NPSNET: A NETWORK SOFTWARE ARCHITECTURE FOR LARGE SCALE VIRTUAL ENVIRONMENTS

Michael R. Macedonia
Michael J. Zyda *
David R. Pratt
Paul T. Barham
Steven Zeswitz
*contact author

Naval Postgraduate School
Department of Computer Science
Monterey, CA 93943-5100
{macedonia, zyda, pratt, barham}@cs.nps.navy.mil

Abstract

This paper explores the issues involved in designing and developing network software architectures for large scale virtual environments. We present our ideas in the context of NPSNET-IV, the first 3D virtual environment that incorporates both the IEEE 1278 Distributed Interactive Simulation (DIS) application protocol and the IP Multicast network protocol for multi-player simulation over the Internet.

Key Words: Virtual reality, networked virtual environments, large scale distributed interactive simulation, DIS, IP Multicast.

I. Introduction

The construction of large scale virtual environments has been a long-stated goal of virtual environment (VE) proponents and now is a major objective of both commercial and government organizations. However, there exists major technical challenges which will require new network software architectures for distributed VEs. This paper examines these issues and presents the efforts by the NPSNET Research Group to develop such an architecture. We have developed the first 3D virtual environment suitable for multi-player participation over the Internet that uses IP multicast network protocols and the IEEE 1278 Distributed Interactive Simulation (DIS) application protocol [Deer89][IEEE93]. Though many factors have influenced our design this paper concentrates on the networking aspects of our distributed environment, NPSNET.

A. Motivation

The motivation for our effort is to expand the capabilities of simulations and virtual environments to serve medium to large numbers (more than 1,000) of simultaneous users. In fact, this has been a core but as yet unrealized idea of virtual reality. As Jaron Lanier recently wrote:

Virtual reality (VR) was intended to emphasize the social nature of shared (networked) virtual environments, and to emphasize that one's own body was in the world as well as in the external environment [Lanier94].

Moreover, interest by the government, military, and telecommunications industry in distributed virtual environments has been rapidly growing. For example, the US Army has plans for the Louisiana Maneuvers (LAM) initiative later in this decade which envisions 10,000 to 100,000 autonomous and human players participating in a simulation over a global wide-area network [USA93]. In a military context, a virtual environment of this magnitude enables a sense of collocation - sharing the same virtual battle space - among many geographically dispersed players such as an aviation unit in Texas flying in support of an armored cavalry squadron in Georgia. The Federal Aviation Administration is also interested in large scale virtual environments in order to train the complete system of air controllers, aviators, and equipment. Boeing is building a large scale virtual environment for its B777 airliner, the first modern aircraft built without a mock-up, to speed the process of concurrent engineering. AT&T, Sega and other companies have already publicly expressed their desire to construct distributed virtual environments for entertainment applications.

B. Challenges

The question we now have to answer is: how to do we build such environments?

The good news for those interested in developing distributed virtual environment is that advances in computer architectures and graphics, as well as standards such as the DIS and SIMNET protocols have made small scale (less than 300 players) realistic simulations possible [Miller89].

The bad news is that, until recently, the network software components of virtual environments has often been overshadowed by issues such as human interface design or graphics realism. This has led to the fact that there currently are not any good solutions to this hard networking problem. In a hypothetical example, in schemes such as SIMNET, 100,000 players could require 375 Mbit per second (Mbps) of network bandwidth to each computer participating in the simulation, an unrealistic requirement for an affordable system in this decade [Lora92].

Even commercial systems are finding the limits to current technology in creating networked environments for large numbers of users. American Online (AOL) provides low-bandwidth multi-user games, messaging, and chat services for over 600,000 subscribers, yet has been forced to block new users because customer network demand has overwhelmed AOL's host computer. As one observer put it - "America Online's problems are not trivial" [Merc94]. AOL will likely delay introduction of new services to dampen growth - not a good sign for proponents of the information superhighway.

Another key challenge is that the appropriate systems involving human operators must deliver packets with minimal latency (less than 100 ms) and generate textured 3D graphics at 30-60 Hz to guarantee the illusion of reality. On top of this is the need to provide real-time audio, video, and imagery services for the simulation of player communication services.

The result is that we need to address the "technology formula" for scalability, as James Chung has referred to in a study of enabling technologies for large scale distributed simulation [Chun92]. The formula involves the host processor processing and bandwidth, local area network speeds, and wide area network switching and bandwidth. In addition, our own experience and research with

previous versions of NPSNET has also caused us to conclude that the development of a software architecture for a real-time distributed environment requires a multi-faceted approach to tackle these issues while adding an additional goal: reducing the per-person cost of the simulation [Pratt93].

C. Software Testbed for Solutions

The NPSNET-IV networked virtual environment, developed at the Computer Science Department of the Naval Postgraduate School (NPS) in Monterey, California [Zyda94] is our primary experimental testbed in this effort. Furthermore, we have been exploring new technologies such as multicast networks, constructing methods for substantially reducing bandwidth requirements with distributed simulation protocols, and designing our software to exploit parallel computing architectures.

The NPSNET Research Group has devoted itself to exploring several functional areas of interactive simulation including:

-- application and network level communication protocols.
-- object-oriented techniques for virtual environment construction.
-- hardware and operating system optimization.
-- real-time physically-based modeling (e.g. smoke, dynamic terrain, and weather).
-- multimedia (audio, video and imagery).
-- artificial intelligence for autonomous agents or entities.
-- integrating robots into virtual worlds.
-- human interface design (stereo vision, system controls).

NPSNET-IV is unique in distributed simulation because it incorporates all of the following in an operational visual simulator to study the above areas:

1) **Distributed Interactive Simulation (DIS 2.03) protocol** for application level communication among independently developed simulators (e.g. legacy aircraft simulators, constructive models, and real field instrumented vehicles).

2) **IP Multicast**, the Internet standard for network group communication, to support large scale distributed simulation over internetworks.

3) **Heterogeneous parallelism** for system level pipelines (e.g. draw, cull, application, and network) and for the development of a high performance network software interface.

II. Taxonomy

Before continuing discussion of our approach we need to develop our perspective for distributed virtual environments. Virtual environments mimic many aspects of operating systems. For example, the system developed by the Human Interface Technology Laboratory, Virtual Environment Operating Shell (VEOS), provides much of the functionality of a distributed operating system in way of programming language services [Kalaws93][Brick93].

However, we are still primarily concerned with a single application and not a general purpose computing environment. Therefore, the most important questions about virtual environment software architectures we address here are:

What is distributed?

What are the modalities of the distribution?

Why is it distributed?

A. Communication

Several aspects of communication are largely responsible for answering the questions above. First, the available network bandwidth determines the size and richness of a virtual environment because as the number of participants increases so do the bandwidth requirements. On local area networks (LANs), this is not a major issue because technologies such as Ethernet (10 Mbps) are relatively inexpensive and the number of users is often limited. In contrast, wide area networks (WANs), bandwidth is generally limited to T1 (1.5 Mbps) but the potential user base is much larger. Some distribution schemes scale better than others. We explore this issue later when discussing multicast and internetworking.

Second, latency controls the interactive and dynamic nature of the virtual environment -- how well the players mesh in behavior. If a distributed environment is to emulate the real world, it must operate in real-time. This becomes a major challenge in systems that use wide area networks because of delays induced by long paths, switches and routers.

Finally, communications reliability often forces a compromise between bandwidth and latency. Reliability means that systems can logically assume that data sent is always received correctly and thus obviating the need to periodically re-send the information. Unfortunately, to guarantee delivery, the underlying network architecture must use acknowledgment and error recovery schemes that can introduce large amounts of delay - a common case on WANs and with large distributed systems. Additionally, some transport protocols such as TCP use congestion control mechanisms that are unsuitable for real-time traffic because they throttle back the packet rate if congestion is detected.

B. Views

Views are the windows into the virtual environment from the perspective of the people or processes who use it. We define two kinds of useful views for distributed environments. The first one is the synchronous view. An example of this is in a distributed flight simulator where one machine controls the forward image, and two other hosts each process the left and right cockpit window perspectives. The images are coordinated to give the illusion that they are all part of single cockpit view. Synchronism requires both high reliability and low latency. Therefore, virtual environments that require synchronous views are for practical reasons restricted to local area networks. An example of such an environment is the RAVEN simulator developed by Southwest Research Institute for NASA synchronizes Shuttle astronaut viewpoints which are rendered on different machines to improve rendering performance. [Cater94]

The second and most general concept is the asynchronous view. In this paradigm, multiple users have individual control over when and what they can see in the virtual environment concurrently. Participants can be physically separated over a local area network or a wide area network. Their awareness of each other's presence, if they are represented by an object, is brought about *inside* the virtual environment. NPSNET uses the asynchronous model where each view is typically that of the simulated entity. Views not associated with an entity are often referred to as "magic carpets" or "stealth" vehicles. Stealth entities do not need to communicate with the distributed world because there is no need for the world to have knowledge of the viewer.

C. Data

Perhaps the most difficult decision of building a distributed environment is determining where to put the data. One way is to initialize the state of every system participating in the distributed environment with a homogeneous world database containing information about the terrain, models, and behavior of all that is represented in the virtual environment. Communicated among all the users of the environment are object state changes such as vehicle location or events such as the detonation of a simulated missile or collisions between two objects. The advantage of this approach is that messages are relatively small. The disadvantages are that it is relatively inflexible and that as virtual environment content increases so must everyone's database. This is the model for SIMNET, a distributed military simulation system for use on Ethernet LANs developed by BBN for ARPA. However, once a simulation begins, each host maintains its own database without making any effort at guaranteeing consistency [Pope89].

On the other hand, the Virtual Space Teleconferencing System (VISTEL) uses a static shared world database. As its name implies, VISTEL is a teleconferencing system that displays 3D models of each conference participant. Changes in a model's shape, reflecting changes in a person's facial expression, are sent via messages to a central server and redistributed. Only one user at a time can modify the database [Ohya93].

DIVE also has a homogeneous distributed database. However, unlike SIMNET the entire database is dynamic and uses reliable multicast protocols to actively replicate new objects. A disadvantage with this approach is that it is difficult to scale up because of the communications costs associated with maintaining reliability and consistent data. For example, modelling terrain interactions, such as building a berm, still would be very expensive (though highly desirable) in terms of the number of polygons that would need to be created, changed, and communicated in DIVE [Carls93].

Another technique is to use the client-server model in which the database is partitioned among clients and communication is mediated by a central server. For example, in BrickNet, as an entity moves through the virtual environment, its database is updated by an object-request broker on a server that has knowledge of which client maintains that part of the world [Singh94]. BrickNet maybe most appropriate for large CAD environments because it attempts to tackle the "Boeing" problem of a virtual environment that has huge numbers of component models and provides multiple views simultaneously to a group of users. However, in a dynamic large scale world, the servers can quickly become I/O bottlenecks (ala AOL), increasing the inherent latency of the virtual environment. The developers of BrickNet have suggested some possible solutions including providing multiple distributed servers.

D. Processes

Distributing processes to multiple hosts increases the aggregate computing power associated with a simulation. We can use this not only to provide the capability to distribute views but also handle a variety of input devices. SIMNET and its descendants, such as DIS-compliant systems, also make use of the aggregate computing power by taking advantage of a technique called dead-reckoning, discussed in detail later, to reduce the need for network communication.

Processes can be homogeneous or heterogeneous. With the exception of the DIS model practically all distributed environments assume that the same kind of processes are running on each host that has the same function (architectures may differ). The advantage of this approach is consistency. The disadvantage is that it is very inflexible. DIS is a protocol designed so that

different developers can create different simulations on different machines that theoretically can share in the same virtual environment. The problem with this is that no protocol is complete and DIS is not an exception. More on this when we come to discuss DIS specifically.

III. NPSNET-IV

NPSNET has evolved significantly since its early origins as a distributed environment. Previous incarnations of NPSNET (I and II) used a locally designed network scheme that required Ethernet as the local area network protocol and incorporated an ASCII-encoded application level protocol to convey the simulation state. The packet, or (in ISO terminology), application protocol data unit (PDU) lengths were disproportionately long for the amount of information they contained and they did not comply with any particular standard. Moreover, the application protocol did not use any internetworking protocol, therefore it restricted use of NPSNET to the local LAN segment and a single network technology (Ethernet).

Another early developmental effort of NPSNET was the NPSStealth. NPSStealth was a version of NPSNET that integrated a translator for the BBN developed SIMNET protocol for interaction over local and long-haul networks. The inclusion of the SIMNET protocol enabled NPSStealth to participate in distributed simulations with other simulators that used the SIMNET protocol.

These efforts provided our group with a substantial amount of experience in designing a distributed virtual environment. However, these early versions were not intended for large scale simulations. In order to develop such a system, we needed to adopt the IEEE 1278 DIS protocol for interoperability with other simulators and create a scaleable software and network architecture [IEEE93].

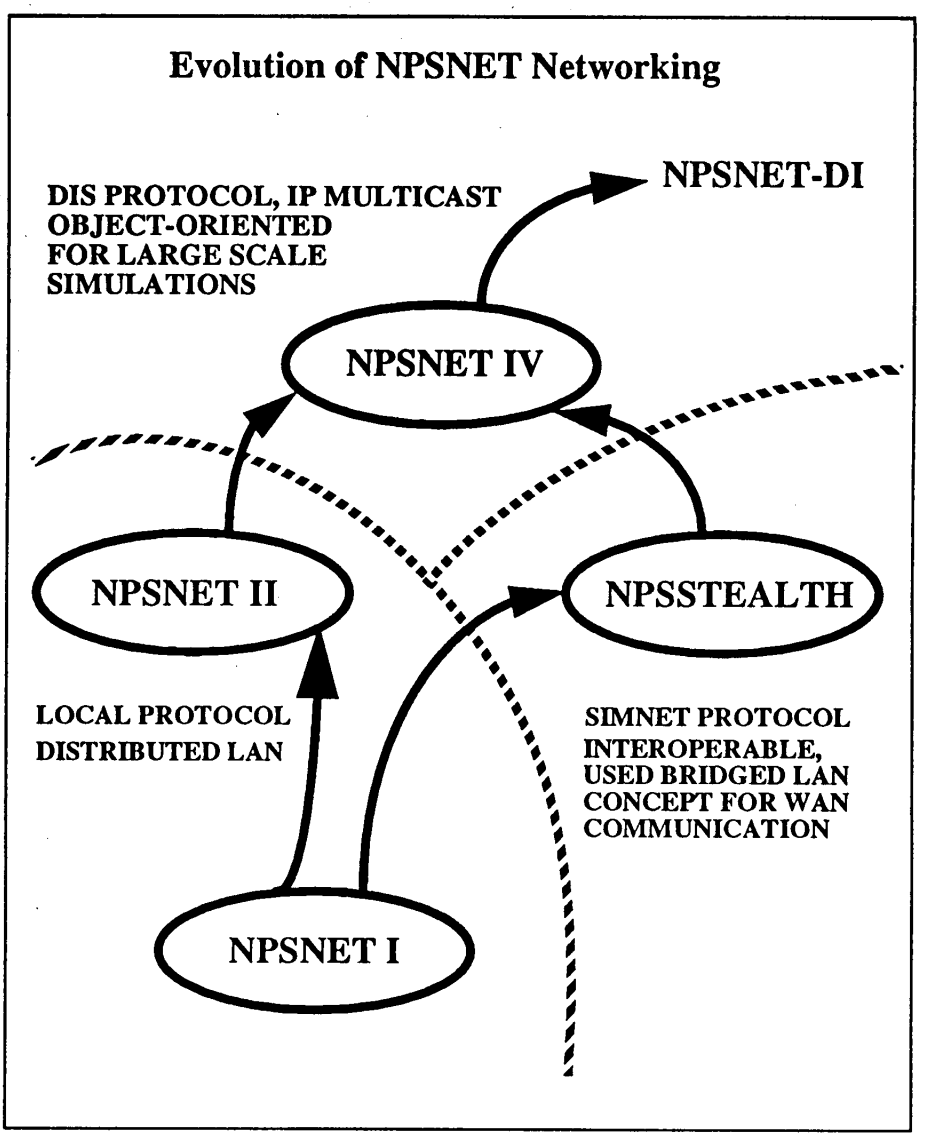

Figure 1. Evolution of NPSNET Networking.

NPSNET-IV can be configured by the user as a simulator for an air, ground, nautical (surface or submersible) or virtual vehicle. (A virtual vehicle is a non-invasive entity that maneuvers in the simulated world but is not represented by a model - a stealth vehicle.) The user controls the vehicle by selecting one of three interface devices which include a flight control system (throttle and stick), a six degree of freedom SpaceBall, and/or a keyboard. The system models vehicle movement on the surface of the earth (land or sea), below the surface of the sea, and in the atmosphere. Other vehicles in the simulation are controlled by users on other workstations. These users can either be human participants, rule-based autonomous entities, or entities with scripted behavior.

The system is automatically configured for distributed network operations at start-up. In this mode, the operator interacts with other human participants who are "piloting" other vehicles. The virtual environment is populated with vehicles operated by players at NPS and other participants

from around the country; and a variety of static and dynamic objects which exhibit numerous visual and multimedia behaviors including sound. For example, flying by a farm evokes a chorus of animal noises.

IV. DIS PROTOCOL

Large scale virtual environments will likely be composed of many different and unique hardware and software platforms developed independently. The DIS protocol, which replaced the SIMNET protocol in NPSNET, has the distinct advantage that it attempts to provide a basis for which to tie those systems together. DIS is a group of standards being developed by the Department of Defense and industry that address communications architecture, format and content of data, entity information and interaction, simulation management, performance measures, radio communications, emissions, field instrumentation, security, database formats, fidelity, exercise control and feedback. A second purpose is to provide specifications to be used by government agencies and engineers that build simulation systems. The effort is also meant to define the terminology of DIS [IST93a]

A. Protocol Data Units

Simulation state and event information is conveyed by twenty-seven PDUs defined by the IEEE 1278 DIS standard, but only four of these are for entity interaction. The remainder of the PDUs are for transmitting information on supporting actions, electronic emanations, and for simulation control [IST93].

The Entity State PDU is used to communicate information about a vehicle's current state, including position, orientation, velocity, and appearance (Table 1). The Fire PDU contains data on any weapons or ordnance that are fired or dropped. The Detonation PDU is sent when a munition detonates or an entity crashes. The actual structure of a PDU is very regimented and is explained in full detail in [IST93].

Field Size (Bytes)	Entity State PDU Fields	
12	PDU Header	Protocol Version. Exercise ID. PDU Type. Padding. Time Stamp. Length in bytes.
6	Entity ID	Site. Application. Entity.
1	Force ID	
1	Number of Articulation Parameters	N.

Field Size (Bytes)	Entity State PDU Fields	
8	Entity Type	Entity Kind. Domain. Country. Category. Subcategory. Specific. Extra.
8	Alternative Entity Type	Same type of information as above.
12	Linear Velocity	X,Y, and Z (32 bit components).
24	Location	X,Y, and Z (64 bit components).
12	Orientation	Psi, Theta, Phi (32 bit components)
4	Appearance	
40	Dead Reckoning Parameters	Algorithm. Other Parameters. Entity Linear Acceleration. Entity Angular Velocity.
12	Entity Markings	
4	Capabilities	32 Boolean Fields.
N * 16	Articulation Parameters	Change. ID. Parameter Type. Parameter Value.

Table 1: Entity State PDU

B. Simulation Philosophy

The networking technique used in NPSNET-IV, evolved from SIMNET, and embodied in DIS follows the *players and ghosts* paradigm presented in [Blau92]. In this paradigm, each object is controlled on its own host workstation by a software object called a Player. On every other workstation in the network, a version of the Player is dynamically modeled as an object called a Ghost.

The Ghost objects on each workstation update their own position each time through the simulation loop, using a dead-reckoning algorithm. The Player tracks both its actual position and the predicted position calculated with dead-reckoning. An updated Entity State PDU is sent out on the network when the two postures differ by a predetermined error threshold, or when a fixed amount or time has passed since the last update (nominally 5 seconds). When the updated posture (location and orientation) and velocity vectors are received by the Ghost object, the Ghost's is corrected to the updated values, and resumes dead-reckoning from this new posture.

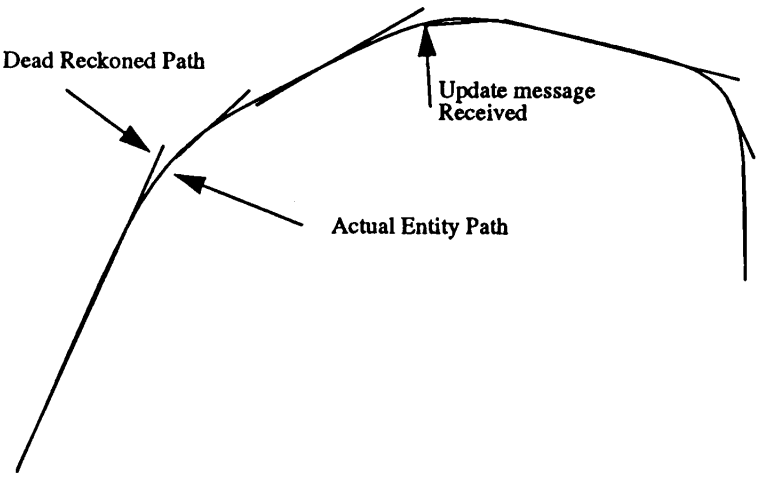

Figure 2. Error Based Dead Reckoning and Actual Paths

The primary purpose of the Ghost in the machine concept is to reduce network traffic by minimizing updates at the cost of extra computation by the host system. Analysis of this method has confirmed that it works well with a wide variety of simulated vehicles with different performance characteristics, including high-speed aircraft. Harvey and Shaffer showed that even with a 3 meter position error threshold, first order dead-reckoning algorithms were sufficient to model an F16 with a 66% decrease in network traffic -- from 50 Hz to 17 Hz for Entity PDU updates [Harv92]. Alternatively, an updated PDU could be sent after every frame. However, even at 10 Hz frame rate this is wasteful - 11520 bits per second (bps) for one entity - when considering that in a large simulation many entities are stationary.

Figure 3 shows how dead-reckoning works when modeling munitions. In this case, when 5 bombs are dropped in NPSNET-IV, 5 Fire PDUs are generated followed by the instantiating of 5 Entity PDUs representing each bomb. Initially, the bombs are horizontal and they rotate to point downward. As the bombs change orientation, the dead-reckoning thresholds are exceeded. Less time is required for updates as the bombs maintain their orientation. They hit the ground and a Detonate PDU is issued for each one and they stop sending Entity PDUs.

Figure 4 demonstrates the advantages of the Ghost technique. It traces the number of PDUs per second generated by 9 simulated aircraft in a highly dynamic situation (everyone was trying to shoot each other down). Even the spike of approximately 76 PDUs per second is half the bit rate expected of a system not using dead-reckoning. Typical bandwidth used by one entity in this simulation is approximately 3200 bps. The graph also shows another characteristic of DIS traffic induced by dead-reckoning -- very bursty and not easily modeled.

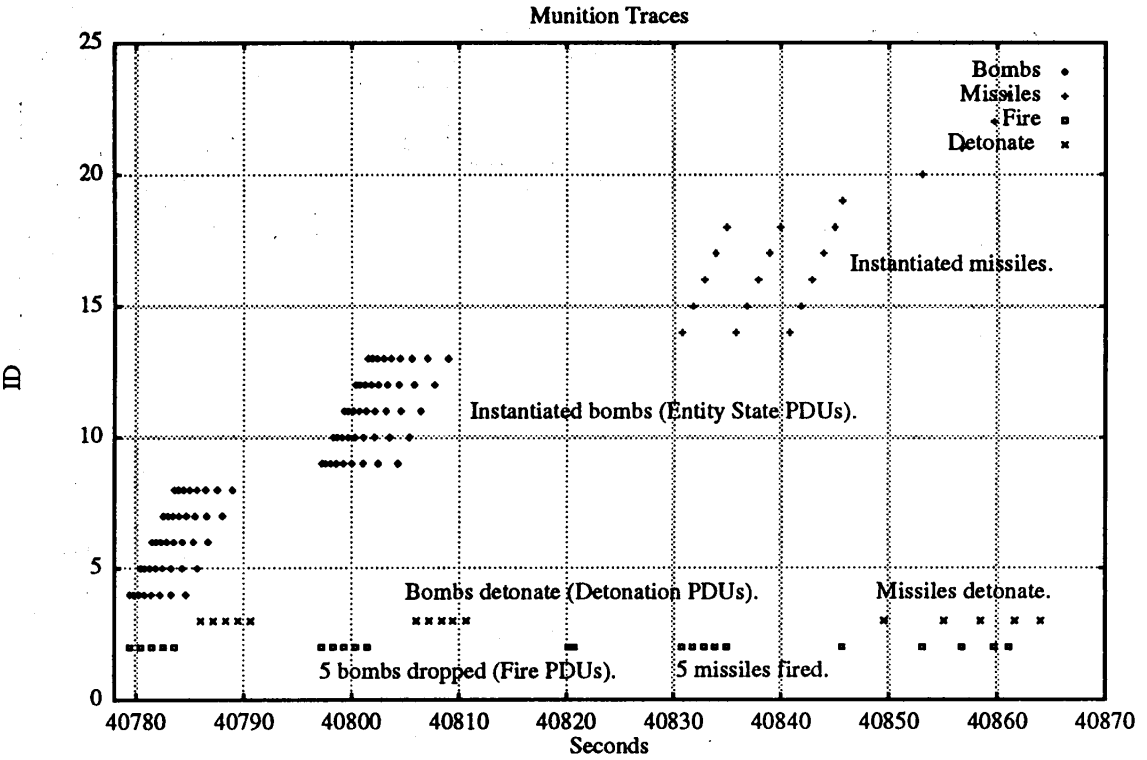

Figure 3. Time Traces of PDU Events

DIS also inherited two other concepts from SIMNET that influence bandwidth and reliability. First, with the exception of resupply operations, there are no transactions among entities about their state. Every entity is presumed honest. For example, when a vehicle fires a weapon, it is the determiner of who was hit and communicates that to all other entities. The targeted entities inform everyone including the shooter what damage occurred. In this manner, only one entity must make a ballistic computation and only the targets must compute the effects; minimizing overall simulation computation and avoiding the need to consume bandwidth and simulation time by negotiating 'who shot who'.

Secondly, there is no central database therefore, an entity has to 'learn' about the world by state updates from other entities. Entities that are not doing anything (e.g. parked vehicles) still must send an update packet, as mentioned above, even though their state does not change.

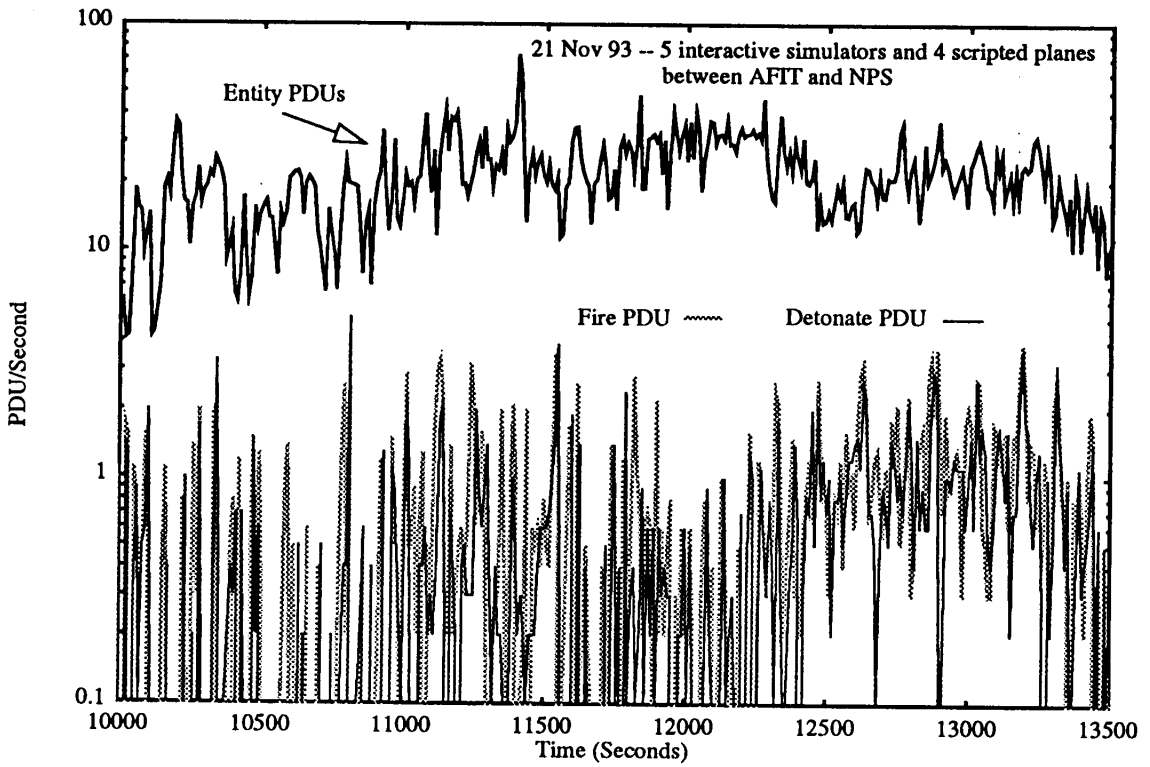

Figure 4. PDU Rates (10 Second Sample Rate)

C. NPSNET DIS Performance Values

NPSNET-IV is capable of generating one PDU at a peak rate of one per frame. For example, at thirty frames per second it can generate thirty packets per second. Weapons are restricted to one firing per second. Upon a firing, the simulation generates two additional packets: one Fire PDU to establish the firing and one Entity State PDU to model the munition (Figure 3). Upon munition impact, one Detonation PDU is transmitted. However, during past experimentation, we have observed a peak rate of twelve Entity State PDUs per second for the aircraft. The packet rate was constrained by dead reckoning thresholds. In addition to the twelve packets per second, the aircraft simulator generated at most three packets for a weapon firing sequence totaling fifteen packets per second. The peak packet transmission rate for this type of simulator is fifteen packets per second [Zesw93].

As can be seen in Figure 4, Entity State PDUs dominate DIS network traffic. Figure 5 shows how these rates, along with bandwidth, affect the number of simultaneous users that can participate in the distributed virtual environment using NPSNET-IV. The bandwidths are representative of current communications technology -- 28 Kbps for high speed modems, 1.5 Mbps for T-1 lines, and 10 Mbps for Ethernet (though Ethernet becomes saturated at roughly 70% utilization). Note that this is a simple model of performance. It assumes (falsely) that traffic would be perfectly distributed and that hosts and routers could actually process the PDUs without delay at the higher rates. We discuss these issues and provide some experimental data later. However, the graph is provided to give a "ball-park" feel for the entity-bandwidth-PDU relationship. With NPSNET-IV

we approach a maximum of 300 players on an Ethernet LAN without modification to the DIS protocol.

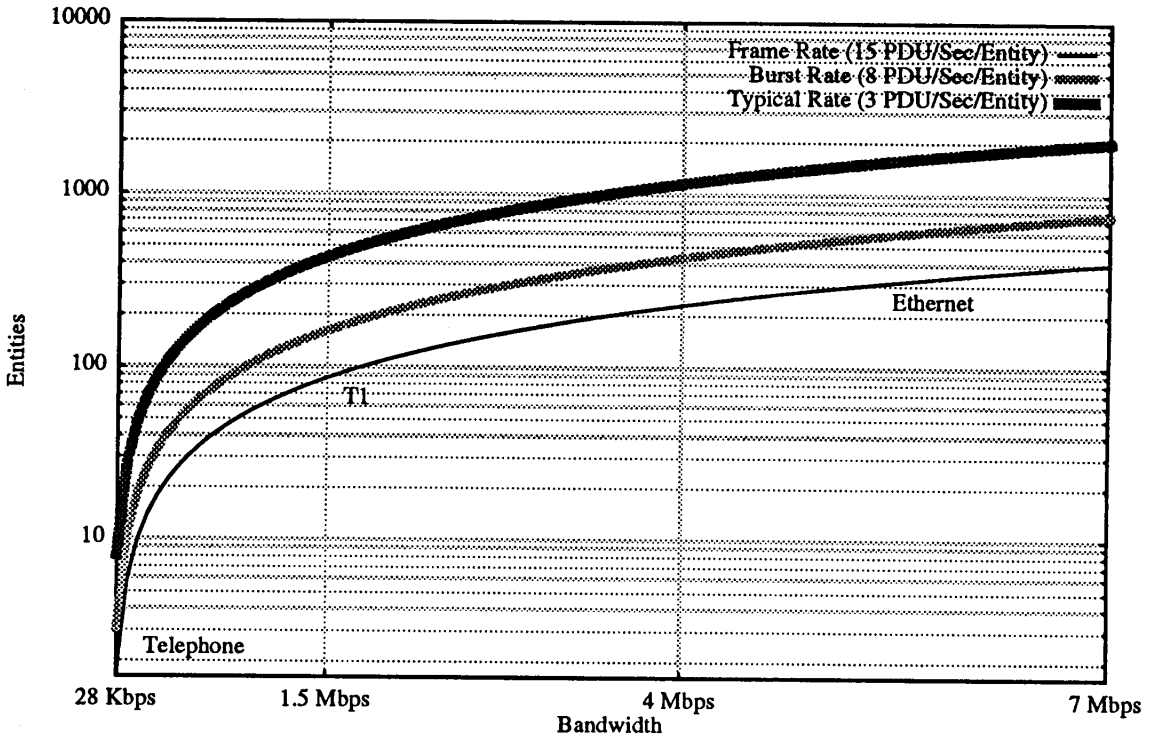

Figure 5. Bandwidth vs. Number of Players

D. Problems with the DIS Protocol

IEEE 1278 is still an immature protocol and is very much a work in progress. DIS involves and affects so many users and developers that it embodies many different and sometimes competing ideas. This section outlines some of those problems.

For example, DIS has struggled with the important question of how to adapt to different demands for realism in simulations. One community is interested primarily in using DIS for *training* large numbers of soldiers on a virtual battlefield while another sees it as a way to *test* new systems. Unfortunately, the requirements for precision and minimal latency are much greater in a testing environment leading to specifications that would be considered somewhat excessive in a training simulation such as using 64 bit floating point numbers.

Furthermore, the protocol's SIMNET heritage has had several negative consequences primarily because SIMNET was designed for a local area network with small numbers of players (less than 50). SIMNET's reliance on broadcasting PDU's over bridged networks (with many attendant problems -- discussed later) is still the most common mode of communication for DIS. DIS also avoids centralized databases or servers of any kind. Therefore, the protocol requires that *all* the data about an entity be transmitted when its status changes.

The result is that much of the data sent is redundant (e.g. the markings on the aircraft). It remains to be proved that the redundant data in the packet is of much consequence because, in

general, they are typically small (144 bytes). Rather, the major concern of developers are the number of packets per second. The packet rate is affected by the number and activity (state changes) of the entities, and the type of dead-reckoning algorithms being used.

However, a major influence on the PDU rate is that late arriving players have no idea of what has transpired so the only way they can learn is to listen for a while and accumulate knowledge from the Entity PDUs being issued. Therefore, the DIS protocol requires that Entity PDUs be sent after a predetermined time *even though their state has not changed.* How much time? If the time is too long then new entrants will operate for that period without complete knowledge of the world and realism is lost (e.g., being shot by vehicles they did not know existed but, should have known). Update too often and a major burden is added to the network in the form of additional PDUs per second. Moreover, it is presumed that in large simulations many of the entities will be static (e.g. missile batteries or parked tanks).

Unfortunately, we do not know for sure. Failing to properly test ideas has been a regrettable hallmark of DIS development. Contrary to practices of standards organizations such as the Internet Engineering Task Force (IETF), there is no requirement to have working tested implementations before the IEEE 1278 standard is changed. This results in the adoption of ideas such as the Signal PDU which is used for simulating radio traffic by sending voice or data between participants. This may have several unintended consequences such as burdening the distributed simulation application to perform a task better suited for real-time voice such as Van Jacobson's and Steven McCanne's Visual Audio Tool (vat) or to supplant protocols that have enjoyed years of testing such as the Real Time Protocol (RTP) [Casner93b].

Furthermore, DIS lacks a "middleware" layer - a software layer that mediates between the distributed applications and the network. A problem with this is demonstrated by the fact that models and world databases must be replicated at each simulator because no mechanism exists like BrickNet's object exchange to distribute objects on demand. For large scale simulation, this is a necessity, particularly when the simulators are heterogenous, controlled by different organizations, and little coordination is expected prior to an exercise. Furthermore, it is not feasible nor efficient for each simulator to store every model and database for a 100,000 entity simulation. For example, an F15 simulator normally does not need to concern itself with naval vessels -- unless some unique scenario has it going over the ocean.

This leads to the unresolved issue of communicating dynamic changes with respect to normally static objects or the environment. Objects could be clouds or static entities like bridges and buildings that may change with respect to an event (e.g. an explosion). Representing realistic views of the weather consistently and efficiently on a large number of heterogenous systems is a major and possibly intractable challenge.

Another, insidious issue is caused by the fact that DIS assumes every simulator to be truthful. An unintended side effect, in a large scale heterogenous simulation, is that the quality of realism expressed by simulators may vary by wide margins, reducing the overall realism of the simulation. For example, a highly realistic and accurately portrayed F15 simulator would not compete favorably against a simulator that is only functionally realistic but without the difficult flight dynamics of a high performance aircraft [Harris93].

However, these are just a few of the many issues regarding the DIS protocol. Several organizations, including the NPSNET Research Group, are examining ways to improve or restructure the protocol, operational concept, and physical network architecture.

V. Multicast

NPSNET-IV is the first DIS application to use the IP Multicast protocol. Multicast network services are necessary to build large scale distributed virtual environments. Multicast promises to provide an efficient mechanism for partitioning player groups in virtual environments to optimize the computational load on the simulation hosts and possibly reduce bandwidth requirements on network 'tail links' extending from the backbone [Doris93].

Multicast services allow arbitrarily sized groups to communicate on a network via a single transmission by the source [Perl92]. Multicast provides one-to-many and many-to-many delivery services for applications such as teleconferencing and distributed simulation in which there is a need to communicate with several other hosts simultaneously. For example, a multicast teleconference allows a host to send voice and video simultaneously to a set of (but not necessarily all) locations. With broadcast, data is send to all hosts while unicast or point-to-point routes communication only between two hosts.

Most other distributed simulations have employed some form of broadcast (hardware-based or IP) or point-to-point communications. For example, the MR Toolkit Peer Package, which is used for creating distributed virtual reality applications over the Internet uses unicast for communications among the applications though the developers have considered using IP Multicast [Shaw93].

However, these schemes are bandwidth inefficient and broadcast, which is used in SIMNET and most DIS implementations, is not suitable for internetworks. Moreover, IP broadcast requires that all nodes examine a packet even if the information is not intended for that host, incurring a major performance penalty for that host because it must interrupt operations in order to perform this task at the operating system level. (SIMNET used the hardware multicast capability of Ethernet but only to create a single multicast group for the entire distributed simulation.) Point-to-point requires the establishment of a connection or path from each node to every other node in the network for a total of N*(N-1) virtual connections in a group. For example, with a 1000 member group each individual host would have to separately address and send 999 identical packets.

A. Internetworking

Internetworking (routing over the network layer) is important for large scale simulations because it provides the capability to:

-- use commercial services as opposed to private networks to bring together diverse, geographically dispersed sites.
-- use different local network topologies and technologies (e.g. Ethernet and FDDI).
-- take advantage of "rich" topologies for partitioning bandwidth, providing robustness and optimization of routes for minimizing latency.

Confining DIS to the data link layer would require the use of bridges which:

-- are a magnitude slower to reconfigure after a topological change than routers.
-- the number of stations are limited to the tens of thousands while with routing it is limited to the numbers accommodated by the address space [Perl92].

We have built NPSNET-IV on top of IP Multicast to provide dynamic group communication services and internetworking to the application. IP Multicast, developed by Steve Deering, is an IETF standard (RFC 1112), and is supported by a variety of operating systems including SGI's Irix and Sun's Solaris [Deer89] [Comer91]. Through the use of multicast routers, NPSNET can communicate across the Internet via the MBONE.

B. MBONE

MBONE stands for Multicast Backbone, a virtual network that originated from an effort to multicast audio and video from the Internet Engineering Task Force (IETF) meetings [Casner93a]. MBONE today is used by several hundred researchers for developing protocols and applications for group communication.

Multicasting has existed for several years on local area networks such at Ethernet and FDDI. However, with IP multicast addressing at the network layer, the service group communication can be established across the Internet. Categorized officially as an IP Class D address, an IP multicast address is mapped to the underlying hardware multicast services of a local area network.

However, the reason that MBONE is a virtual network is that in most cases it shares the same physical media as the Internet, though it must use a system of parallel routers that can support multicast (i.e. Sun workstations) augmented with 'tunnels'. Tunneling is a scheme to forward multicast packets among the islands of MBONE subnets through Internet IP routers which, for the most part, do not support IP multicast. This is done by encapsulating the multicast packets in regular IP packets.

When a host on a subnet establishes or joins a group, it announces that event via the Internet Group Management Protocol (IGMP). The multicast router on the subnet forwards it the other routers in the network. Groups are disestablished when everyone leaves, freeing up the IP multicast address for reuse. The routers occasionally poll hosts on the subnets to determine if any are still group members. If there is no reply by a host, the router stops advertising group membership to the other multicast routers [Comer91].

The routing protocols for MBONE are still immature and a part of the central experiment of the network. Most of the MBONE routers employ the Distance Vector Multicast Routing Protocol (DVMRP) which is commonly considered inadequate for rapidly changing network topologies because routing information propagates too slowly. A multicast extension to the Open Shortest Path (MOSPF) link-state protocol has been proposed that addresses this problem [Moy93]. However, with both protocols each router must compute a source tree for each participant in a multicast group. MBONE is now small enough that this is not a problem. However, for a large network with constantly changing group memberships, these routing techniques could be computationally inefficient [Perl92]. However, the success of MBONE is spurring more research into providing for this possibility. For example, Cisco Systems, a major router vendor, intends to include IP Multicast routing capability in the near future using the Protocol- Independent Multicast (PIM) protocol being considered by the IETF Internet Domain Multicast Routing group.

A further problem is that the topology and the multicast sessions must now be actively managed by the MBONE community to minimize congestion. MBONE is currently experimenting with automatically pruning and grafting subtrees but, for the most part uses truncated broadcasts to the leaf routers. The truncating is based on the setting for the time-to-live (ttl) field in a packet which is decremented when it passes though a router. A ttl value of 16 would limit multicasts to a

campus versus a value of 100 which could send it to every subnet on the entire MBONE (and at least twelve countries).

However, we have used MBONE to demonstrate the feasibility of IP Multicast for distributed simulations over a wide area network. In the past, participation with other sites required prior coordination for reserving bandwidth on the Defense Simulations Internet (DSI). DSI, funded by ARPA, is a private line network composed of T-1 (1.5 Mbps) links, BBN switches and gateways using the ST-II network protocol. It has been necessary to use DSI because ARPA sponsored DIS simulations used IP broadcast - requiring a unique wide area bridged network.

With the inclusion of IP Multicast, sites connected via the MBONE can immediately participate in a simulation. MBONE uses a tool developed by Van Jacobson and Steven McCanne called the Session Directory (*sd*) to display the advertisements by multicast groups. SD is also used for launching multicast applications like NPSNET-IV and for automatically selecting an unused address for the new group. Furthermore, we have integrated other multicast services such as video with NPSNET-IV; for example, allowing participants to view each others simulation with a derivative of a video tool, nv, developed by Ron Fredrickson at Xerox Parc and Stephen Lau at SRI [Casner93a].

C. Rationale

IP Multicast uses the User Datagram Protocol (UDP) and therefore, by itself, is an unreliable, connectionless service. However, reliable multicast protocols are currently not practical for large groups because in order to guarantee that a packet is properly received at every host in the group an acknowledgment and retransmission scheme is required [Part94]. With a large member distributed simulation, reliability, e.g. as provided in the Transmission Control Protocol (TCP), would penalize real-time performance merely by having to maintain timers for each host's acknowledgment. Flow control is also not appropriate for DIS which can recover from a lost packet more gracefully than from late arrivals.

This does not mean that researchers are not trying to develop both a reliable and *scaleable* multicast service. The communication for DIVE is provided by ISIS, developed by Ken Birman, which uses reliable multicast to guarantee that the virtual environment databases are accurately and synchronously replicated [Carls93]. (A recent version of ISIS implements a reliable transport layer on top of IP Multicast). However, a peer group with more than 20 or 30 members is about as large as can be efficiently supported [ISIS92]. Brian Whetten and Simon Kaplan have recently developed the Reliable Multicast Protocol (RMP) which is based on a token ring protocol that sits atop IP Multicast. They claim that RMP should scale to hundreds of users across the Internet [Whet94].

Fortunately, within the context of DIS, a certain amount of unreliability is tolerable and is mediated through the use of the dead-reckoning and smoothing algorithms and by the fact that entity postures are frequently updated [Harv92] [Lora92]. Other applications such as packet voice and video can use adaptive techniques to handle lost packets and delays [Part94].

If a truly scaleable reliable multicast protocol was developed then much of the redundant communication caused by periodic entity updates and over-descriptive PDUs could be eliminated. Moreover, using DIS for system development testing would become a reasonable concept since we could guarantee the reliability or ordering of data. For example, accurately finding out 'who shot who when' is important to system developers. We could possibly move DIS to a model that would allow us to observe and control this transaction atomically.

IP Multicast also allows the creation of transient multicast groups. Thus, it is suitable for developing virtual environments that associate geographical location, simulator services (e.g. environment servers), radio frequencies, or sensor capabilities with particular dynamic groups. For example, a virtual battlefield could be partitioned into grid squares where each square coincided with a single multicast group. When a player moved through this world and crossed into another grid, the simulated entity would join the new group.

As mentioned before, this partitioning is necessary to reduce the enormous computational requirements of large scale (100,000 player) simulations. For a 1000 object exercise conducted in 1990 with SIMNET, the limiting factor was not network bandwidth, with loads running at 50%, but the local host processor performance [Chun92]. Network simulations done by Van Hook of Loral have shown that a 90% reduction in traffic to a particular node is achievable for a 10,000 player exercise using multicast services [VanH93]. For the estimated peak bandwidth of 350 Mbps for a 100,000 player exercise, this translates into 35 Mbps peak and 10 Mbps for the "expected" average loads. We emphasize "expected" because no one really knows. No research has been done to characterize the traffic of distributed simulation other than some extrapolations of from small groups. The models used so far make simplistic assumptions like Poisson distribution, which are known to be inappropriate for bursty activities.

These are reasonable numbers for constructing a system with current (albeit expensive) technology. A system composed of T-3 (45 Mbps) WAN links and FDDI or ATM LANs could be a reality today. However, long distance carriers have been quietly installing Synchronous Optical Network (SONET) switches to provide ubiquitous support for those speeds. SONET is a US and international standard for optical signals, the synchronous frame structure for multiplexed digital traffic, and operations procedures for fiber optic switching and transmission systems. SONET allows lower speed channels such as OC-3 (155 Mbps) to be inserted and extracted from the main rate. SONET defines data transmission speeds to 2.4 Gbps. Both MCI and Sprint have announced that they will have SONET completely deployed by 1995.

D. ATM

New SONET switches will also incorporate asynchronous transfer mode (ATM) technology. ATM provides fast variable rate packet switching using fixed-length 53-byte cells. This permits ATM networks to carry both asynchronous and isochronous (video and voice) at SONET speeds. AT&T will offer its Interspan ATM service to over 300 locations by the middle of 1994 with access speeds initially ranging from 512 Kbps to 34 Mbps.

ATM will likely support multicasting and CCITT has written a standard for interfacing ATM with SONET. The IETF has also developed an interface standard for ATM and IP, though an interface to ATM's multicasting service has yet to be defined. However, several researchers are investigating how ATM could support IP Multicast [Wei92].

ATM may offer another significant advantages because it is a cell-based virtual network service offered by the major carriers. Multicasting will not necessarily reduce bandwidth requirements on the backbone. However, the aggregate bandwidth of a large scale simulation could be absorbed by the carriers backbone network. A 50,000 player distributed environment would roughly require, for simulation traffic alone, a 150 Mbps backbone, approximately the bandwidth of the standard ATM OC-3 rate.

ATM is also expected to provide on-demand quality of service guarantees in terms of latency and bandwidth that are not now available with most networks. This will be necessary for real-time

applications like distributed virtual environments. Finally, we believe that the DIS protocol could be optimized for ATM by making the PDU's conform to the 48 byte size of the ATM cell. For example, Danny Cohen, of Perceptronics, has suggested that much of the Entity State PDU could be reduced in size by eliminating redundant information such as the host and site identification which is already contained in the IP address of the sender. Some other ways to reduce the size include:

-- use canonical versus hierarchial representation of the entity types, and appearances.
-- reducing precision. For example, DIS using 64 bit floating point numbers for coordinates which give an accuracy of 0.2nm in a 500 KM box -- as Cohen points out, a bit of over-kill [Cohen93].
-- eliminate the use of the Entity State PDU to communicate markings (use some other means to do this).
-- use a single dead-reckoning algorithm.
-- eliminate guises (alternate identities).
-- eliminate the capabilities field. Only the first 4 bits are defined and they should be part of the entity type.

More importantly, current computer architectures (particularly with respect to the memory and system interfaces) such as SGI's can accommodate T-3 (DS3) bandwidths through the reduced packet rates brought about by multicasting. This was demonstrated by the XUNET testbed which developed an ATM host interface for the SGI Indigo VME bus [Beren93]. Their software and interface was able to simultaneously send and receive at 40 Mbps TCP packets encapsulated in ATM cells through SGI Power Series hosts acting as routers. Moreover, by partitioning the groups, a host only needs to maintain state information for the entities in its particular group. We conjecture that 1000 entities in a highly dynamic simulation would form an appropriate size multicast group for 3D graphics workstation architectures before the end of this decade. This is another area requiring study.

We are currently researching the appropriate scheme and algorithms that exploit multicast - used as the DIS middleware - for this partitioning. Our future architecture must include a mechanism for abstracting information regarding players or entities based on their location and relationships with each other. Additionally, a means for mediating the fidelity of the simulation (much like the Level of Detail (LOD) function in computer graphics) is needed for entities that require global information like simulated intelligence platforms and fast aircraft, as well as for system management functions. Fidelity can be adjusted by aggregating information about entities on special servers or increasing the time period for updates on a special multicast channel or group. Finally, we are examining ways to provide a set of universal services to the virtual environment including synchronization, load balancing, simulation management, weather and hypermedia.

VI. Heterogeneous Parallelism

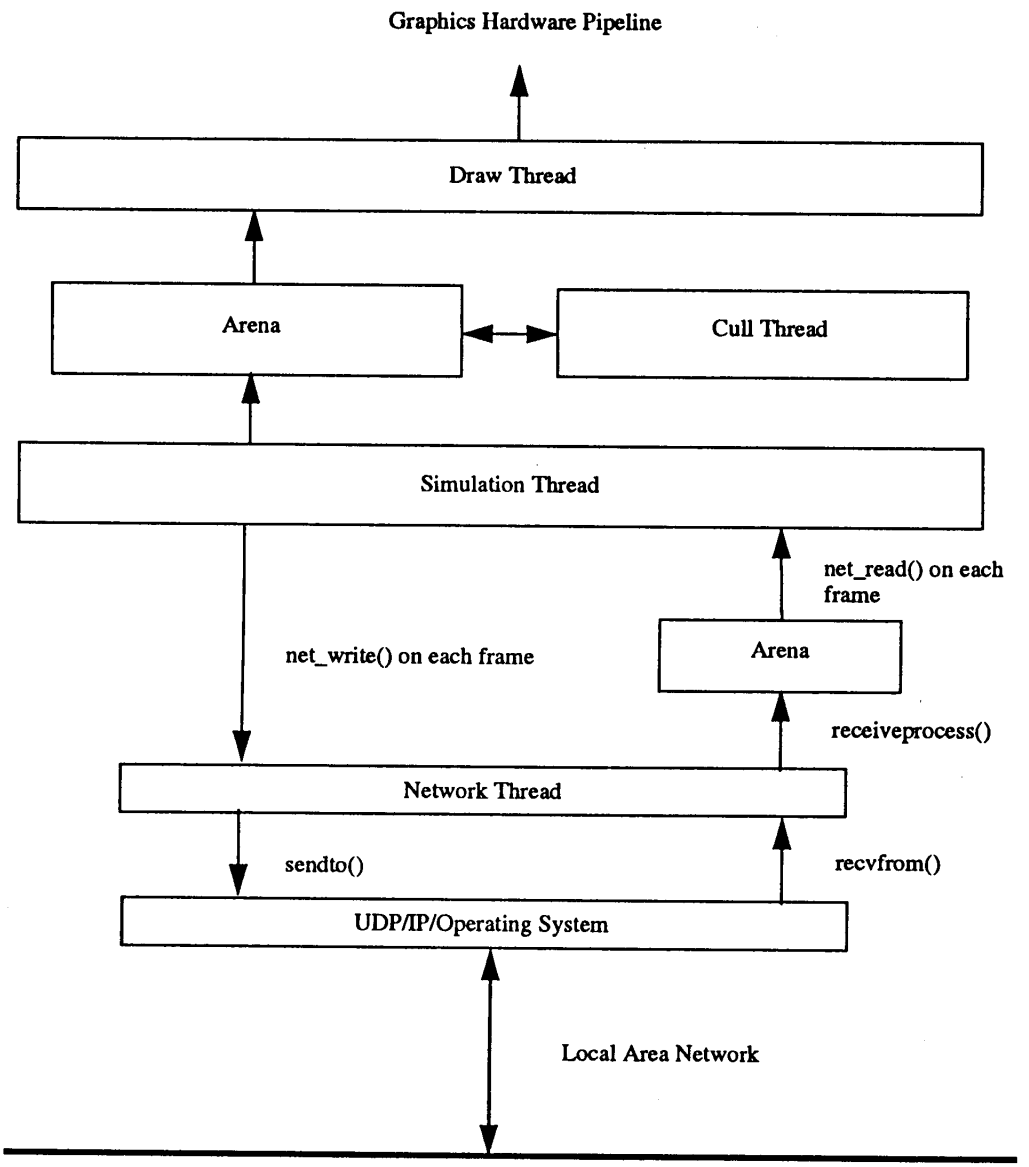

Figure 6. NPSNET-IV Software Architecture

Man-in-the-loop simulators function in real (wall-clock) time and are measured in the human perception time frame, which is approximately 100 milliseconds. Update information must be received by all other participating simulation hosts in sufficient time for reading the information from the network, updating the database, and rendering the display.

NPSNET-IV is designed to minimize system latency and maintain a frame rate (throughput) of at least 10 frames per second. To accomplish this we have taken advantage of the multiprocessor

architecture and heterogenous parallelism offered by the SGI Onyx computers with Reality Engine2 graphics and the Performer graphics library.

Exploiting parallelism to achieve high performance graphics is not a new concept and has in fact been incorporated into the Performer library. As an example, Akeley showed in 1989 that a six-fold increase in frame rates could be achieved using a four processor architecture instead of a uniprocessor system [Akeley89]. This is because out-of-view regions of a geometric database can be culled using spatial partitioning by one of the processors.

The effect is to reduce the number of polygons needed to be rendered by the drawing processor and eventually sent to the graphics hardware. In NPSNET-IV, the cull thread traverses the database, selects the geometry that is in the view volume, does Level of Detail selection, and constructs the display list for the draw function. In turn, draw renders the display list data structure.

We have expanded the use of parallelism to improve the *network throughput* as well as graphics throughput. Network processing makes significant demands for system resources, particularly within the context of large distributed environments. When a simulator receives data, a network frame is examined and copied from the interface to a system buffer by means of a direct memory access (DMA) block write. The operating system checks the IP and UDP headers to see if the packets are correctly addressed. For IP multicast, this may require demultiplexing among several addresses and destination application ports. The datagram is again copied by the CPU to the application memory space where it must be decoded by the network thread. The type of PDU and length are determined to allocate the appropriately sized structure in the arena.

With a four-processor machine, NPSNET-IV divides the draw, cull, application, and network threads into a system level pipeline using shared memory (IRIX arena's) for communication among the tasks. At the first stage, the network thread continuously reads the IP multicast receive socket and immediately writes new PDUs to an arena (Figure 6).

A. Advantages

The advantages of placing the network thread on its own processor are two-fold: it off-loads the work that would otherwise compete with the application, cull, or draw processes; and it immediately receives and buffers PDUs when offered by the network interface (rather than waiting for the other threads to complete) thus, reducing the probability that packets are dropped at the lower system network levels. For example, with a simulation frame rate of 10 Hz and an arrival rate of 1000 PDUs per second, approximately 15K bytes must be buffered by the network thread while the simulation loop performs the following:

-- updates the terrain database and the visual database maintained by Performer.
-- computes dead-reckoning and detonation effects.
-- takes user commands to the vehicle via flight control sticks or a spaceball.
-- updates the posture of the vehicle and computes the firing solutions for weapons.
-- writes new PDUs to the network process.
-- hands off to the cull process.

Note that the size required for this buffer (arena) is variable and dependent on the network load and the complexity of the scene rendered which influences the cycle time for frame updates.

B. Disadvantages

There are several disadvantages to the NPSNET-IV approach. First, the implementation is specific to SGI systems. Second, there is a penalty incurred by the extra data copying which causes more contention for memory, bus bandwidth, and the CPU which must do the copying. Another cost is that latency may increase in a particular stage with respect to the CPU buffering data. However, these problems are more than offset by the overall gains in system throughput and by the fact that a CPU is dedicated to the network data movement in a multiprocessor machine [Clark89]. Furthermore, the handling of DIS traffic at the application level will likely become more complicated as the protocol matures, demanding more processing resources. An example of this is that future DIS PDUs are expected to be aggregated within a UDP datagram to reduce data link and network level processing while further increasing the overhead of decoding by the receiving application.

VII. Demonstrated Results

We were able to demonstrate NPSNET-IV to a large audience at the SIGGRAPH'93 Tomorrow's Reality Group albeit using IP broadcast over a bridged wide area network for compatibility with other sites. We have also successfully conducted demonstrations and test of our multicast version over the MBONE.

A. SIGGRAPH 93

At SIGGRAPH, participants were able to operate in a three-dimensional virtual environment via local and wide area communication networks with other players using independently developed simulations at geographically dispersed sites. Our SGI hosts in Anaheim, California communicated locally through Ethernet and to other sites via a T-1 based private network. Other participating sites were the Air Force Institute of Technology (AFIT), Dayton, Ohio, the Simulation Center, Advanced Research Projects Agency (ARPA), Arlington, Virginia, the Naval Postgraduate School (NPS) in Monterey, California, and Naval Research and Development (NRaD) in San Diego, California. The connection with ARPA was unique in that it included simultaneous high-quality two-way video and audio over the network. The video from Arlington was displayed at the convention center on a Sony high-definition television using the Advanced Television (ATV) format -- while images from the simulation were shown at ARPA on a 15 meter "video wall".

Five sites were interconnected using the Defense Simulation Internet (DSI) and other leased T-1 facilities. NPS, AFIT, and the Naval Research and Development (NRaD) facility in San Diego, California were connected by DSI. ARPA was connected to NRaD using their own facilities. NRaD bridged the two networks to a leased T-1 link that terminated in the Anaheim Convention Center. Because the T-1 was multiplexed with video, simulation bandwidth was restricted to 704 Kbps.

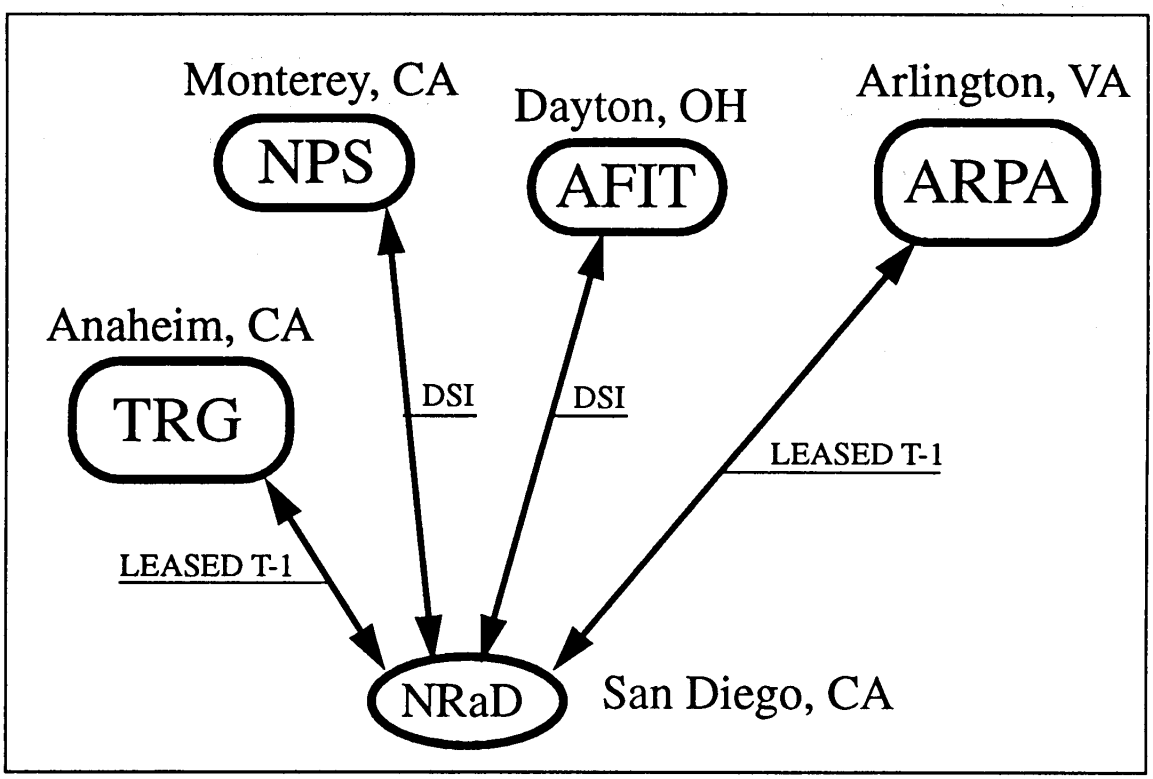

Figure 7. SIGGRAPH'93 Setup

A network load analysis was conducted during SIGGRAPH to approximate an upper bound of the number of simulation hosts and entities that can reliably participate on our Ethernet segment. The bandwidth required for the distributed simulation experiments was predictable based on the previous discussion. There was a maximum of eleven hosts simulating 50 entities during the free-play scenario. Seven high-performance aircraft were simulated while four of the simulators modeled multiple slow moving vehicles.

The average broadcast packet length was 190 bytes, including network overhead. Over 90% of the traffic was represented by Entity State PDUs. Simulation traffic peaked at 168 packets per second accounting for 2.5% of Ethernet and 16% of T-1 bandwidth. Extrapolating from these numbers Ethernet could handle at most roughly 600 players at 70% utilization while each T-1 link could accommodate only 130. Note, again, that this extrapolation only gives us only upper bounds using the current DIS protocol.

We met our goals of minimizing latency and conforming to DIS 2.0.3. Comments from observers and players indicated that there were no noticeable delays in interacting with other players over the wide-area and local nets. Moreover, NPSNET-IV operated smoothly with AFIT's Virtual Cockpit which also uses the DIS standard. With the configuration used at SIGGRAPH, we could expect no more than several hundred simultaneous interactive players over the network. Overall, NPSNET-IV demonstrated robust network capabilities.

B. MBONE

After SIGGRAPH, we completed the multicast version of NPSNET-IV. After some initial tests over the Naval Postgraduate School campus network, we began experimentation over the MBONE with the help of Stephen Lau at SRI. Communicating between SRI and NPS presents a significant challenge for interactive simulation. SRI, which is on DARTNET, and NPS, attached to BARRNET, are separated by six routers with a variable delay of between 100 to 1000 ms. Figure 8 shows the routing over the MBONE. The annotations indicate physical or tunneled links and the metric associated with the links. For example, P1 indicates that a link is a direct multicast link with its metric set to 1. Moreover, DIS traffic must compete with MBONE video and audio multicasts. Figure 9 shows the PDU traffic generated across the MBONE between two SGI workstations running NPSNET-IV at NPS and SRI.

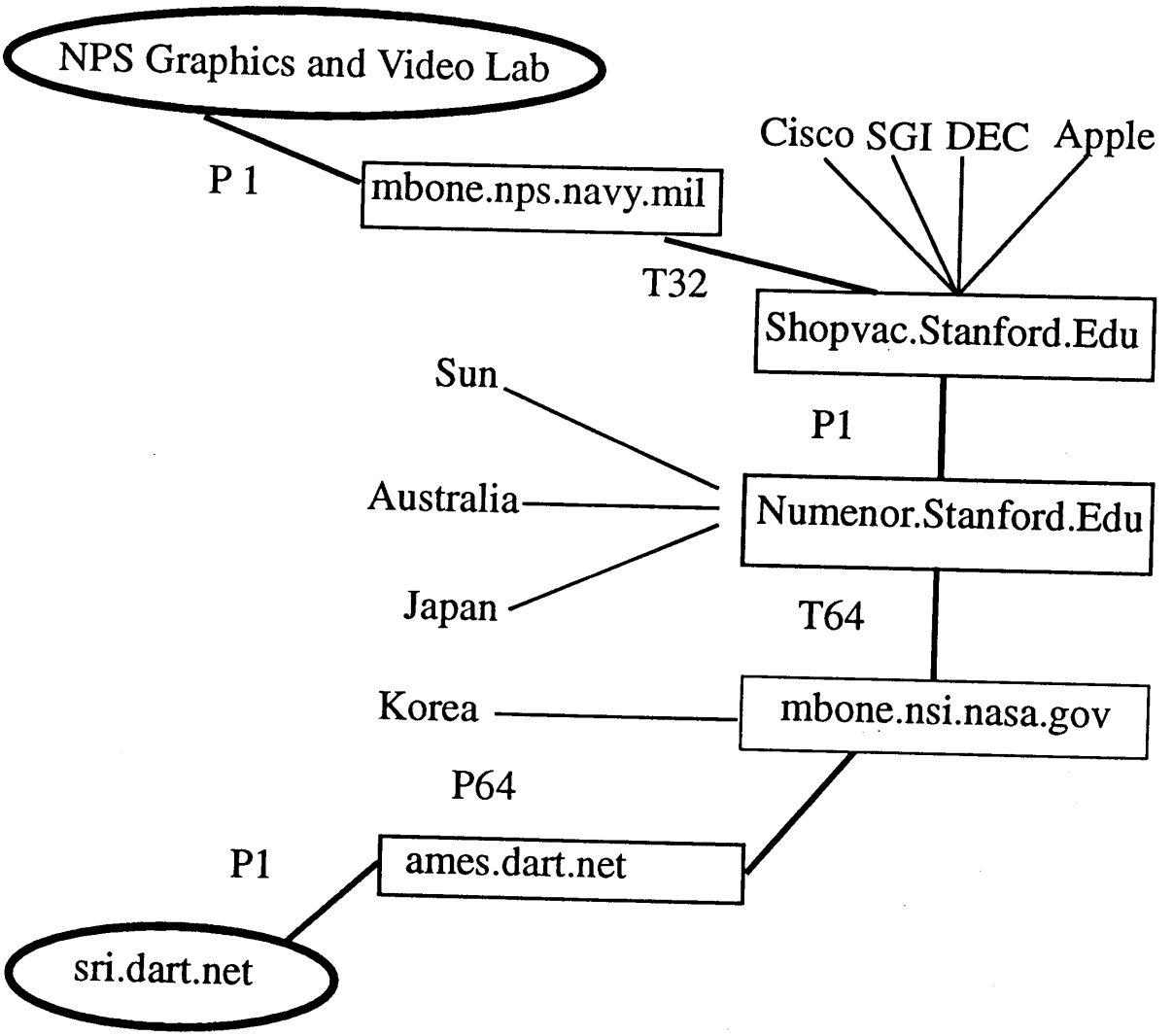

Figure 8. MBONE Routing between NPS and SRI

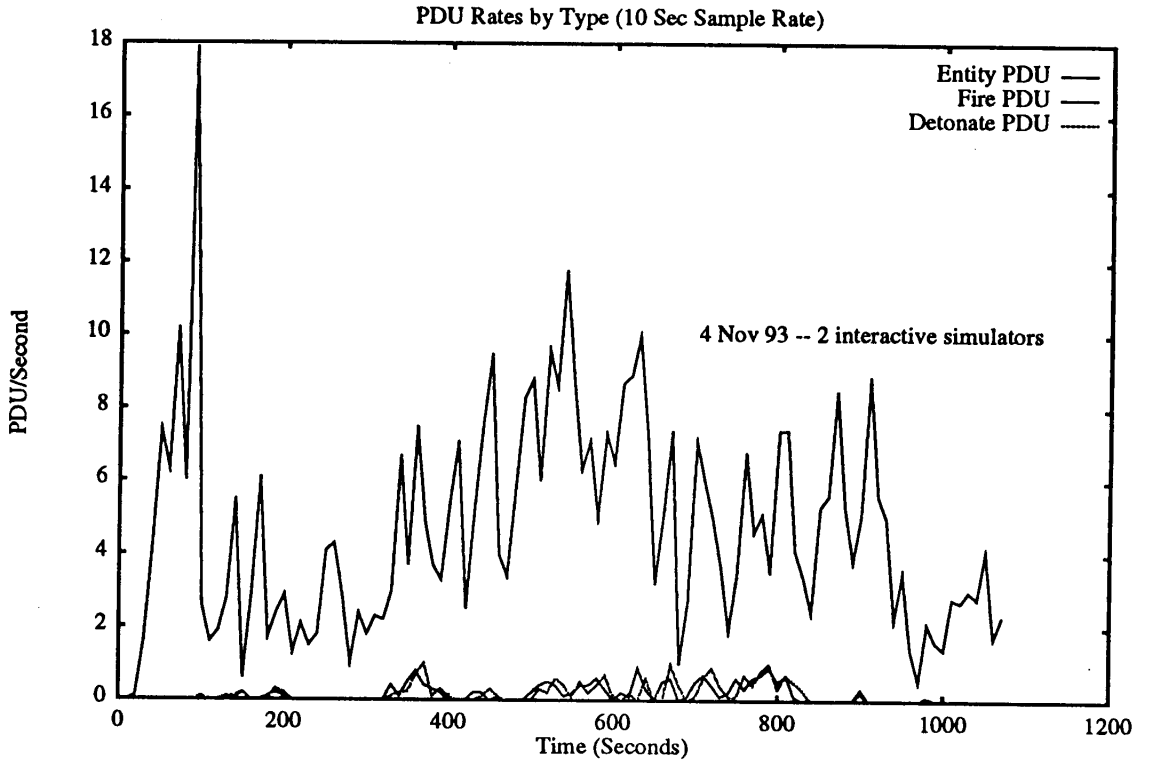

Figure 9. IP Multicast between SRI and NPS.

Despite a hostile network environment, NPSNET-IV showed little perceptual latency. We used Lau's version of a multicast video tool to transmit images of his simulation running on an SGI Onyx to the players at NPS. We could observe what was been viewed at SRI, including weapons fire on our aircraft.

We are now currently collecting additional network performance data and expanding the number of players over the MBONE. We are constrained at present by the limited number of sites which possess both access to MBONE and have the SGI platforms. Volunteers should contact the authors.

VIII. Conclusion

We see the importance of distributed virtual environments continuing to grow within the next several years. Government organizations like ARPA and the US Army have recognized this through programs such as the Combined Arms Tactical Trainer (CATT), a distributed 3D simulator for training tank crews and units, and the Louisiana Maneuver (LAM) exercises which will involve major elements of the Army in large simulated battles at the end of the decade to test new operational concepts [USA93].

Commercial interest is also keen. Companies such AT&T, TCI, and Nintendo are pursuing distributed VR for multi-player games. Compare Jaron Lanier's vision of networked VR with the description by Robert Kavner's, CEO for Multimedia Products and Services, AT&T, in a speech to the 1994 Winter Consumer Electronics Show:

The only dedicated gaming network in today's narrowband world is ImagiNation Network, in which we are a part-owner -- and, more importantly, with whom we are working to develop new services. It has advanced graphics and lots of interactive flexibility. As people use this communications-intensive service, they're seeing its potential and adapting it to their life-styles. We are often asked why are we working with this small, online network? We are working with ImagiNation Network to find ways that people can use the network to strengthen their sense of community.

We want to build software that allows such a capability and more. However, much remains to be done in the development of a network architecture to support large scale virtual environments. Further integrating networked hypermedia and improving the DIS protocols are both major challenges we intend to pursue. Additionally, we are exploring the use of ATM and local shared memory networks to meet future real-time performance requirements for large scale simulations.

This paper has presented these challenges and the work by the NPSNET Research Group to develop such an architecture. The emphasis of this paper has been on the networking issues of distributed environments with respect to NPSNET-IV. We believe that with the creation of the first 3D distributed virtual environment suitable for use on the Internet, we have shown the viability of using DIS, IP Multicast and heterogeneous parallel processing for developing large scale virtual environments virtual environments.

LIST OF REFERENCES

[Akeley89] Akeley, Kurt. *The Silicon Graphics 4D/240GTX Superworkstaion.* IEEE Computer Graphics and Applications. Vol. 9 No. 4, July 1989, pp. 71-83.

[Beren93] Berenbaum, Alan, Dixson, Joe, Iyengar Anand, and Keshav, Srinivasan, Keshav. *A Flexible ATM-Host Interface for XUNET II.* IEEE Network. Vol. 7, No. 4. July 1993. pp. 18-23.

[Brick93] Bricken, William and Coco, Geoffrey. *The Veos Project.* 1993. Available via anonymous ftp from ftp.u.washington.edu:/public/VirtualReality/HITL/papers/tech-reports/Veos_Project.ps.

[Casner93a] Casner, Steve. *Frequently Asked Questions on the Multicast Backbone.* 6 May 1993. Available at venrera.isi.edu:/mbone/faq.txt.

[Casner93b] Casner, Steve and Schulzrinne H. *RTP: A Transport Protocol for Real-Time Applications.* 20 October 1993. IETF Draft.

[Casner93c] Casner, Stephen and Deering, Stephen. *First IETF Internet Audiocast.* ACM SIGCOMM Computer Communication Review. July 1992. pp. 92 -97.

[Carls93] Carlsson, Christer and Hagsand, Olof. *DIVE -- A Platform for Multi-User Virtual Environments.* Computer & Graphics Vol. 17, No. 6, 1993. pp. 663-669.

[Cater94] Cater, John P. *Use of the Remote Access Virtual Environment (RAVEN) for Coordinated IVA-EVA Astronaut Training and Evaluation.* 1994. Submitted to Presence.

[Clark89] Clark, David, Jacobson, Van, Romkey, John, and Salwin, Howard. *Analysis of TCP Processing Overhead.* IEEE Communications. Vol. 27, No. 6. June 1989. pp. 23-29

[Chun92] Chung, James W., *An Assessment and Forecast of Commercial Enabling Technologies for Advanced Distributed Simulation,* Institute for Defense Analysis, Arlington, Virginia, October 1992.

[Cohen93] Cohen, Danny. *Joint scalability Environment.* Presented to the ARPA Scalability Working Group. 16 November 1993.

[Comer91] Comer, Douglas E. *Internetworking with TCP, Vol. I.* 1991. Prentice-Hall, New Jersey.

[Deer89] Deering, Stephen. *Host Extensions for IP Multicasting.* RFC 1112. August 1989.

[Deer93] Deering, Stephen. *MBONE-The Multicast Backbone*. CERFnet Seminar. 3 March 1993.

[Doris93] Doris, Ken. *Issues Related to Multicast Groups*. The Eighth Workshop on Standards for the Interoperability of Defense Simulations. March 1993. pp. 269-302.

[Frie88] Friedman, Dan, Haimo, Varda, *SIMNET Ethernet Performance*, BBN Communications Corporation, Cambridge, Massachusetts, January 1988.

[Harv92] Harvey, Edward P., Schaffer, Richard L., *The Capability of the Distributed Interactive Simulation Networking Standard to Support High Fidelity Aircraft Simulation*, BMH Associates, Inc. and BBN Systems and Technologies, Norfolk VA, Cambridge, Massachusetts, July 1992.

[Harris93] Harrison, Lynn T., Sawler, Robert J., and Bouwens, Christine. *Challenges to CGF Interoperability*. CAE-Link Technical Report. Binghampton, NY. 1993.

[IEEE93] Institute of Electrical and Electronics Engineers, International Standard, ANSI/IEEE Std 1278-1993, *Standard for Information Technology, Protocols for Distributed Interactive Simulation*, March 1993.

[IEEE85] Institute of Electrical and Electronics Engineers, International Standard, ANSI/IEEE Std 802.3-1988, *Information Processing Systems, Local Area Networks, Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA-CD) Access Method and Physical Layer Specifications*, First Edition, December 1989.

[ISIS92] *The Isis Distributed Toolkit Version 3.0 User Reference Manual. Isis Distributed Systems*. 1992. pp. 3-9.

[IST93] Institute for Simulation and Training, IST-TR-93-20, *Communication Architecture for Distributed Interactive Simulation (CADIS) [Final Draft]*, University of Central Florida, Orlando, Florida, June 1993.

[IST93a] Institute for Simulation & Training, IST-TR-93-11, *Distributed Interactive Simulation Operational Concept [Draft 2.2]*, University of Central Florida, Orlando, Florida, March 1993.

[IST93b] Institute for Simulation and Training, IST-TR-93-20, *Communication Architecture for Distributed Interactive Simulation (CADIS) [Final Draft]*, University of Central Florida, Orlando, Florida, June 1993.

[IST93c] Institute for Simulation and Training, IST-TR-93-20, *Guidance Document Communication Architecture for Distributed Interactive Simulation (CADIS) [Draft]*, University of Central Florida, Orlando, Florida, June 1993.

[IST93d] Institute for Simulation and Training, IST-TR-93-20, *Rationale Communication Architecture for Distributed Interactive Simulation (CADIS)*, University of Central Florida, Orlando, Florida, June 1993.

[IST93e] Institute for Simulation and Training, IST-CR-93-02, *Enumeration and Bit Encoded Values for Use with Protocols for Distributed Interactive Simulation Applications*, University of Central Florida, Orlando, Florida, March 1993.

[IST93f] Institute for Simulation and Training, IST-CR-93-15, *Standard for Information Technology, Protocols for Distributed Interactive Simulation Applications [Proposed IEEE Standard Draft]*, University of Central Florida, Orlando, Florida, June 1993.

[IST93g] Institute for Simulation and Training, IST-TR-93-08, *Simulator Networking Handbook*, University of Central Florida, Orlando, Florida, June 1993.

[Kalaws93] Kalawsky, Roy. *The Science of Virtual Reality and Virtual Environments*. Addison-Wesley. Workingham, England. 1993.

[Lanier94] Lanier, Jaron. *The Origin of VR*. IEEE Spectrum. Vol 31. No. 2. February 1994. pp. 6.

[Lock92] Locke, John, Pratt, David R., and Zyda, Michael J., *Integrating SIMNET with NPSNET Using a Mix of Silicon Graphics and Sun Workstations*, Naval Postgraduate School, Monterey, California, March 1992.

[Lora92] Loral Systems Company, *Strawman Distributed Interactive Simulation Architecture Description Document Volume 1*, Advanced Distributed Simulation Technology Program Office, Orlando, Florida, March 1992.

[Merc94] *Membership Surge Strains Access to American Online*. Mercury News. 2 February 1994. pp. 1F.

[Miller89] Miller, Duncan C.,Pope, Arthur C.; and Waters, Roland M. *Long-Haul Networking of Simulators*.Proceedings: Tenth Interservice/Industry Training Systems Conference; Orlando, Florida; December 1989. p. 2.

[Moy93] Moy, John. *Multicast Extensions to OSPF*. July 1993. IETF Draft.

[Ohya93] Ohya, Jun, Kitamura, Yasuichi, Takemura, Haruo, et. al. *Real-time Reproduction of 3D Human Images in Virtual Space Teleconferencing*. IEEE Virtual Reality International Symposium. September1993. pp. 408-414.

[Part94] Partridge, Craig. *Gigabit Networking*. Addison-Wesley. Reading, Massachusetts. 1994.

[Perl92] Perlman, Radia. *Interconnections: Bridges and Routers.* 1992. Addison-Wesley. New York. p. 258.

[Pope89] Pope, Arthur, BBN Report No. 7102, *The SIMNET Network and Protocols*, BBN Systems and Technologies, Cambridge, Massachusetts, July, 1989.

[Pratt93] Pratt, David R., *A Software Architecture for the Construction and Management of Real Time virtual environments*, Dissertation, Naval Postgraduate School, Monterey, California, June 1993.

[Shaw93] Shaw, Chris, and Green Mark. *The MR Toolkit Peers Package and Experiment.* IEEE Virtual Reality International Symposium. September1993. pp. 463-469.

[Singh94] Singh, Gurminder. *A Software Toolkit for Network-Based virtual environments.* Presence. Vol 3, No.1. 1994.

[SRI92] SRI International, *ATD-1 Architecture White Paper Edit Draft*, Menlo Park, CA, undated.

[USA93] *United States Army Posture Statement.* US Army. March 1993, pp. 65.

[VanH93] Van Hook, Daniel J. *Simulation Tool for Developing and Evaluating Networks and Algorithms in Support of STOW 94.* Presented for scalability Peer Review 19-20 August 1993.

[Wei92] Wei, Liming, et al. *Analysis of a Resequencer Model for Multicast over ATM Networks.* Proceedings of the 3rd International Workshop on Network OS Support for Digital Audio and Video, San Diego, Nov 1992.

[Wei93] Wei, Liming and Estrin, Deborah. *A Comparison of Multicast Trees and Algorithms.* Submitted to INFOCOM 94.

[Whet94] Whetten, Brian and Kaplan, Simon. *A High Performance Totally Ordered Multicast Protocol.* Presented to SIGCOMM'94.

[Zesw93] Zeswitz, Steven. *NPSNET: Integration of Distributed Interactive Simulation Protocol for Communication Architecture and Information Interchange. Master's Thesis.* Naval Postgraduate School. September 1993.

[Zyda94] Zyda, Michael J., Pratt, David R., John S. Falby, Chuck Lombardo, Kelleher, Kristen M. *The Software Required for the Computer Generation of Virtual Environments.* Presence. Volume 2, Number 2. Spring 1993. pp. 130-140.

[Zyda93] Zyda, Michael J., Pratt, David R., Kelleher, Kristen M. *1992 NPSNET Research Group Overview*, Naval Postgraduate School, Monterey, California, May 1993.

A special note of thanks to Jerry Prothero of HITL, who gave me a survey he wrote of distributed virtual environments.