O subsystem [13], and a global resource allocation and scheduling framework [11].

In this project, we will develop a network-embedded version of our TAO ORB that provides a uniform access interface to network controls, services, and resources. Embedded TAO will support signaling, admission control, and network management, while still providing uniform access across ANs. Unlike related middleware efforts [21, 22, 23, 24, 25], however, embedded TAO will be a minimal footprint [26] ORB that can be installed into switches, routers, and other network element controllers to improve the performance and control of services across ANs.

The real-time features of embedded TAO will conform to the forthcoming Real-time CORBA standard [27]. These features will decrease fault detection/resolution latency and improve the predictability of network/system agents by supporting end-to-end priorities across heterogeneous hosts over autonomous networks.

2.2.2 DiSp

DiSp (Differentiated Services over IP) (see Figure 1) supports QoS in a scalable manner to support service discrimination across autonomous networks (ANs) [28]. DiffServ, in general, is still in its infancy and has not yet matured into a service framework that can satisfy diverse application requirements. Some issues to be resolved include: (1) precise service class definitions, (2) admission control policies, (3) strategies for policing and shaping of aggregate flows and (4) congestion handling mechanisms.

We will develop DiffServ modules for both DiSp and IETF DiffServ [29] and evaluate both approaches on our Multi-Service Switching Network testbed. Moreover, we will build a set of modules and use them to construct various DiffServ services. Embedded TAO (Task 1) will be used to provide a uniform access interface to DiSp's mechanisms, such as (1) lightweight signaling protocols to facilitate third-party resource negotiations and (2) admission control algorithms.

DiSp builds on the basic DiffServ concept of *flow aggregation* to provide user-controlled traffic services and facilitates the construction of end-to-end services through simplified service agreements at network boundaries. It supports two fundamental service enhancements: (1) receiver subscriptions, and (2) statistical bandwidth guarantees. DiSp has four key features: (1) three service classes: real-time (RT), statistical-bandwidth (SB) and best-effort (BE), (2) mechanisms for policing and shaping aggregate flows, (3) service guarantees on a per-flow basis even though RT flows are treated in an aggregated manner, (4) efficient monitoring mechanisms that can provide accurate feedback for congestion control and overall network management.

In keeping with the DiffServ ideals, DiSp does not maintain per-flow state information in ANY router and uses simple priority scheduling mechanisms among the three classes. However, DiSp does have a simple congestion control facilitates. Thus, DiSp is both *efficient* (*e.g.*, it will provide per-flow service guarantees without per-flow router state), and *simple* (*e.g.*, it will push QoS management to autonomous network edges). Moreover, since DiSp views IntServ and DiffServ as complementary models, it supports and facilitates their seamless integration.

2.2.3 AFS

The Agent Filtering System (AFS) provides extensible multi-attribute event filtering for highly scalable data/event collection. Using embedded TAO, AFS can be remotely configured and installed together with probes on network nodes.

Agents achieve scalability by flexible, high-performance event filtering mechanisms. These mechanisms use a dynamic trie-based filter composition technique [30] that reduces the work required to classify E events of size L through N event filters from $O(E \times L \times N)$ to $O(E \times L)$, which is the minimum possible [31]. Administrators and management applications can strategically install filters within agents to route events of interest to remote nodes (such as NOCs or other agents). Thus, multiple agents and filters can be composed and/or arranged hierarchically to reduce unnecessary network traffic and enhance the scalability of event notification in a large-scale network.

2.2.4 POLKA++

The execution of large networks is difficult to observe, understand, and control. Our proposal is based on an alternative approach: *distributed visualization using graphical animation.* Graphical animation taps into well-developed human abilities to detect patterns and recognize anomalies in those patterns. Thus, visualization can help administrators detect recurring traffic patterns, determine appropriate quality of service parameters, and fine tune network performance.

To be an effective tool for managing large-scale, high-speed networks, visualizations must exhibit scalability and celerity. Scalability is achieved through the hierarchical presentation of views at various levels of abstraction and through support for interactive navigation through these views. Clarity or speed is achieved through careful design of displays and by semantic filtering of animation events resulting in timely display that approximates the current global state of the network.

The features of POLKA++ will include: (1) algorithms for efficient online ordering of events, to ensure that the event stream arriving at the visualization is consistent, (2) a view choreographer through which users can specify mappings from network events to higher-level events and from both low-level and high-level events to changes in graphical views, and (3) a visualization system and visualizations, through which network administrators may monitor network performance, investigate patterns or anomalies, and interact with displays to alter parameters and policies

POLKA++ will help network users reconfigure distributed applications and helps administrators detect traffic patterns, make decisions regarding QoS parameters, and fine-tune network performance. Displays and interactions can be customized by level of expertise, administrative authority, and features of concern. POLKA++'s *View Choreographer* will provide user-specified mappings from network events to higher-level events and to changes in graphical views, will support the creation of custom displays and interactions, and will perform customizable filtering on the stream of events at the display system to provide configurable, scalable displays. POLKA++'s visualizations will support concurrent animations, featuring color, real-time, smooth animations.

2.2.5 Distributed Simulation Application Demonstration and Testbed Evaluation

Our NMVC++ testbed will support QoS across ANs and improve programmability and extensibility through standardized CORBA middleware and pluggable protocols, as illustrated in Figure 1. To validate the testbed, we will conduct systematic performance benchmarks of NMVC++ hardware and software components using synthesized traffic patterns and actual applications. Synthetic traffic experiments will help us explore portions of the design space that are unexercised by existing applications. Actual application results will help direct our focus to representative problem areas needing additional attention.

Our demonstration applications will support groups of scientists collaborating over high-speed networks, downloading large multimedia data sets, interacting remotely using visual displays, spawning analysis computations dynamically, monitoring and controlling long-running computations, and updating/refining the data in a database, as well a serve as a vehicle for empirical testing and evaluation. This demonstration will verify that our testbed can support an actual application and will allow us to verify the efficiency and scalability of our NMVC++ components.

2.3 Summary of NMVC++ Benefits

For network users and NGI applications, the benefits of NMVC++ will include: (1) validating the levels of QoS received by various classes of applications and (2) automatically adapting configurations of distributed applications and distributed systems to maximize their performance (*e.g.*, time to completion) and achieve their requirements (*e.g.*, resilience to network failures).

For network service providers and administrators, the benefits of NMVC++ will include: (1) real-time operational monitoring of large-scale, high-speed networks; (2) rapid fault detection and system resource reconfiguration to repair the cause and effects of faults; (3) real-time traffic pattern adaptation by tuning network control parameters at multiple protocol layers; and (4) visual validation and assurance that applications are receiving their QoS guarantees.

3 Related Research

3.1 High-Speed Network Infrastructures

The last decade has seen the development of a number of high-performance network testbeds, intended to support networking research. Currently, the National Science Foundation's vBNS [32] network offers the largest example of such a high-speed network testbed. The vBNS now spans several tens of sites (including Washington University) and provides an effective mechanism for delivering higher bandwidth to application researchers. Its value to networking researchers has been more limited, however, since the network provides only standard IP datagram services and many aspects of the networks operation are largely off-limits to networking researchers. This makes vBNS unsuitable as an environment in which to develop, deploy, and evaluate experimental services. It is also relatively low performance by emerging standards, *e.g.*, for cost reasons, most institutions access to the network is limited to 45 Mbps and only a small handful have connections of more than 150 Mbps.

CAIRN [33] is a national testbed for research in Internet protocols and technology. CAIRN now connects dozens of sites at speeds from 1.5 Mbps up to 150 Mbps. CAIRN is built using a common experimental routing platform, which allows researchers to experiment with new network services by developing and loading new software for the router. The current routing platform is limited in performance since it is built around a single general-purpose computer with multiple network interface cards. The architecture is comparable to commercial routers of the mid-1980s, but lags far behind the current state-of-the-art. This makes it difficult for CAIRN researchers to tackle many of the performance-related challenges that are at the cutting edge of current research.

The National Transparent Optical Network (NTON) [34] is a network testbed now in the planning stages that will span a number of institutions along the west coast of the U.S. NTON planners are attempting to structure the testbed to support both applications researchers and networking researchers operating at different levels of the network hierarchy. If they can in fact develop mechanisms to enable network and application researchers to effectively co-exist in a common testbed infrastructure, network researchers will have the opportunity to develop and deploy new network services and evaluate them in the context of a high-performance testbed with a substantial user population. Realizing this vision will require inclusion of experimental network equipment that is open to modification and extension.

3.2 Network Management

After studying several commercially available SNMP and CMIP based network monitoring and management systems/tools, we find that most are not suitable for the monitoring, control and visualization of large high-speed ATM and other networks that are the target of this project. While these tools provide basic features like network discovery and MIB browsing, they have many weaknesses. Most of these tools allow only *status polling*, which helps detect network failures. However, failure detection is a small component of today's network management. There are some tools that do both status and performance polling. However, they keep these two separate, and performance polling is done more often than status polling. This means that its takes longer to detect failures. Also, scalability is a major weakness of these tools. That is, they will not work or work well with large networks and with high speed networks [35, 36]. For example, most tools redundantly poll devices that are part of more than one network map. This generates excessive management traffic when administering large-scale MAN and WAN networks. For some of the tools, notably HP OpenView [37] and IBM NetView/6000 [38], the architecture is geared towards managing discrete entities with no knowledge of the overall network environment.

With few exceptions (such as Cabletron's SPECTRUM and OSI NetExpert), the tools we surveyed did not handle correlations between network entities. Thus, many alarms and traps can potentially be generated due to a single point of failure. SPECTRUM is built around an artificial intelligence engine, called the Inductive Modeling Technology (IMT), which together with its object- oriented design, permits SPECTRUM to understand dependencies. The SPECTRUM server allows both server initiated and operator initiated device polling. Further, no redundant monitoring of devices occurs when multiple clients are monitoring the same device on different maps. However, the package has numerous reported performance inefficiencies, is much slower than other commercially available tools, and cannot scale to networks operating at Gb/s.

The ATTILA [39] project tries to measure and monitor traffic on an ATM/SONET link using a network probe. However, a SUN SparcStation10 is required for each network probe, and the data collected is at the cell/link level only. Further, no attempt is made to get higher level statistics such as end-to-end throughput or Quality of Service for different TCP connections.

The Center for Telecommunication Research at Columbia University has proposed an Integrated Reference Model (IRM) [40] for the management and control architecture of gigabit networks based on the OSI CMISE. The core of CTR IRM is a centralized network telebase that contains extensive information about network operations. However, the structure of the database and issues involved in data collection, event-ordering, and feedback control are not adequately addressed. The system runs on a WAN emulator with a software emulation of a subset of IRM on a supercomputer.

Empirical results based on actual networks have not been obtained. Also, the thrust seems to be on developing a Virtual Reality GUI rather than on real-time measurement, visualization, and control [35, 36].

The group at LBNL [41] recognized the problem that the applications on high speed networks typically get only a small fraction of the network bandwidth. To analyze the problem and pin-point the bottle-necks, they have developed a logging facility and have used it for a distributed application running on their testbed ATM network. This has been very useful in identifying performance bottlenecks and understanding interactions of protocols at various layers. However, while events are logged in real-time, all analysis is done offline, that is, after all the logging has terminated. This does not give much scope for online feedback control. There is no auto-detection of faults or human-in-the-loop, an issue which our proposed scheme addresses fully. The design is tightly coupled to and oriented towards their chief application - the Distributed Parallel Storage Server [41]. Also, except for the logging of events, only commercially available monitoring systems are used which can be inadequate. Also, there is no online event-ordering and visualization is limited to that provided by the vendor of the Network Monitoring System.

3.3 Differentiated Services

The current Internet supports only best-effort service, irrespective of user expectations. Moreover, application heterogeneity dictates that there be service heterogeneity and service differentiation. The Integrated Services (IntServ) architecture was developed to meet these QoS needs and resulted in the development of the RSVP protocol for signaling requirements to the network. Critics have pointed out that IntServ suffers from *scalability*, and *complexity* problems because it requires per-flow state at the routers, fast packet classification, and per-flow packet scheduling. Moreover, IntServ lacks a pricing structure. All of these deficiencies have made it difficult to deploy on a wide scale.

These deficiencies have led to the beginnings of an alternative QoS delivery model known as *Differentiated Services* (DiffServ) [29]. To address the scalability issue, DiffServ aggregates flows into service classes rather than maintaining per flow state. Moreover, QoS requirements are specified out-of-band removing the necessity for a signaling protocol such as RSVP. Packet classification is based on the setting of a few bits in the IP header. Flow aggregation in DiffServ has several beneficial consequences. First, DiffServ routers map a large number of flows to a small number of per-hop behaviors which contain information about the precedence or priority level of the packet. Thus, instead of every router having to manage individual flows, only the edge routers need to be concerned with QoS. Second, aggregation facilitates the construction of "end-to-end" services by linking multiple autonomous domains together using simplified service agreements at the boundaries of the domains.

The primary DiffServ effort is the IETF Differentiated Services Working Group. Their architecture is summarized in RFC 2475 [29]. is based on a simple model in which traffic is mapped to a small number of classes (flow aggregates) and conditioned at the boundaries of an autonomous network. IP packets are marked using a field in the IP header. At the edge of each network, packets are classified and marked to receive a certain per-hop forwarding behavior within the network. Per-hop behaviors (PHBs) are implemented by some combination of buffer management and packet scheduling mechanisms within nodes along the path of a packet.

The bulk of the work in dealing with service differentiation is done at the boundaries of the network. Internal nodes, for the most part, are responsible for forwarding packets according to the PHB of the packet. Although the Working Group will be standardizing only a small number of PHBs, they recognize the need for experimentation with other PHBs. The motivation for the relatively small and simple PHBs being developed is to define simple building blocks upon which a variety of differentiated services can be implemented and evaluated.

The IETF model of differentiated services is one of a variety of models for service differentiation. Some alternative models include Integrated Services (IntServ) and label switching. IntServ is defined in RFC 1633 [42] and is intended to provide QoS transport over IP internets. IntServ uses RSVP (Resource ReSerVation Protocol) for signalling resource requirements. IntServ requires flow classification and forwarding state for each active flow at each router along each QoS path. DiffServ is an attempt to: (1) Reduce the state through aggregation and (2) Eliminate the need for in-band resource reservations and therefore a signalling protocol.

Label switching (or virtual circuit switching) is a generic term refering to the use of a forwarding label in each packet header that indexes into a forwarding table at the next-hop. The index is used to lookup the per-hop forwarding behavior and the replacement label at each hop. Examples include ATM and MPLS (Multiprotocol Label Switching). In this approach, QoS state is established and maintained at each hop along a path.

3.4 Event Filtering

Work on event filtering spans a number of domains, including distributed systems [43, 44], network management and network monitoring [37, 45, 46, 47, 48], user-level communication protocols [49, 50, 51, 52], and active databases [53, 54, 55]. This section outlines relevant related work on event filtering.

Distributed systems:

• **Isis News:** Isis [44] supports event filtering in its *Reliable Distributed Objects* (RDO) News service [43]. In Isis RDO News, filtering is performed at the destinations since all consumers in a process group receive all events sent by suppliers. Therefore, the RDO News filtering architecture will not scale to accommodate a large numbers of consumers that possess complex filtering requirements. In contrast, the event filtering mechanism proposed in our effort allows filtering at the agents, which supports much greater scalability in gigabit network environments.

Network management:

• **HP OpenView:** HP OpenView [37] provides an implementation of the ISO OSI event report management services [47]. HP OpenView filtering allows *Event Forwarding Discriminators* (EFDs) to be installed on remote agents in a network. An EFD contains an expression that filters events based on their type, value, generation time, and/or frequency. EFD filter expressions are described using GDMO, which is a schema for defining managed objects in a network. OpenView's implementation of OSI EFDs is inadequate for high-performance event filtering. It suffers from a highly inefficient process architecture that requires 4 context switches and 3 interprocess communication (IPC) exchanges to receive, filter, and deliver each event end-to-end. In contrast, the event filtering mechanism proposed in our effort is tightly integrated with the network probes and the agent. This reduces the overhead to 1 context switch and 1 IPC exchange per event, which is the minimum possible.

Packet filters and packet classifiers: The majority of related experimental research has focused upon various types of *packet filters* (also known as *packet classifiers* [52]). Packet filters were developed originally to support efficient demultiplexing for user-level implementations of network protocols [49]. In addition, they have been used to monitor traffic unobtrusively on promiscuous-mode networks [46, 45, 48].

The CMU/Stanford Packet Filter (CSPF) [49] and the Berkeley Packet Filter (BPF) [50] are two influential first-generation packet filter tools. Second-generation tools (such as the Mach Packet Filter (MPF) [51] and the PathFinder packet classifier [52]) enhance the scalability of first-generation tools by enabling the composition of multiple filters.

• **CSPF and BPF:** CSPF is a stack-based packet filter that operates on binary instructions (*e.g.,* `pop` and `push`). CSPF uses boolean expressions, along with a tree model, to configure its filtering engine. BPF extends and enhances CSPF to overcome certain performance limitations with CSPF. BPF uses a register-based execution model and and acyclic control graph, instead of CSPF's stack-based language and tree graph, respectively. In addition, BPF extends CSPF by handling packet fragmentation, delivery of out-of-order fragments, and variable-length headers.

Both CSPF and BPF maintain a list of configured packet filters (the list maybe reorganized dynamically to move frequently accessed filters to the front of the list). This approach works well if there are relatively few packet filters, or if only a small number of consumers are active simultaneously. When there are hundreds of filters and/or hundreds of consumers, however, this approach does not scale up since the time required to filter packets grows linearly with the number of filters. In contrast, the event filtering mechanism proposed in our effort uses a dynamic trie-based filter fusion technique that reduces the work required to classify E events of size L through N event filters from $O(E \times L \times N)$ to $O(E \times L)$, which is the minimum possible.

• **MPF:** The Mach Packet Filter (MPF) is designed to support user-level implementations [56] of layered protocol stacks (such as TCP/IP). In this context, composing multiple packet filters helps to reduce demultiplexing overhead. Often, packets destined for separate connections on an endsystem contain a common prefix, followed by variation at a single point in the packet. For example, active TCP connections on an endsystem will have many IP header and TCP header fields in common (such as source and destination IP addresses). In this case, the only demultiplexing field that varies among arriving packets is the TCP destination port number. Thus, MPF associates a hash table with this field to demultiplex packets to different endpoints efficiently.

The MPF composition technique assumes the existence of a common prefix across protocol headers. This assumption enables low latency setup and removal of composite packet filters. However, this particular optimization strategy does not generalize to compose more complex filters. In contrast, the event filtering mechanism proposed in our effort utilize advanced filter optimization techniques that allow complex filter fusion.

- **PathFinder:** The PathFinder tool described in [52] presents a more general technique for coalescing filters with common prefixes. PathFinder is a packet classifier that combines software and hardware techniques to optimize the composition of complex filtering patterns. The software portion of PathFinder builds a directed-acyclic graph (DAG) interpreter based upon patterns specified by a user-defined declarative syntax. The PathFinder interpreter matches fields of incoming packets using information stored in the DAG.

PathFinder uses several novel features to support high-performance packet classification. For example, it adaptively reorders the DAG when pattern matches a composite pattern. This reordering process synthesizes and caches a new customized representation that reduces the number of comparisons for subsequent packets matching the pattern. In addition, PathFinder also offloads a portion of the packet classification logic into hardware, which may run on a network adapter board.

One limitation with PathFinder is the relatively high latency required to setup/remove patterns from the DAG (since the DAG must be searched from the root to the leaves when inserting/deleting a new pattern). In addition, the software implementation of the DAG uses an interpreter, rather than a compiler. In contrast, the event filtering mechanism proposed in our effort utilize "just-in-time" compilation technology to transform JAVA bytecodes (which are interpreted) into instructions that can execute directly on the native agent hardware.

3.5 Visualization

Visualization tools for parallel and distributed systems assist developers in debugging, evaluating, and tuning parallel and distributed programs by presenting displays of communication behavior, utilization, and computation. However, none of the existing tools meet the needs of this project. Paragraph [57] is a portable trace analysis and visualization package for message-passing programs, producing displays of processor utilization, communication, etc. However, it relies on the production of PICL [58] traces, does not support the creation of new views, and is designed for the visualization of application programs rather than networks. Similarly, the Pablo project [59] addresses source code instrumentation, data capture, and data analysis for massively parallel systems, but does not adequately address the needs of high-speed networks.

The PARADE project [60] is composed of the POLKA animation toolkit [61], designed to support the visualization of parallel and distributed systems, an Animation Choreographer [62], and various libraries of predefined animation views for particular architectures and programming paradigms such as PVM, Conch, parallel C threads, and High-Performance FORTRAN. We will extend this approach to the online visualizations of network status and behavior.

Falcon is a system for on-line monitoring and steering of large-scale parallel programs. The purpose of such interactive steering is to improve the program's performance or to affect its execution behavior. The Falcon system is composed of an application-specific on-line monitoring system, an interactive steering mechanism, and a graphical display system. However, Falcon is designed for the monitoring and steering of threads-based application programs [63].

4 Research Plan

This proposed project is relatively large and broad in its scope. This section describes the five tasks that comprise the proposed work.

4.1 Task 1 – Embedded TAO

In our proposed effort we will create a version of TAO that conforms to the forthcoming Real-time CORBA [27] and Minimum CORBA [26] standards so it can be embedded in a Fore ATM switch to improve the flexibility and control of applications and services across high-speed networks without sacrificing performance. Our target switch control environment includes support for the GSMP as defined in the IETF RFCs [64, 65]. We will also follow closely the GSMP IETF working group. We will use the resulting GSMP middleware to build robust, efficient, and scalable ATM signaling applications. In particular, end-user applications can use our GSMP framework to transparently setup end-to-end flows with associated Quality of Service (QoS) guarantees.

The result of this effort will be middleware based ATM switch control framework based on GSMP and CORBA. Within our GSMP middleware framework we will have ORBs (1) embedded on the switches and (2) resident on the client hosts and the controllers (*e.g.* switch controllers). The broader goal of our work is to provide a switch/router control framework that can provision end-to-end QoS guarantees for real-time and high-bandwidth applications running over high-speed networks, such as ATM, Gigabit Ethernet, and high-speed IP routers.

Figure 3 show a comparison of a conventional GSMP implementation and one that uses TAO middleware embedded in the switch. Embedding the ORB is a more powerful and flexible configuration since the middleware can be used to control the switch throughout a network or set of autonomous networks.

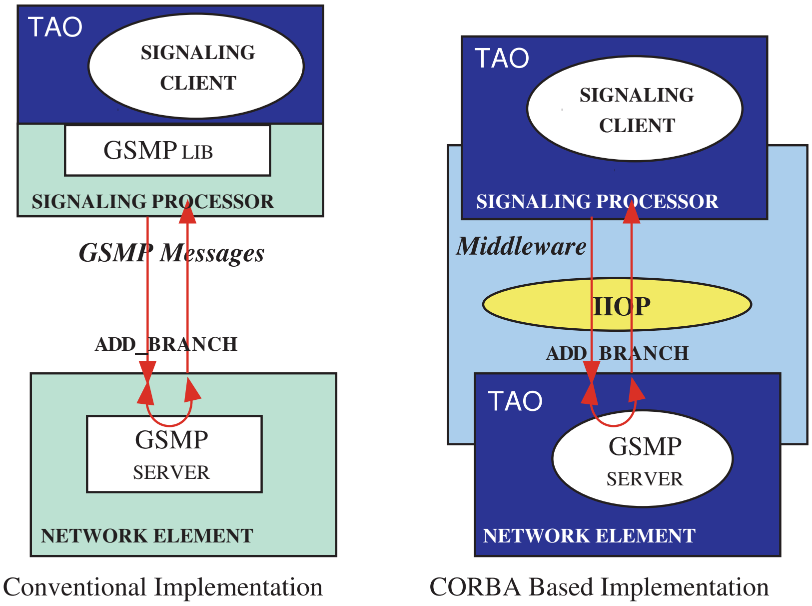

Figure 3: Alternative Signaling Configurations

By embedding the ORB in the switch and exposing the GSMP interface we are able to take advantage of the inherent semantics and flexibility of CORBA. There is the immediate advantage of simplifying the programming, provisioning, monitoring, and control of ATM signaling applications. Plus the applications and services developed using our ORB-enabled GSMP middleware will yield more modular, extensible, and standard solutions that can be reused across multiple projects and application families. For example, new switch control functionality or interfaces can be added without exposing application developers to underlying details or complexities. Using this approach, we can add support for new signaling standards or enhanced features while maintaining a consistent interface to developers.

Figure 4 shows another view of the embedded TAO-based GSMP server configuration where multiple signaling processors communicate with a network switch. In this case all control and management communication occurs within the context of the ORB. An endstation will request that a connection be established from between itself and another endstation. The signaling processors will process the request, determine a route and request each switch along the route to allocate necessary resources. The signaling processor communicates with with a switch controller using the exported GSMP based interfaces.

4.2 Task 2 – DiSp

We will implement DiffServ functions (including DiSp and IETF) using our MultiService Switching Network (MSSN) and provide configuration and management access to these functions through the embedded TAO middleware. Task 4 (described later) will evaluate the DiffServ architectures in an application setting. Heterogeneous service guarantees accommodate heterogeneous user expectations and permit differentiated service pricing. Supporting service guarantees (even statistical ones) requires that functions such as resource reservation, flow discrimination, and traffic shaping, policing, and marking be implemented within network components. At each router, packets are steered through a series

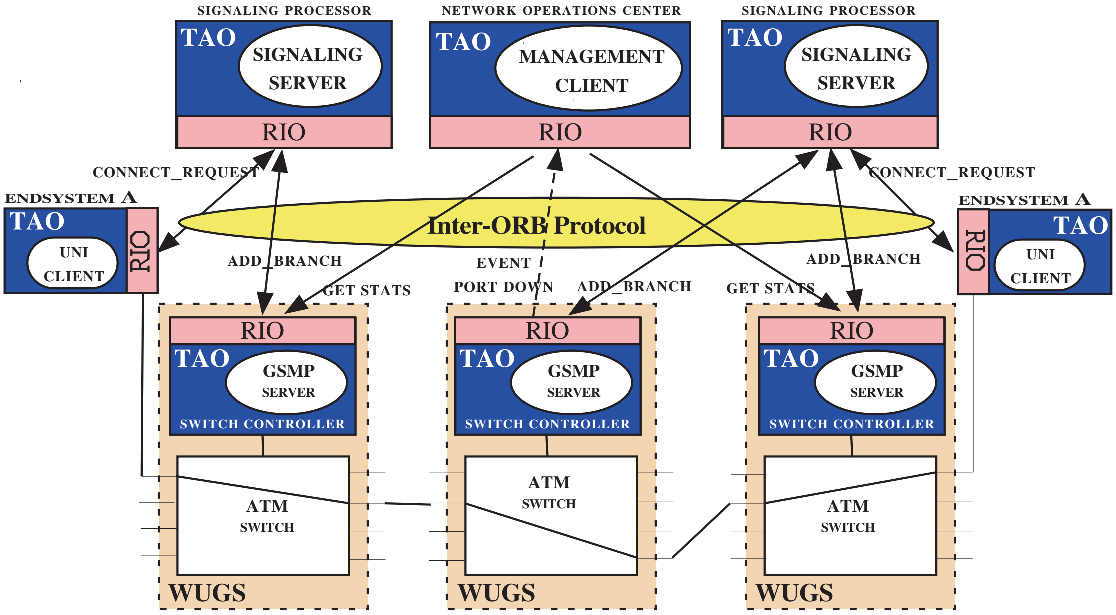

Figure 4: Applying Embedded TAO to GSMP Signaling and Control

of these functions based on their service guarantees. This section describes the functions in our DiffServ architecture, how these functions can be implemented in the MultiService Switching Network (MSSN), and how other DiffServs can be implemented.

A variety of DiffServ architectures can be constructed by defining different functions and function combinations[2]. A reference model for DiSp, our DiffServ architecture, is shown in Figure 5.

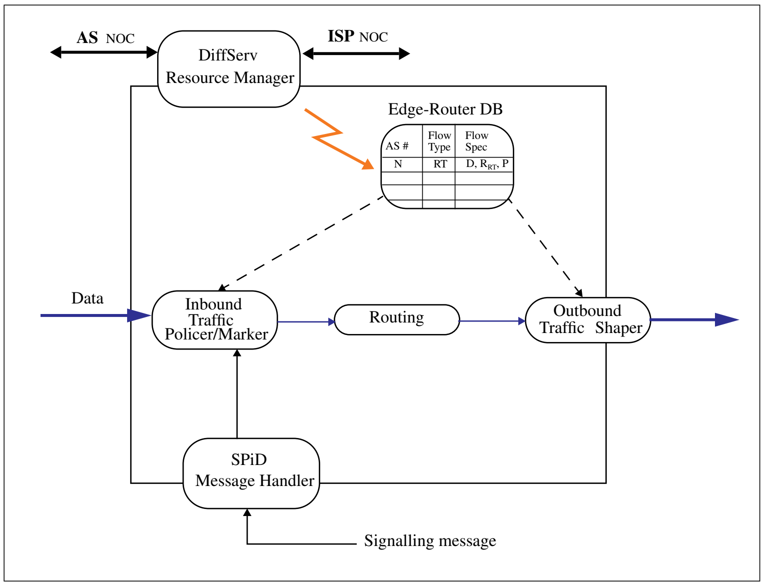

Figure 5: Edge-router Internals

We plan to design and prototype dynamically loadable kernel DiSp modules for edge routers and the Network Operations Center (NOC):

Admission Control Module: The NOC is responsible for admission control and network management. Users negotiate with the NOC for QoS-based connections. The NOC uses a database to determine the admissibility of new QoS

[2] We are using the term DiffServ loosely to mean any service architecture that supports differentiated service. IntServ may be thought of as one extreme case of DiffServ in which there is per flow state for every flow. At the other extreme is an architecture that recognizes only a few aggregate flows and no individual flows.

flows and updates the database when a new flow is admitted. DiSp admission control determines if there are sufficient resources to deliver guaranteed services. Since DiSp does not use explicit link partitioning or complex per-flow scheduling algorithms at internal routers, admission control provides the only means whereby we can provide bandwidth and delay guarantees. But, again, DiSp does this without per-flow state at the routers and uses only simple priority scheduling mechanisms among the three classes (RT: Real-Time, SB: Statistical Bandwidth, and BE: Best Effort).

Golestani showed that even if a flow is paced at the edges of a network, packets in the flow can cluster together inside the network to form bursts [66]. DiSp's admission control algorithm takes into account the worst case delay that can be suffered by an RT flow.

Resource allocation for SB flows in DiSp is a difficult problem. Since users negotiate only an *aggregate* profile for SB flows without any *a priori* indication of individual flow profiles or destinations, it is impossible to construct fixed paths and reserve resources for individual flows the form the SB aggregate. Naive solutions (worst-case fixed paths) are inefficient. We will develop efficient heuristics for resource allocation and admission of SB flows. This may be quite reasonable in situations (*e.g.*, Virtual Private Networks) where the number of participating remote sites (and therefore egress routers) is small and reasonably static. In cases with large fan-out, we plan to explore a measurement-based approach where each link in the network is monitored for current bandwidth usage and long-term average bandwidth utilization per-link is computed.

Flow Policing/Shaping Modules: DiSp edge-routers are responsible for policing and shaping *aggregated* RT and SB flows and marking the packets accordingly. For each input link in the edge-router, a modified token bucket scheme is used to police real-time flows. High bandwidth flows are policed using a fixed size queue and a pacer. Each output link has three output queues, one for each service class, which are served in strict priority order: *i.e.*, (1) RT, (2) SB, and (3) BE.

Each (ingress) edge-router of an ISP stores the individual profiles of RT flows and aggregate profiles of SB flows from each connected AS. Although policing of real-time flows from a particular AS will be done in an aggregated manner, the edge-router may have to adjust the policer to selectively drop packets in accordance with acceptable loss rates for individual RT flows when the guarantees cannot be met because of hardware failures.

Packet Classification and Forwarding Modules: Routers are responsible for scheduling packets in strict priority among the service classes. In order to properly steer the packets in the correct direction, it must classify each packet according to its service class.

Link Monitoring and Feedback Modules: Internal routers also monitor best effort traffic and generate explicit congestion notification messages during congestion.

IETF DiffServ has many of the same functions as DiSp. For example, it has traffic classifiers and conditioners (meters shapers, markers and droppers) although the form of some of these functions may be different. In IETF DiffServ, the notion of per-hop behaviors (PHBs) define the forwarding treatment (behavior) to be given to packets. Two such PHBs are Assured Forwarding and Expedited Forwarding [67, 68].

We will use our MSSN to implement DiffServ functions and configure and manage these functions through the embedded TAO middleware. The MSSN is a highly programmable hardware system that can be configured to support a number of networking services (*e.g.*, UNI, RSVP). A key component in the MSSN is the multiservice routing switch (MRS) shown in Figure 6 along with its switch controller (SC). The MRS consists of a scalable switch core backplane and several processing stages at each port. Each port controller (PC) contains an embedded processor and memory and provides per port control functions including IP forwarding and queueing and programmable services (*e.g.*, reliable multicast, encryption).

As packets stream through the switch ports, they can be *steered* through a wide diversity of functions (*e.g.*, DiffServ, firewall, virtual private network). When fully configured, the most demanding functions (*e.g.*, packet classification, traffic shaper) would be implemented as programmable hardware (*i.e.*, FPGA) modules and other functions would be implemented as software modules that are dynamically loaded into the OS kernel of the PC. The reconfigurable hardware is included in an add-on card that extends the core functionality provided by the Port Processors.

In addition, the PCs can be equipped with SPC-based network probes for monitoring port traffic. We are developing the network probes (both hardware and software) as part of an NSF project. One unique aspect of the probe is that its execution environment will combine our active networking OS kernel environment with an embedded TAO prototype. The NOC working through TAO will be able to remotely configure and manage the probe. An agent local to the probe

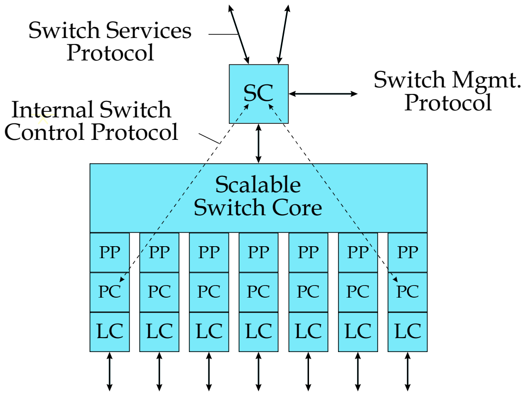

Figure 6: MSSN Router

will mediate between the NOC and the probe functions.

Setting up a connection involves one or more Switch Controllers (SCs) sending control messages to the PCs along the connection path. The PCs, in turn, configure the ports so that packets are steered through the appropriate hardware and software modules. The communication channel between hardware components is established using virtual circuit switching. When packets must be steered through software modules, the virtual circuit will lead to a PC where it is demultiplexed using zero-copy DMA to the appropriate software module by a function dispatcher in the PC's OS kernel (the OS kernel will be based on the NodeOS being developed by our Active Networking project). The packet is processed and then forwarded out to the next module.

4.3 Task 3 – AFS

Network management agents perform various tasks (such as MIB maintenance, threshold monitoring, event filtering, and fault correlation) on behalf of management applications or administrators. As shown in Figure 1, agents in our proposed NMVC++ are built atop network probes and are used to track event flows, as well as classify and report events of interest to NOCs. Common events reported to NOCs include *alarms* (such as delay or packet loss thresholds being exceeded for a service class), and *quality of service statistics* such as the performance of particular IPv6 packet flows, and ATM connection blocking rates. When these types of events are detected by the agent, it notifies the NOC via a trap and/or performs a corresponding local action.

In large-scale high speed networks, a tremendous volume of events can be generated by nodes. However, administrators and management applications may only be interested in a subset of these events. Therefore, *event filtering* is necessary to reduce the volume of events that agents deliver to NOCs. A unique feature of our proposal is its declarative approach to event filter specification and optimization. We propose the following techniques to enhance the flexibility, scalability, and performance of event filtering in NMVC system agents:

Filter declarations: Filter declarations will be specified in a high-level specification language, transmitted across the network to the node agents, and compiled into machine code that is optimized for each network node platform. We propose to build a high-level declarative specification language that defines filters as a set of predicates. Filter predicates can be used to specify protocol header fields (such as TCP port numbers, IP source and destination addresses, ATM VCs), application-level information (such as timestamps, telemetry measurand values, and image type), as well as node parameters (such as buffer occupancy, cumulative packet loss, and average delay). The use of a declarative language facilitates external reasoning about the static and dynamic properties of events, supports automated optimizations like compilation and filter composition, and allows remote configuration and control. For instance, filters can be programmed by NOCs to adjust their sampling rate for different classes of events.

Filter composition: Another optimization will compose multiple filters to reduce the number of operations required to process E messages through N event filters with P predicates from $O(E \times N \times P)$ to $O(M \times P)$, which is the minimum possible. This enables scalable performance improvements when N becomes large (as is the case for agents in a large-scale networks). Efficient filter composition will be implemented using a dynamic trie-based filter engine. To reduce search overhead, the trie coalesces common predicates according to criteria such as frequency of duplication or predicate priority. Each node in the trie implements a particular branching mechanism. The branching mechanism selected at a node employs a search method (such as hashing, binary search, or a series of relational expression comparisons) that is optimized to reduce filtering overhead. To eliminate the cost of interpretation, we will use dynamic compilation and linking techniques [69] to flexibly configure (and reconfigure) high-performance event filters.

4.4 Task 4 – POLKA++

Components of POLKA++ will include algorithms for efficient online ordering of events, a distributed visualization system, a library of interactive visualizations designed to support the tasks of network monitoring, visualization, and control, and a View Choreographer through which users of the visualization system may specify mappings from network events to higher-level events and from both low-level and high-level events to graphical updates in the visualizations.

The algorithms for efficient online ordering are required to ensure that the events stream arriving at the visualizations is consistent with the state of the network being visualized. As event streams from individual nodes in the network are merged at a NOC, variation in network congestion and processing delays can "misorder" events in the combined event stream. These out-of-order events can adversely affect analysis tools that process the event stream and can result in misleading displays. We will develop and employ advanced ordering filters that ensure that the event stream reaching NOC displays and analysis tools adheres to causal ordering.

A visualization system employing the POLKA paradigm, which supports the display of concurrent systems through the separation of the scheduling and execution of graphical updates, will be developed to support the rapid, informative, interactive display of various network operations, states, and statistics. In addition to the underlying visualization infrastructure, we will develop a library of visualizations, through which network administrators may monitor network performance, investigate patterns or anomalies, and explore and alter parameters and policies. These concurrent displays will provide color, 2 1/2 dimensional, smooth animations.

The View Choreographer supports user-specification of mappings from network events to higher-level events, from events of any level to graphical updates in selected visualizations, and from interactions with the displays to interactions with the network. In this way, new displays and interactions may be defined dynamically. The choreographer is also responsible for filtering the stream of events that reaches the display system, preventing redundancy in the graphical updates, and streamlining the display process.

4.5 Task 5 – Distributed Simulation Application Demonstration and Testbed Evaluation

In consultation with DoE, we will develop an application similar to the one shown in Figure 7. This application will

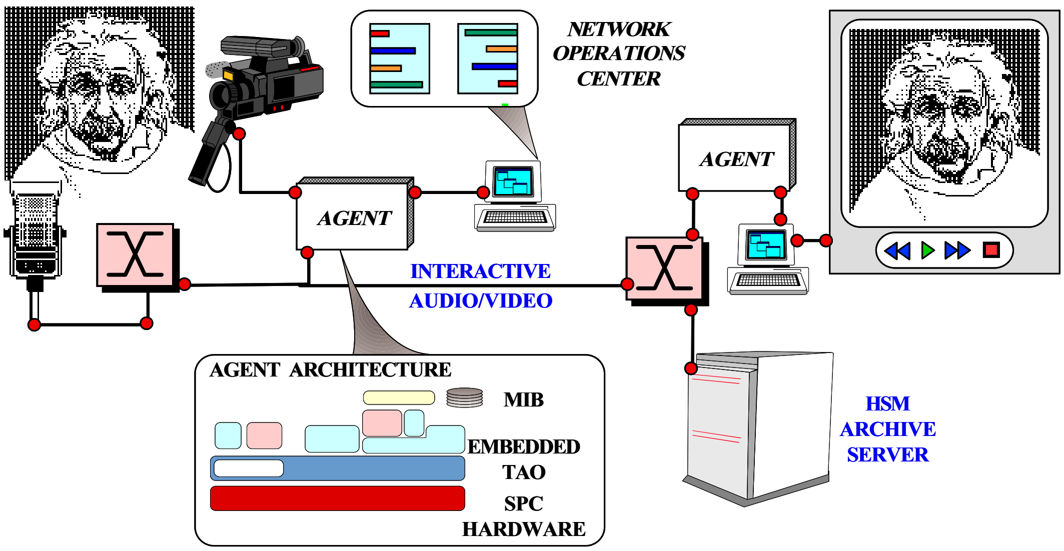

Figure 7: Collaborative Application Running Over the NMVC++ System

support groups of scientists collaborating over high-speed networks, downloading large multimedia data sets, interacting remotely using visual displays, spawning analysis computations dynamically, monitoring and controlling long-running computations, and updating/refining the data in a database, as well a serve as a vehicle for empirical testing and evaluation.

Our collaborative application will showcase the advanced QoS control capabilities of the integrated NMVC++ configuration infrastructure. These capabilities will allow the application and/or administrators to manipulate a variety of network control parameters in response to observed behavior and to specify policies to suit specific requirements. This will include vertically and horizontally control mechanisms at the network (*e.g.*, MSSN and IP in the switches and routers), and middleware (*i.e.*, embedded TAO), Agent, and application levels.

At the MSSN level, we will control parameters that govern such things as routing, packet level discarding, and overbooking ratios. At the embedded TAO middleware level, we will control interface services that can detect resource contention at run-time and react dynamically to improve the amount and type of data that can be delivered while maintaining end-to-end real-time application timing constraints. At the Agent level, we will optimize adaptive feedback loop to minimize latency to a point representative for these types of collaborative multimedia applications.

5 Statement of Work

The proposed project is a three year effort where we will create our NMVC++ testbed by (1) integrating and extending the current projects outlined above and (2) including the following new components and innovations we will develop as tasks for our proposed effort:

Task 1 – Embedded TAO (a small footprint, real-time ORB): This task will focus on enhancing the existing TAO framework to included support for QoS specification and enforcement for specific networking technologies and protocols, produce a minimal footprint ORB for embedding in network switches, and integrate with existing signaling protocols. Task 1 is comprised of the following subtasks:

Task 1.1: Make TAO comply to the Minimum CORBA specification [26].

Task 1.2: Embed TAO in switches (*e.g.*, switch controllers) and on the client hosts and the signaling processors.

Task 1.3: Implement selected signaling protocol(s) using embedded TAO.

Task 2 – DiSp (a differentiated services framework): This task involves the design, prototype implementation, and testing of several differentiated service architectures (including our own **DiSp** architecture and the IETF architecture) using our MultiService Switching Network (MSSN). Task 2 is comprised of the following subtasks:

Task 2.1: Design and prototype dynamically loadable kernel DiSp modules for edge routers (traffic shaping, packet forwarding, congestion control and link monitoring).

Task 2.2: Design and prototype NOC DiSp modules (admission control, fault and performance management).

Task 2.3: Test the the DiSp modules in our MSSN testbed.

Task 2.4: Prototype the IETF DiffServ framework including internal router modules.

Task 2.5: Test the the IETF DiffServ modules in our MSSN testbed.

Task 2.6: Develop a general framework for constructing families of differentiated services.

Task 3 – AFS (a network management agent filtering system): This task will develop a network management agent system that uses embedded TAO to provide highly scalable multi-attribute event filtering. Task 3 is comprised of the following subtasks:

Task 3.1: Define a framework for remotely configuring and installing Agents with Probes on network nodes.

Task 3.2: Integrate with ANN features to activate/deactivate protocol probes in response to requests from the NOC.

Task 3.3: Define a protocol for installing mobile probes to isolate root causes of complex and anomalous network behavior that can not be identified with statically installed probes.

Task 3.4: Define a dynamic trie-based filtering mechanism that reduces the work required to classify E events of size L through N event filters from $O(E \times L \times N)$ to $O(E \times L)$, which is the minimum possible.

Task 4 – POLKA++ (a distributed visualization system): This task involves the design, implementation, and testing of a distributed visualization system for network management. Task 4 is comprised of the following subtasks:

Task 4.1: Develop an interactive, graphical interface that supports the rapid, informative display of network operation statistics and causally ordered events [70].

Task 4.2: Develop advanced ordering filters that ensure that the event streams reaching View Choreographers on an NOC adhere to causal ordering.

Task 5 – Demonstration of a distributed simulation application and results of empirical studies in our NMVC++ testbed: The performance and effectiveness of the differentiated services architectures will be evaluated on the Multi-Service Switching Network (MSSN) testbed. Task 5 is comprised of the following subtasks:

5.1: Assemble and test a prototype MSSN testbed built around a WUGS-20 switching fabric and 16 OC-12 interface cards.

5.2: Experimentally evaluate the MSSN with synthesized and real traffic.

5.3: Experimentally evaluate the DiSp and IETF DiffServ architectures with synthesized and real traffic.

References

[1] SLAC, "BaBar Collaboration Home Page." http://www.slac.stanford.edu/BFROOT/.

[2] A. Kruse, "CMS Online Event Filtering," in *Computing in High-energy Physics (CHEP 97)*, (Berlin, Germany), Apr. 1997.

[3] T. H. Harrison, D. L. Levine, and D. C. Schmidt, "The Design and Performance of a Real-time CORBA Event Service," in *Proceedings of OOPSLA '97*, (Atlanta, GA), ACM, October 1997.

[4] D. C. Schmidt, "A Family of Design Patterns for Application-level Gateways," *The Theory and Practice of Object Systems (Special Issue on Patterns and Pattern Languages)*, vol. 2, no. 1, 1996.

[5] P. Jain, S. Widoff, and D. C. Schmidt, "The Design and Performance of MedJava – Experience Developing Performance-Sensitive Distributed Applications with Java," *IEE/BCS Distributed Systems Engineering Journal*, 1998.

[6] I. Pyarali, T. H. Harrison, and D. C. Schmidt, "Design and Performance of an Object-Oriented Framework for High-Performance Electronic Medical Imaging," *USENIX Computing Systems*, vol. 9, November/December 1996.

[7] Center for Distributed Object Computing, "Successful Project Deployment of ACE and TAO." www.cs.wustl.edu/∼schmidt/users.html, Washington University.

[8] J. Turner, "An optimal nonblocking multicast virtual circuit switch," in *Proceedings of the Conference on Computer Communications (INFOCOM)*, pp. 298–305, June 1994.

[9] M. Waldvogel, G. Varghese, J. S. Turner, and B. Plattner, "Scalable High Speed IP Routing Lookups," in *Proceedings of SIGCOMM '97*, (Cannes, France), ACM, August 1997.

[10] J. Turner and N. Yamanaka, "Architectural Choices in Large Scale ATM Switches," *ICICE Transactions*, 1998.

[11] D. C. Schmidt, D. L. Levine, and S. Mungee, "The Design and Performance of Real-Time Object Request Brokers," *Computer Communications*, vol. 21, pp. 294–324, Apr. 1998.

[12] I. Pyarali, C. O'Ryan, D. C. Schmidt, N. Wang, V. Kachroo, and A. Gokhale, "Applying Optimization Patterns to the Design of Real-time ORBs," in *Proceedings of the* 5^{th} *Conference on Object-Oriented Technologies and Systems*, (San Diego, CA), USENIX, May 1999.

[13] F. Kuhns, D. C. Schmidt, and D. L. Levine, "The Design and Performance of RIO – A Real-time I/O Subsystem for ORB Endsystems," in *Proceedings of the* 5^{th} *IEEE Real-Time Technology and Applications Symposium*, (Vancouver, British Columbia, Canada), IEEE, June 1999.

[14] G. Parulkar, D. C. Schmidt, E. Kraemer, J. Turner, and A. Kantawala, "An Architecture for Monitoring, Visualization, and Control and Gigabit Networks," *IEEE Network*, vol. 11, September/October 1997.

[15] T. Chaney, A. Fingerhut, M. Flucke, and J. S. Turner, "Design of a Gigabit ATM Switch," in *IEEE INFOCOM '97*, (Kobe, Japan), IEEE Computer Society Press, April 1997.

[16] G. Parulkar, D. C. Schmidt, and J. S. Turner, "$\mathrm{a^It^Pm}$: a Strategy for Integrating IP with ATM," in *Proceedings of the Symposium on Communications Architectures and Protocols (SIGCOMM)*, ACM, September 1995.

[17] Z. D. Dittia, J. Jerome R. Cox, and G. M. Parulkar, "Using an ATM Interconnect as a High Performance I/O Backplane," in *Proceedings of the Hot Interconnects Symposium*, August 1994.

[18] D. C. Schmidt, S. Mungee, S. Flores-Gaitan, and A. Gokhale, "Software Architectures for Reducing Priority Inversion and Non-determinism in Real-time Object Request Brokers," *Journal of Real-time Systems*, To appear 1999.

[19] A. Gokhale and D. C. Schmidt, "Optimizing a CORBA IIOP Protocol Engine for Minimal Footprint Multimedia Systems," *Journal on Selected Areas in Communications special issue on Service Enabling Platforms for Networked Multimedia Systems*, to appear, 1999.

[20] D. Decasper, Z. Dittia, G. Parulkar, and B. Plattner, "Router plugins - a modular and extensible software framework for modern high performance integrated services routers," Department of Computer Science TR-98-08, Washington University in St. Louis, February 1998.

[21] A. Mehra, A. Indiresan, and K. G. Shin, "Structuring Communication Software for Quality-of-Service Guarantees," *IEEE Transactions on Software Engineering*, vol. 23, pp. 616–634, Oct. 1997.

[22] T. Abdelzaher, S. Dawson, W.-C.Feng, F.Jahanian, S. Johnson, A. Mehra, T. Mitton, A. Shaikh, K. Shin, Z. Wang, and H. Zou, "ARMADA Middleware Suite," in *Proceedings of the Workshop on Middleware for Real-Time Systems and Services*, (San Francisco, CA), IEEE, December 1997.

[23] W. Feng, U. Syyid, and J.-S. Liu, "Providing for an Open, Real-Time CORBA," in *Proceedings of the Workshop on Middleware for Real-Time Systems and Services*, (San Francisco, CA), IEEE, December 1997.

[24] V. Kalogeraki, P. Melliar-Smith, and L. Moser, "Soft Real-Time Resource Management in CORBA Distributed Systems," in *Proceedings of the Workshop on Middleware for Real-Time Systems and Services*, (San Francisco, CA), IEEE, December 1997.

[25] J. A. Zinky, D. E. Bakken, and R. Schantz, "Architectural Support for Quality of Service for CORBA Objects," *Theory and Practice of Object Systems*, vol. 3, no. 1, 1997.

[26] Object Management Group, *Minimum CORBA - Joint Revised Submission*, OMG Document orbos/98-08-04 ed., August 1998.

[27] Object Management Group, *Realtime CORBA 1.0 Joint Submission*, OMG Document orbos/98-12-05 ed., December 1998.

[28] A. Kantawala, S. Norden, K. Wong, and G. Parulkar, "DiSp: An Architecture for supporting Differentiated Services in the Internet," in *Submitted to INET '99*, Feb. 1999.

[29] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," *Network Information Center RFC 2475*, December 1998.

[30] D. C. Schmidt, "High-Performance Event Filtering for Dynamic Multi-point Applications," in 1^{st} *Workshop on High Performance Protocol Architectures (HIPPARCH)*, (Sophia Antipolis, France), INRIA, December 1994.

[31] M. Jayaram and R. Cytron, "Efficient Demultiplexing of Network Packets by Automatic Parsing," in *Proceedings of the Workshop on Compiler Support for System Software (WCSSS 96)*, (University of Arizona, Tucson, AZ), February 1996.

[32] MCI and NSF, "Very high performance Backbone Network Service (vBNS)." http://www.vbns.net/.

[33] CAIRN, "Collaborative Advanced Internet Research Network (CAIRN)." http://www.cairn.net/.

[34] DARPA, NASA, and NSF, "The National Transparent Optical Network (NTON)." http://www.ntonc.net/.

[35] L. Crutcher and A. A. Lazar, "Management and Control for Giant Gigabit Networks - are we ready for B-ISDN," Tech. Rep. 341-93-21, Center for Telecommunications Research, Columbia University, 1993.

[36] L. A. Crutcher, A. A. Lazar, S. K. Feiner, and M. Zhou, "Management of Broadband Networks Using a 3D Virtual World," in *Proceedings of the 2nd International Symposium on High-Performance Distributed Computing*, July 1993.

[37] K. S. Klemba, "Openview's Architectural Models," in *Proceedings of the* 1^{st} *International Symposium on Integrated Network Management* (B. Meandzija and J. Westcott, eds.), pp. 565–572, IFIP, 1989.

[38] I. NetView/6000.

[39] D. Wilkelstein, "ATTILA: Performance Analysis System for ATM Networks," tech. rep., MCNC, NC, 1994.

[40] A. A. Lazar, "A Real Time Control, Management and Information Transport Architecture for Broadband Networks," in *Proceedings of the International Zurich Seminar on Digital Communications*, Mar. 1992.

[41] B. Tierney, W. Johnston, J. Lee, and G. Hoo, "Performance Analysis in High-Speed Wide Area IP over ATM Networks: Top-to-Bottom End-to-End Monitoring," *IEEE Networking*, 1996.

[42] B. Braden, D. Clark, and S. Shenker, "Integrated services in the internet architecture," *Network Information Center RFC 1633*, June 1994.

[43] Isis Distributed Systems, Inc., Marlboro, MA, *Isis Users's Guide: Reliable Distributed Objects for C++*, April 1994.

[44] K. Birman and R. van Renesse, *Reliable Distributed Computing with the Isis Toolkit*. Los Alamitos: IEEE Computer Society Press, 1994.

[45] R. T. Braden, "A Pseudo-machine for Packet Monitoring and Statistics," in *Proceedings of the Symposium on Communications Architectures and Protocols (SIGCOMM)*, (Stanford, CA), ACM, August 1988.

[46] Sun Microsystems, Inc., Mountain View, CA, *NIT(4P); SunOS 4.1.1 Reference Manual*, Part number: 800-5480-10 ed., October 1990.

[47] I. O. for Standardization, "Information Processing Systems - Open Systems Interconnection - Part 5: Event Report Management Function," Tech. Rep. ISO/IEC DIS 10164-5, ISO.

[48] J. C. Mogul, "Efficient Use of Workstations for Passive Monitoring of Local Area Networks," in *Proceedings of the Symposium on Communications Architectures and Protocols (SIGCOMM)*, (Philadelphia, PA), ACM, Sept. 1990.

[49] J. C. Mogul, R. F. Rashid, and M. J. Accetta, "The Packet Filter: an Efficient Mechanism for User-level Network Code," in *Proceedings of the* 11^{th} *Symposium on Operating System Principles (SOSP)*, November 1987.

[50] S. McCanne and V. Jacobson, "The BSD Packet Filter: A New Architecture for User-level Packet Capture," in *Proceedings of the Winter USENIX Conference*, (San Diego, CA), pp. 259–270, Jan. 1993.

[51] M. Yuhara, B. Bershad, C. Maeda, and E. Moss, "Efficient Packet Demultiplexing for Multiple Endpoints and Large Messages," in *Proceedings of the Winter Usenix Conference*, January 1994.

[52] M. L. Bailey, B. Gopal, P. Sarkar, M. A. Pagels, and L. L. Peterson, "Pathfinder: A pattern-based packet classifier," in *Proceedings of the* 1^{st} *Symposium on Operating System Design and Implementation*, USENIX Association, November 1994.

[53] N. Gehani, H. V. Jagadish, and O. Shmueli, "Compose: A System for Composite Event Specification and Detection," in *Book chapter in Advanced Database Concepts and Research Issues* (N. R. Adam and B. Bhargava, eds.), Lecture Notes in Computer Science, Springer Verlag, 1994.

[54] S. Gatziu and K. R. Dittrich, "Detecting Composite Events in Active Database Systems Using Petri Nets," in *Proceedings of the* 4^{th} *International Workshop on Research Issues in Data Engineering: Active Database Systems*, IEEE, February 1994.

[55] O. Wolfson, S. Sengupta, and Y. Yemini, "Managing Communication Networks by Monitoring Databases," *IEEE Transactions on Software Engineering*, vol. 17, pp. 944–952, Sept. 1991.

[56] C. Maeda and B. Bershad, "Protocol Service Decomposition for High-Performance Networking," in *Proceedings of the* 14^{th} *Symposium on Operating System Principles*, (Asheville, North Carolina), ACM, December 1993.

[57] M. T. Heath and J. A. Etheridge, "Visualizing the performance of parallel programs," *IEEE Software*, vol. 8, pp. 29 –39, Sept. 1991.

[58] G. Geist, M. Heath, B. Peyton, and P. Worley, "A user's guide to PICL: A portable instrumented communication library," Tech. Rep. TM-11616, Oak Ridge National Laboratory, Sept. 1990.

[59] D. A. Reed, R. A. Aydt, R. J. Noe, P. C. Roth, K. A. Shields, B. Schwarta, and L. F. Tavera, "An overview of the pablo performance analysis environment," in *Proceedings of the Scalable Parallel Libraries Conference*, (Mississippi State, MS, USA), pp. 104–113, Oct. 1994.

[60] J. T. Stasko, "The parade environment for visualizing parallel program executions: A progress report," Tech. Rep. GIT-GVU-95-03, Graphics, Visualization, and Usability Center, Georgia Institute of Technology, Atlanta, GA, Jan. 1995.

[61] J. T. Stasko and E. Kraemer, "A Methodology for Building Application-specific Visualizations of Parallel Programs," *Journal of Parallel and Distributed Computing*, vol. 18, pp. 258–264, June 1993.

[62] E. Kraemer and J. T. Stasko, "Toward flexible control of the temporal mapping from concurrent program events to animations," *Proceedings Eighth International Parallel Processing Symposium*, pp. 902–908, 1994.

[63] W. Gu, G. Eisenhauer, K. S. E. Kraemer, J. Stasko, J. Vetter, and N. Mallavarupu, "Falcon: on-line monitoring and steering of large-scale parallel programs," in *Proceedings of the Fifth Symposium on the Frontiers of Massively Parallel Computation*, pp. 422–429, Feb. 1995.

[64] P. Newman, W. Edwards, R. Hinden, E. Hoffman, F. Ching Liaw, T. Lyon, and G. Minshall, "Ipsilon's General Switch Management Protocol Specification Version 2.0," Standards Track RFC 2297, Network Working Group, March 1998.

[65] P. Newman, W. Edwards, R. Hinden, E. Hoffman, F. Ching Liaw, T. Lyon, and G. Minshall, "Ipsilon's General Switch Management Protocol Specification Version 1.1," Standards Track RFC 1987, Network Working Group, August 1996.

[66] S. J. Golestani, "Congestion-free transmission of real-time traffic in packet networks," in *Proc. of SIGCOMM '90*, August 1990.

[67] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group." Internet Draft, Feb. 1999.

[68] V. Jacobson, K. Nichols, and K. Poduri, "An Expedited Forwarding PHB." Internet Draft, Feb. 1999.

[69] D. C. Schmidt and T. Suda, "An Object-Oriented Framework for Dynamically Configuring Extensible Distributed Communication Systems," *IEE/BCS Distributed Systems Engineering Journal (Special Issue on Configurable Distributed Systems)*, vol. 2, pp. 280–293, December 1994.

[70] E. Kraemer, "The Visualization of Parallel Systems: An Overview," *Journal of Parallel and Distributed Computing*, vol. 18, June 1993.

[71] R. Lachenmaier, "Open Systems Architecture Puts Six Bombs on Target." www.cs.wustl.edu/$\sim$schmidt/TAO-boeing.html, Dec. 1998.

[72] J. Hu, I. Pyarali, and D. C. Schmidt, "Measuring the Impact of Event Dispatching and Concurrency Models on Web Server Performance Over High-speed Networks," in *Proceedings of the* 2^{nd} *Global Internet Conference*, IEEE, November 1997.

[73] J. Hu, S. Mungee, and D. C. Schmidt, "Principles for Developing and Measuring High-performance Web Servers over ATM," in *Proceeedings of INFOCOM '98*, March/April 1998.

[74] J. O. Coplien and D. C. Schmidt, eds., *Pattern Languages of Program Design*. Reading, MA: Addison-Wesley, 1995.

[75] M. Fayad, R. Johnson, and D. C. Schmidt, eds., *Object-Oriented Application Frameworks: Problems & Perspectives*. New York, NY: Wiley & Sons, 1999.

[76] M. Fayad, R. Johnson, and D. C. Schmidt, eds., *Object-Oriented Application Frameworks: Applications & Experiences*. New York, NY: Wiley & Sons, 1999.

[77] K. Wong and S. Garg, "Improving the speed of a distributed checkpointing algorithm," in *Proc. Int'l Conf. on Parallel and Distributed Computing Systems*, October 1993.

[78] K. Wong and M. Franklin, "Distributed computing systems and checkpointing," in *Proc. Conf. on High-Performance Distributed Computing*, (Spokane, Washington), July 1993.

[79] K. Wong, M. Franklin, and M. Wolfe, "Distributed system checkpointing: Model and verification," Tech. Rep. WUCS-94-18, Department of Computer Science, Washington University, May 1994.

[80] M. Franklin, D. Wann, and K. Wong, "Parallel machines and algorithms for discrete event simulation," in *Proc. 1984 Int'l Conf. on Parallel Processing*, pp. 449–458, August 1984.

[81] K. Wong, M. Franklin, R. Chamberlain, and B. Shing, "Statistics on logic simulation," in *Proc. 23rd Design Automation Conf.*, pp. 13–19, June 1986.

[82] K. Wong and M. Franklin, "Performance analysis and design of a logic simulation machine," in *Proc. 14th Int'l Symp. on Comp. Arch.*, pp. 46–55, June 1987.

[83] K. Wong and M. Franklin, "Performance analysis and design of a logic simulation machine," in *Proc. 14th Int'l Symp. on Comp. Arch.*, pp. 46–55, June 1987.

[84] K. Wong and M. Franklin, "Load and communications balancing on multiprocessor logic simulation engines," in *Proc. Int'l Workshop on Hardware Accelerators*, October 1987.

[85] K. Wong and M. Franklin, "Performance analysis and design of a logic simulation machine," *Jrnl. of Parallel and Distributed Comp.*, vol. 7, pp. 416–440, October 1989.

A Facilities

Washington University has excellent facilities and infrastructure to support the proposed research program. Computing facilities in the department include over 200 servers and workstations for use by faculty, research staff and students in various labs and centers. All these reside on the University's campus-wide network, which provides communication with the campus and the Internet. Washington University also has one of the largest operational campus ATM networks.

The proposed research project will be carried out in Washington University's Applied Research Laboratory (ARL) and Center for Distributed Object Computing (DOC). The purpose of ARL and DOC is to develop high performance hardware and real-time software/hardware technologies by building practical prototype systems and deploying them in testbed and production settings [71].

Both ARL and the DOC Center have several projects in progress, with Projects Zeus and TAO being the oldest. The goal of Project Zeus has been to develop an ATM campus network that can support multirate data, CD quality audio, video, and high resolution images in an integrated fashion. The phase-0 of this project led to prototyping of the ATM switch and then to the nation's first multi-switch metropolitan area ATM network. The goal of the TAO project is to develop flexible, efficient, and predictable OO communication middleware based on CORBA that can automate common communication software tasks, such as connection establishment, event demultiplexing, event handler dispatching, message routing, dynamic configuration of services, priority-based real-time thread scheduling, and flexible management of parallel protocol and service processing.

Our existing network includes eleven 16-port, 155 Mb/s per port (commercial versions of our own) ATM switches and over 50 endpoints that include multimedia workstations, compute and storage servers, and other interesting imaging devices. Connected by several hundred miles of fiber optic cable, these switches are located throughout the Hilltop and Medical campuses and also provide a connection to Barnes West County Hospital. The network supports routine network applications and a variety of multimedia and imaging applications including multi participant teleconferencing and collaboration, electronic radiology, and video-on-demand.

The multimedia capability at a workstation is provided by a platform-independent "pizza box" called the MultiMedia eXplorer (MMX). The MMX incorporates an ATM network adapter, an ATM extension port, an RS232 link for control by the workstation, a full-duplex motion JPEG codec for production quality video, a DSP-based codec for CD quality audio, and a high-speed serial port for the delivery of high-resolution images.

ARL's Washington University Gigabit Switching (WUGS) ATM Testbed Project (sponsored by ARPA/ITO) is concerned with the design of two key gigabit network technologies: highly scalable multicast ATM switching systems and host-network interfaces suitable for supporting gigabit data rates to and from application-level programs. Delivering gigabits to applications has meant rethinking the host I/O subsystem and protocols and their implementations within the host operating system. The project will culminate in the creation of a gigabit testbed with multiple switching systems supporting link speeds of 600 Mb/s, 1.2 Gb/s and 2.4 Gb/s and supporting workstation and server applications at 1.2 Gb/s.

The DOC Center's TAO project (sponsored by ARPA/ITO, NSF, and many major telecommunication and aerospace companies) provides an open-source, standards-based, high-performance, real-time ORB endsystem that supports applications with deterministic and statistical QoS requirements, as well as "best-effort" requirements, and is the first ORB to support end-to-end QoS guarantees over ATM/IP networks [16]. TAO's features and optimizations include an ORB Core that minimizes context switching, synchronization, dynamic memory allocation, and data movement [18]; a highly-scalable Object Adapter that demultiplexes requests in constant-time [12]; an optimizing IDL compiler [19]; real-time I/O subsystem [13], and a global resource allocation and scheduling framework [11].

ARL and and DOC currently have a combined full-time hardware and software engineering staff of 15 individuals who provide the continuity and consistency required to ensure that demanding systems projects, such as this one, can be carried through to a predictable completion.

B Technology Transfer Plan

An important goal of the Applied Research Laboratory (ARL) and the Center for Distributed Object Computing (DOC) is to develop high performance hardware and software technologies by building practical prototype systems and deploying them in production and testbed settings. Research ideas, particularly in the area of computing and communication, increasingly require the construction of prototypes and a capability to produce timely demonstrations. Industry collaborators find it essential to understand performance and engineering issues before making large commitments to new products. Moreover, the rapid development of prototypes in universities can often substantially reduce the time-to-market for new products.

ARL and the DOC Center take pride in prototyping systems that can be licensed to industry for product development and external R&D. ARL has licensed the following technologies from its laboratory:

- A scalable ATM switch architecture and associated ASIC (Application Specific Integrated Circuit) designs to SynOptics, a Santa Clara, California-based company with offices worldwide (SynOptics later became Bay Networks and was recently acquired by Nortel).
- ATM signaling software to SynOptics, Southwestern Bell Co., and Ascom Nexion, a Massachusetts based company with a software division in St. Louis (Ascom Nexion was recently acquired by Fujitsu).
- MMX, the multimedia explorer, a hardware and software system which allows any workstation to do high performance multimedia over an ATM network, to STS Technologies, a St. Louis-based spinoff company.
- Gigabit ATM switches and associated network interface cards. This technology has been distributed to 30 universities in the form of *Gigabit Network Kits*, providing a flexible, open research platform that can be modified and extended to pursue a variety of research objectives. This technology has been licensed to STS Technologies.
- Advanced network technology for the construction of high performance routing switches capable of routing IP packets at link rates of 10 Gb/s and with aggregate system capacities of move than 10 Tb/s. This technology has been licensed to Growth Networks, Inc., a California-based startup company.

Likewise, the DOC Center has developed the following middleware technologies, which are now supported commercially and used in many industry R&D centers and product groups:

- The ADAPTIVE Communication Environment (ACE) – ACE is a widely used OO framework containing components that implement key patterns for high-performance and real-time communication systems [69]. ACE has been used for many communication software systems in research labs and commercial projects, including Bellcore, Boeing, CERN, Cisco, DEC, Ericsson, Kodak, JPL, Lucent, Lockheed Martin, Motorola, NASA, Raytheon, SAIC, Siemens, SLAC, and StorTek. A commercial company, Riverace, supports ACE using an open-source software model.
- The ACE ORB (TAO) – TAO is a high-performance, real-time ORB endsystem targeted for applications with deterministic and statistical QoS requirements, as well as best effort requirements [11]. The TAO ORB endsystem is developed using components in ACE and has been used in many commercial products, including a family of avionics mission computing systems at Boeing [3], satellite communications at Raytheon, and HLA RTI simulation middleware at SAIC. A commercial company, OCI, supports TAO using an open-source software model.
- The JAWS Adaptive Web Server (JAWS) – JAWS is a high-performance, adaptive Web server framework [72, 73]. It is also developed using components in ACE and has been integrated into commercial systems at a number of companies, including Siemens. A commercial company, Entera, supports JAWS using an open-source software model.

The number of network providers has been rapidly increasing in recent years, and there is significant evidence that the trend will continue. These providers will need network technologies capable of supporting a range of communication services and will need improved management tools to better manage their networks and better serve their customer community.

We believe that network providers will find high quality comprehensive network management an increasingly important requirement, as growing reliance on communication networks makes it more and more important to ensure reliable, trouble-free operation.

Successful completion of this project will lead to the following technologies or systems that would be appropriate for licensing to industry, and thus for technology transfer:

The APIC-based network probe: The network probe can be packaged as a stand-alone box, such as today's LAN analyzers, and can be used with other network management systems for data collection and event filtering for Gb/s links.

The AFS event filtering language and optimization techniques used in the network management agent: The filtering language and optimization techniques will be compatible with JAVA and the remote installation mechanisms will be compatible with HTTP and Applet downloading technology available in the WWW. Thus, these components can be directly applied by other researchers and developers to a broad range of applications that require high-performance filtering in the Internet.

Languages and code generators: The language and method for specifying ordering relationships and for generating code to perform the online reordering of events. Similar technology is needed in any system that attempts to perform event-based visualization of distributed systems. Thus, such a component would be needed for interactive program steering, and variations of this technology could be useful in ensuring fairness in distributed games and interactive simulations.

The View Choreographer: This toolkit performs mappings from program events to graphical updates, in combination with the suite of display views. These same displays could be used with other events and event formats. Similarly, new displays can be created. The use of a View Choreographer separates the events from the displays, forming a system of "composable parts", facilitating portability, maintainability, and re-use.

C Biographical Sketches and Qualifications

The Co-PIs for the project are Douglas Schmidt, Jonathan Turner, Ken Wong, Fred Kuhns, and David Levine from Washington University and Eileen Kraemer from the University of Georgia. As indicated below, the team of researchers is uniquely qualified to undertake the proposed effort. In particular, Dr. Schmidt is a leading expert in the area of distributed object computing systems and real-time middleware. Dr. Turner is a leading expert in the field in the design, implementation, and deployment of high speed ATM networks. Dr. Kraemer is an expert in the area of distributed and parallel visualization. Furthermore, our team has extensive experience building large-scale distributed applications, middleware, and high-speed networks and transferring their state-of-the-art technology to government R&D groups, universities, and commercial ventures.

Dr. Douglas C. Schmidt is an Associate Professor and Director of the Center for Distributed Object Computing in the Department of Computer Science and in the Department of Radiology at Washington University in St. Louis, Missouri, USA. His research focuses on design patterns, implementation, and experimental analysis of object-oriented techniques that facilitate the development of high-performance, real-time distributed object computing systems on parallel processing platforms running over high-speed ATM networks and embedded system interconnects.

Dr. Schmidt is an internationally recognized expert on distributed object computing and real-time CORBA middleware, and has published over 75 papers in IEEE, ACM, IFIP, and USENIX journals, technical conferences, and workshops. He is the chief architect and designer for two significant software systems: (1) The ADAPTIVE Communication Environment (ACE), which is a widely used OO framework containing components that implement key patterns for high-performance and real-time communication systems [69] and (2) the ACE ORB (TAO) which is a high-performance, real-time ORB [11, 3].

Dr. Schmidt is the editor of several books on patterns [74] and frameworks [75, 76] and is currently writing two book on these topics for Addison Wesley and Wiley & Sons. In addition to his academic research, Dr. Schmidt has served as a consultant for dozens of companies, including Ericsson, Motorola Iridium, Siemens, Kodak, Boeing, Lucent, Nortel, and Lockheed, where he has helped develop real-time avionics computing systems, distributed telecommunication switch management systems, network management software for global personal communications systems, and high performance distributed medical imaging systems over ATM networks. He is a member of IEEE, ACM, and USENIX.

Dr. Eileen T. Kraemer is an Assistant Professor of Computer Science at the University of Georgia. Previously, she was an Assistant Professor at Washington University in St. Louis. She received the Ph.D. in Computer Science from the Georgia Institute of Technology (1995), the M.S. in Computer Science from Polytechnic University, Brooklyn (1986), and the B.A. in Biology from Hofstra University (1980). Her research interests include the monitoring, analysis, and interactive steering of parallel and distributed networks and systems, and the development of tools for visualization and interaction in support of computational biology.

Her thesis research focused on the treatment of time and event order in the visualization of parallel and distributed systems, and involved the detection of synchronization events, the calculation of consistent event orders, and graphical support for the representation of concurrency. At Georgia Tech, she participated in the PARADE project for the visualization of parallel and distributed systems, and developed the Animation Choreographer component of that system. Also at Georgia Tech, she participated in the Falcon project for interactive program steering, and developed a filtering technique for the on-line reordering of events.

Dr. Jonathan S. Turner is Sever Professor of Engineering, and Director of the Applied Research Laboratory at Washington University. He is also former chairman of the Department of Computer Science and a co-founder of Growth Networks Inc., a venture-funded startup company that designs, develops and markets scalable network and communication products for the WAN market. He received the MS and PhD degrees in computer science from Northwestern University in 1979 and 1981. He is an internationally recognized expert in the design and analysis of switching systems, especially with respect to multicast ATM networks. He has been awarded more than a dozen patents for his work on switching systems and has many widely cited publications in this area.

Dr. Turner has been engaged in research in high speed networks for more than fifteen years. At Bell Laboratories, in the early eighties, he was the lead system architect for a project that sought to integrate voice and data communication using high performance hardware-based packet switching. This *fast packet switching* technology stimulated the development of frame relay and ATM. At Washington University he led a project leading to one of the first scalable multicast switch architectures for ATM. This technology was later licensed to SynOptics, which used it in their first commercial ATM switch products. More recently, he has led the development of switching system technology that can support link speeds of 2.4 Gb/s and aggregate system capacities of over one terabit per second. This technology has now been distributed in the form of *Gigabit Network Kits* to 30 different universities, providing a flexible, open research platform that can be freely modified and extended to pursue a wide variety of research agendas.

Dr. Turner's research interests also include the study of algorithms and computational complexity, with particular interest in the probable performance of heuristic algorithms for NP-complete problems. He is a member of the ACM, SIAM and a fellow of the IEEE.

Dr. Ken Wong is a Senior Research Associate in the Applied Research Laboratory and the Computer Science Department of Washington University. He received his B.S. degree in Electrical Engineering from M.I.T. (1968), and his M.S. and D.Sc. degrees in Computer Science degrees from Washington University, St. Louis (1974 and 1985). He is currently involved in two NSF funded projects. One is in the area of network monitoring, visualization and control, and the other involves a geographically distributed neuroscience data cache as part of the National Partnership for Advanced Computational Infrastructure (NPACI). Dr. Wong's previous research has been in the areas of high-performance simulation of large digital circuits, heterogeneous computing, and checkpointing and reconfiguration in a faulty, distributed computing environment [77, 78, 79]. Some of this work spanned the full range of algorithm design and modeling, implementation, measurement, and analysis of digital simulation using multicomputers, multiprocessors and special-purpose hardware [80, 81, 82, 83, 84, 85].

Fred Kuhns is a Senior Research Associate in the Center for Distributed Object Computing of the Department of Computer Science at Washington University, St. Louis. He received the M.S.E.E. from Washington University, St. Louis, and the B.S.E.E. from the University of Memphis, Memphis TN. His research interests focus on operating system and network support for high-performance, real-time distributed object computing systems. His recent research projects have focused on the design and implementation of real-time I/O subsystems, software support for high-performance interfaces, and QoS support in integrated service routers.

Dr. Levine is a Senior Research Associate in the Center for Distributed Object Computing of the Department of Computer Science at Washington University, St. Louis. He received the Ph.D. in Computer Science from the University of California, Irvine, the M.S.E.E./C.S. from George Washington University, and the B.S.M.E. from Cornell University. His current research interests include testing and performance analysis of real-time systems, and scheduling of distributed real-time systems. In addition, Dr. Levine has contributed substantial amounts to the ADAPTIVE Communication Environment (ACE) framework and The ACE ORB (TAO). Dr. Levine has extensive industry experience developing software for broadband telecommunications, high-fidelity electro-optic sensor system simulation, and both electric/hybrid and internal combustion engine vehicle applications. He is a Registered Professional Engineer in the District of Columbia.