# Interest Management in Large-Scale Distributed Simulations

Katherine L. Morse

Department of Information & Computer Science
University of California, Irvine
kmorse@ics.uci.edu

Science Applications International Corporation[1]
10770 Wateridge Circle
San Diego, CA 92121
katherine_morse@cpqm.saic.com

## ABSTRACT

Large-scale distributed simulations model the activities of thousands of entities interacting in a virtual environment simulated over wide area networks. Originally these systems used protocols which dictated that all entities broadcast messages about all activities, including remaining immobile or inactive, to all other entities, resulting in an explosion of incoming messages for all entities, most of which are of no interest. Using a filtering mechanism called Interest Management, some of these systems now allow entities to express interest in only the subset of information which is relevant to them. This paper surveys seven such systems, describing the purpose of the system, its scope, and the salient characteristics of it Interest Management scheme. I present the first taxonomy for such systems and classify the seven systems according to the taxonomy. The analysis of the classification points to potential areas of research in Interest Management.

## 1. INTRODUCTION

Large-scale distributed simulations (*LSDSs*) consist of thousands of complex entities moving and interacting in a large virtual environment. The physical environment is a collection of Local Area Networks (LANs) connected by a Wide Area Network (WAN), such as the Internet, spanning an area as large as a continent. Each LAN typically consists of several computers, each of which usually runs a single type of simulator. The simulator may simulate one or many entities. The total number of computers may number in the hundreds. Through this type of simulation, users and computers at geographically dispersed locations can interact with each other in a shared, simulated environment just as if they were in the same physical location. In addition, because the environment is simulated, activities can be performed which would be dangerous, expensive, or physically impossible in a real environment.

The remainder of this section describes applications of LSDS, the scope of such applications, and provides a brief history. Section 2 describes the basic mechanism

[1] This work was partially funded by U.S. Army Contract DAHH01-95-C-R109.

of Interest Management including the interest expressions which entities use to invoke it. Section 3 presents a taxonomy for classifying Interest Management systems. Section 4 describes in detail seven systems which have used Interest Management. Section 5 classifies the seven systems surveyed according to the taxonomy described in section 3. Section 6 presents conclusions about the current state and future of Interest Management.

### 1.1 Applications of LSDS

The most common application of this technology to date has been to military war game training exercises. The technology has expanded directly from this origin into the entertainment field in the form of virtual reality games. It also has potential applications to artificial life, molecular dynamics, collision of space debris, and environmental simulation. This application domain introduces two problems. The first is that there is usually no correlation between an entity's physical location in the network, i.e. where it's simulated, and its logical location in the virtual environment. And, of course, most entities are moving, so any grouping relationships that may be found at one point in time won't necessarily hold for any length of time. Secondly, the fact that humans are interacting within the virtual environment introduces the requirement that updates from entities in perceptible range must be reflected at intervals of approximately 100 ms, 80 ms of which is typically consumed by network transmission [6]. The communication model used exacerbates these problems.

The remainder of the paper focuses on war game simulations as they are the most prevalent application of LSDS to date. Figure 1, Virtual Environment, illustrates the potential juxtaposition of entity location in the network environment and entity location in the virtual network. Notice that some entities simulated on the same node or LAN are far apart in the virtual environment while entities simulated far apart on the WAN may be very close in the virtual environment. The entities are shown overlaid on a grid because discretizing the virtual environment into grid cells is a common technique in Interest Management.

The LANs may be ATM, Ethernet, FDDI, ScramNet or SCI, the latter for connecting via gateways to legacy systems which cannot be upgraded to interoperate with new systems and protocols. Such a system is shown in the upper left ring network. The WAN may be IP, ATM, or IP over ATM. Obviously the Internet could serve this purpose. The Defense Simulation Internet is a dedicated DoD internet which has been used for several LSDSs [8].

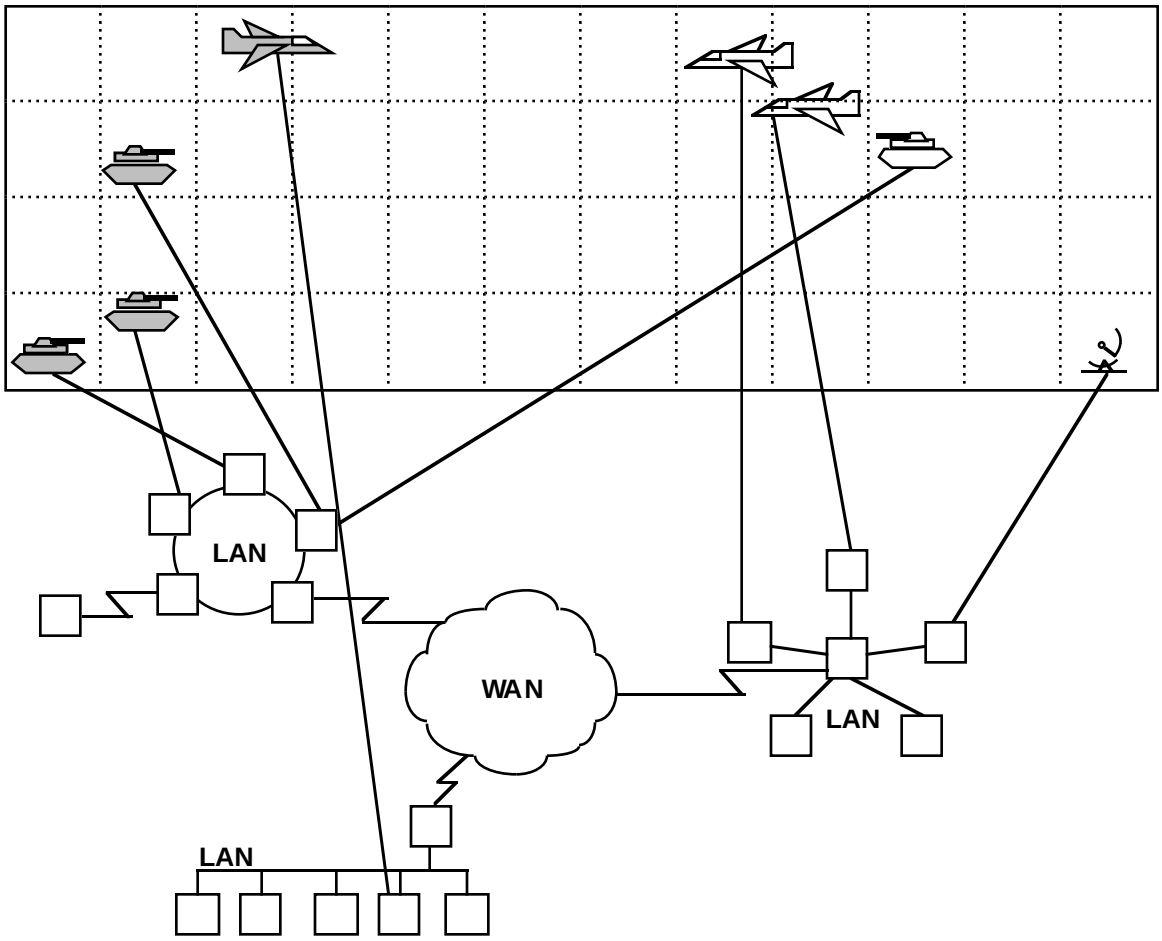


**Figure 1. Virtual Environment**

### 1.2 Scope

The network for an LSDS may encompass dozens of LANs supporting hundreds of host nodes. These nodes may support thousands of simulated entities. DARPA envisions simulations encompassing 100,000 entities [8]. Simulated entities are grouped into three categories. Live entities are actual, physical objects which have communications support and instrumentation which allows them to interact with other entities in the simulation. Virtual entities are man-in-the-loop simulators such as tank or flight simulators in which the individual operator performs operations just as if the vehicle were in the field. Constructive entities are almost entirely synthetic. A single workstation controlled by an individual playing the role of a commander directs the operations of many simulated entities [8]. In the virtual environment, all three types of entities appear equally real to all participants. This feature supports the two primary missions of these types of simulation: training personnel on existing systems and testing the effectiveness of planned systems not yet in production.

### 1.3 History

SIMNET (1983 - 1992) was the precursor of most of the LSDSs surveyed in this paper. Based on the lessons learned with SIMNET, DARPA developed the *Distributed Interactive Simulation* (*DIS*) protocol which has subsequently become IEEE standard 1278 [7]. The DIS protocol specifies the type and format of the *protocol data units* (*PDUs*) to be passed between entities, and responses to be given upon receipt of certain PDUs. Version 1.0 of the standard specified seven types of PDUs. The current version, 2.0.4 (IEEE 1278.1) specifies 27. Version 3 is planned to include terrain description environmental effects. Version 3 will also be subsumed by the *High Level Architecture* (*HLA*), an initiative targeted at unifying almost all existing military simulations [8].

## 2. INTEREST MANAGEMENT

The DIS protocol is stateless and assumes unreliable communication. Entities are required to periodically broadcast "heartbeat" *entity state PDUs* (*ESPDUs*) to accommodate entities joining the simulation late and to compensate for PDUs lost due to communication errors. Using this protocol for a simulation of 100,000 entities, each node in the network would require a 375 Mbps network connection [1]. ESPDUs may account for 62 - 97% [13, 14] of the data traffic. In some experiments, as much as 90% of the data is useless to the receiving entity [16]. The receiving entity is responsible for sorting through and discarding the useless messages. This process cripples the performance of the simulations and restricts their scalability. The concept of Interest Management[2] was developed to address this problem by reducing the arriving messages to a smaller, relevant set. Under Interest Management, an entity expresses its data interests in terms of location and other application-specific attributes. Other agents in the simulation infrastructure, *interest managers* (*IMs*), accept entities' *interest expressions* (*IEs*) and use them to filter messages to sets (or reduced supersets) which meets the entities' needs.

### 2.1 Interest Expressions

An IE is a specification of the data one simulation or entity needs to receive from other simulations or entities in order to interact with them correctly. IEs may refer to several attributes of the simulation and/or entities. They may be geographic. They may refer to some attribute of an entity including its type. A tank may want to know about all ground vehicles within a four kilometer radius around itself, but a soldier only cares about entities within a 400 meter radius. An airborne surveillance radar may express interest in all ships within a radius of 30 nautical miles. IEs may also specify a reduced resolution or frequency of data. A wide-area viewer may need data about all entities in the simulation, but not every time new data is generated. Updates every few minutes may be sufficient. For a small virtual environment on a few hundred square kilometers the problem may not seem insurmountable, but at least one virtual environment already implemented was 15,000 square kilometers [10].

[2] Interest Management is also referred to as relevance filtering and data subscription.

### 2.2 Domain of Interest & Domain of Responsibility

The terms *Area of Interest* and *Region of Interest* have been used interchangeably to refer to both the data of interest to an entity and to the domain of parameter space for which an IM is responsible. To distinguish the different perspectives of entities and IMs, I propose the following clarifying terminology. The data of interest to an entity is its *Domain of Interest* (*DOI*). The domain of parameter space for which an IM is responsible is its *Domain of Responsibility* (*DOR*). This terminology also highlights the fact that the data of interest in both cases is a domain in the multi-dimensional parameter space and not just a geographic area or region, although the geographic interpretation is the most prevalent.

### 2.3 Observations on Interest Management

From the preceding we can make some general observations about Interest Management:

- IEs are often self-referential, e.g. an entity wanting to know about all other entities within a specified radius of itself.
- Notice that such a self-referential statement of the IE is not likely to change over the life of the entity, but its evaluation will. The entity itself may move, changing the center of its DOI, and other entities may move, putting themselves into the DOI.
- The "size" of the DOI may have no relationship to the number of messages the IE does or does not filter. Particularly in war games, entities tend to cluster as the simulation progresses. So, an IE which netted only a few messages early in the simulation may collect significantly more as entities move into closer proximity.
- Interest is not necessarily symmetric. Entities may have different ranges. A wide area viewer is a good, and problematic, example. It needs to know about all other entities, but all other entities don't need to know about it. In addition, range may be directional with the effect that two entities would be able to "see" each other except that one is "behind" the other [2].
- All of the preceding observations assume visual sensing of entities in geographic space. While this is the most intuitive and most prevalent us of Interest Management, the same techniques and observations apply to other types of sensing in other dimensions.

## 3. TAXONOMY

Having described the salient characteristics of all Interest Management schemes, we can now develop a framework for classifying individual implementations. I propose the following three-category taxonomy for classifying Interest Management schemes.

### 3.1 Communication Model

The three types of communication model are *unicast*, *broadcast*, and *multicast*. While the original communication model for LSDSs was broadcast, Interest Management requires more precise direction of message transmissions which clearly may be supported with unicast. Multicast is attractive because it can reduce

the network load. However, the applicability of multicast is reduced by the restriction on the number of groups that current multicast hardware can support and the time it takes to reconfigure the groups. The latter can be on the same order of time as that allocated for sending a message, including formulating it, passing it through the local node's call stack, network latency, retrieving the message, and unpacking it at the receiving node [10].

### 3.2 Filtering Focus

Filtering focus differentiates systems that filter data based on *intrinsic* characteristics of objects, such as the values of their attributes, from those that filter data based on *extrinsic* characteristics of objects, such as their location in a cell of a grid logically overlaid on the virtual environment[3]. Table 1, Filtering Focus, provides a comparison of the characteristics of intrinsic and extrinsic filtering.

**Table 1. Filtering Focus**

| Intrinsic | Extrinsic |
|---|---|
| Filter on attribute (name) | Filter on grid cell, PDU type, or multicast group |
| Requires looking "into" message and possibly performing value comparisons other than "=" | Doesn't require looking "into" message except fro message type (analogous to net routing on destination); doesn't require knowledge of data structure semantics |
| Fine-grained | Coarse-grained |
| One-to-one | Many-to-many |

The differentiation between intrinsic and extrinsic may also be characterized as individual-based vs. environment-based in the context of war game simulation. Van Hook, et al, take a more application-centric view and refer to this categorization as grid-based and object-based [24]. In tests prior to ED-1, but not part of an actual exercise, they demonstrated that object-based filtering achieves greater reduction in the number of packets sent using fewer multicast groups. No measure of processing time was calculated. Mellon refers to this categorization as type-based and value-based [11].

Filtering focus could also encompass frequency (or precision) which is orthogonal to the distinction between intrinsic vs. extrinsic. Regardless of how the data is filtered, an entity may not require updated information from other entities as frequently as it is available, resulting in a less precise view of the state of the virtual environment.

[3]Intrinsic is defined as "belonging to the essential nature or constitution of a thing [26]." Extrinsic is defined as "from without; not forming part of or belonging to a thing ."

### 3.3 IM Domain of Responsiblity (DOR)

An IM's DOR is the area in multi-dimensional parameter space in which the IM is responsible for managing data transmission. If IMs are assigned to predefined grid cells, entities may move in and out of the IMs' DORs, but the DORs themselves don't change.

It may be the case that IMs are not instantiated objects in the system, but logical constructs. In one of the systems surveyed, IEs operate somewhat autonomously on behalf of individual entities. Here the IM is associated with an individual entity and is the collective, distributed action of the entity's IEs at remote nodes. In essence the entity's IEs are the IM. From the perspective of the IM, its DOR is the same as the entity's DOI. The DOR of such a logical IM will change as the entity moves, the entity's IEs change, or other entities move in and out of the DOI.

These two examples illustrate the difference between *static* and *dynamic* DORs. A dynamic DOR may change size and shape as well as location. This change may be effected by adding cells, forming multicast groups, or changing cell sizes.

## 4. IMPLEMENTATIONS

For the purposes of this survey, I differentiate between LSDSs which use DIS PDUs and those which are DIS compliant. Since the broadcast requirement of the DIS protocol causes the greatest degree of overload, several of the systems surveyed use only the DIS PDU formats, but are not compliant with the broadcast portion of the DIS protocol. Specifically, Interest Management techniques are used to reduce the amount of data sent, particularly heartbeat ESPDUs. In some cases these systems maintain some form of database of the current state of the exercise so that simulators which join late can request and quickly receive all necessary state information.

### 4.1 ModSAF

Modular Semi-Automated Forces (ModSAF) [18, 17, 25] is a constructive simulator designed for the Army and fielded in the 1990s. It began as a simple tool to generate targets for initial operator training in tank simulators. The original implementation was in Lisp on a BBN Butterfly. It has since evolved through programs called SIMNET SAF, OdinSAF, and finally ModSAF. ModSAF is DIS compliant. It can scale to 850 entities running on 17 platforms, including SGI, Sun, MIPS, HP, DEC Alpha, IBM, and PowerPC. It was designed to be portable and to support many other simulators, so no specific exercises were performed. It was used in STOW ED-1 and NPSNET (both described below).

ModSAF uses the most basic extrinsic filtering scheme of grid cells which filter PDUs by their type. The grid cells are assigned a priori and their DORs remain static throughout the exercise. IEs are hard-wired into code modules. The modules have code which registers them with a packet filtering mechanism to receive specific types of PDUs from entities in the grid cell of interest. The entities themselves do

not express interest. In fact, entities are passive in terms of both movement and interest expression. Since a single simulator simulates many entities, movement and Interest Management are managed at an aggregate level rather than by individual entities. Most versions, including the current one, 2.0, use broadcast. Version 1.4 experimented with multicast, using high- and low-fidelity grids. Every other PDU is sent to alternating grids. An entity wanting low-fidelity information subscribes to just the low-fidelity grid. An entity wanting high-fidelity information subscribes to both. ModSAF can be characterized as having destination-based filtering. All PDUs arrive at all nodes, but are filtered by a code module before being passed to individual entities.

### 4.2 JPSD

Joint Precision Strike Demo (JPSD) [12, 26] is a prototype simulator built between 1993 and 1996 to train Army tactical commanders. JPSD uses DIS PDUs, but is not DIS compliant. It was designed to support 8,300 constructive entities. A recent demonstration supported 5,000 to 6,000 entities for several hours running on 70 to 80 nodes. Forty of these nodes were located at a main site and connected to an ATM star configured with multicast groups to emulate Ethernet LANs. Six gateways connected the remaining nodes at several other sites, some via ATM.

IEs in JPSD are predicates in conjunctive normal form about attributes of entities. The terms can be is-equal, is-in-range, or function calls which return true or false. The IEs are compiled, but their variables, start time, and end time are bound at run time. Although not yet implemented, there are plans for an entity-centric Modeling Interest Language. Each node has a single IM for all the entities simulated on the node. Entities broadcast (or publish) their IEs to IMs on all other nodes. When a PDU is generated on a node, the local IM checks the contents of the PDU against all received IEs, i.e. it filters the PDUs intrinsically. A pass actor is created for each PDU which satisfies at least one IE. The pass actor collects the IDs of all remote entities whose IE the PDU passes and sends the PDU only after all IDs are collected. Since the DOR of each local IM is the union of the locations of all local entities in the multi-dimensional parameter space, the DOR changes dynamically as entities change their locations.

Two versions of interest publication were implemented: static and dynamic. Static publication performs a priori analysis of PDU types that simulations need and sets up channels allocated at simulation startup for each type. Dynamic publication allows entities to publish IEs when they need the data. Obviously dynamic publication is a more general approach. Although JPSD currently uses point-to-point unicast, the pass actor mechanism is designed to be integrated with multicast.

### 4.3 CCTT

Close Combat Tactical Trainer (CCTT) [10] is an incremental software and system integration of SAFs and manned simulators to train Army tank and mechanized infantry forces. Begun in 1993 and continuing through 1996, it is the first fully DIS compliant system built. It can support 851 entities: 50 manned modules and 15

computer generated forces processors each able to simulate 60 vehicles. The manned simulators are connected via an FDDI LAN using TCP/IP and UDP/IP, with plans to migrate to a WAN architecture.

CCTT's Interest Management scheme uses a combination of grid-based and PDU type-based IEs. Entities can express interest in a collection of predefined square grid cells which are 5 kilometers on a side, and in a prespecified subset of PDU types such as ESPDUs, Fire PDUS, etc. PDUs are filtered extrinsically, first on the basis of the originating cell and then on PDU type. Multicast groups are preassigned to grids and PDU types, and cannot be changed at run time. The local IM on each node (referred to as the network manager) joins and leaves multicast groups to meet the IEs of local entities. If the number of multicast groups a network manager needs to join exceeds the hardware limit of 64, the network manager must take up the slack by doing the filtering in software.

CCTT is the first system surveyed to demonstrate a decoupling of the instantiated IM from any DOR. The IM on the local node is managing the IEs of the local entities whose DOIs, as we observed earlier, must change by defintion. The IM itself has no DOR. The DORs are assigned to multicast groups. We see some level of dynamism with the IM joining and leaving multicast groups, but the system is still classified as having static DOR since the domain of the individual multicast groups never changes. We will see that STOW ED-1 has this same characteristic.

### 4.4 NPSNET

Naval Postgraduate School NET (NPSNET) [9] is a DIS 2.0.3-compliant 3D visual simulator testbed developed by the students and researchers at NPS. ModSAF is integrated with NPSNET, and 10 exercises have been conducted supporting up to 7 sites with up to 198 entities. The network architecture consists of SGIs connected via FDDI at the LAN level and using MBONE at the WAN level. Work is currently underway to move to Realtime Information Transfer and Networking (RITN).

NPSNET uses hexagonal grid cells which more closely approximate a real world DOI which would be round. An entity's DOI consists of a radius of grid cells where the entity is joining new cells at the leading edge and leaving old cells at the trailing edge as it moves forward. An entity is an "active" member of the cell in which it is resident and a "passive" member of all the other cells in its DOI. Since all PDUs are sent to and received from the grid cells without further filtering, the filtering focus is extrinsic. Grid cells are pre-assigned to multicast groups in a straightforward, static manner. The mechanism for maintaining membership in the groups is distributed by making the oldest active member of the grid cell the group leader. The group leader is responsible for adding members, deleting members, and providing new members with an aggregate of all current ESPDUs for the cell. In addition, each node has a local IM. Macedonia, et al envision IMs based on spatial (geographic), functional (type), and temporal (frequency of updates) relationships. To date they have only implemented the spatial relationship with this grid cell approach.

### 4.5 STOW-E

Synthetic Theater of War-Europe (STOW-E) [22, 23, 21, 14] is one in a series of STOW programs lead by DARPA with participation by all the services. The STOW programs began in 1993 and are slated to run until at least 2000. These programs combine live, constructive, and virtual simulations to support readiness and operations, requirements determination, acquisition, and test and evaluation. Later STOWs are logical, but not necessarily physical successors of earlier STOWs, i.e. lessons learned are used, but not necessarily code. The STOW-E exercise was played in November 1994 on a European virtual battlefield using DIS 2.0.3 with experimental PDU additions. Three thousand live, constructive, and virtual entities participated at 24 sites, including 18 on the Defense Simulation Internet with the rest bridged legacy systems. STOW-E's filtering capability was provided by Realtime Information Transfer and Networking.

STOW-E uses extrinsic, grid-based filtering of ESPDUs. The DOI of an entity is referred to as its cell set. Each LAN has an Application Gateway at the LAN-WAN boundary which supports the IM. It collects the cell sets of all entities on its LAN, unions them, and broadcasts the union to all remote Application Gateways. The remote Application Gateways union (cluster) all the received cell sets into a full accuracy region. This is the data of immediate interest to remote entities. All data generated on a LAN which is within its full accuracy region is broadcast by the Application Gateway to all other Application Gateways. Data generated on the local LAN, but outside the full accuracy region (within the reduced accuracy region) is broadcast with reduced frequency, e.g. only 1 in every n PDUs generated in the reduced accuracy region is broadcast. The reason for broadcasting the data at all is that entities may join and move, and their IEs may not take affect in time for them to get data that they may require immediately. Since an Application Gateway aggregates all local cell sets before sending them out to other Application Gateways, the data returned will not be of interest to all entities on all nodes. So, filtering must be done by both the source and destination Application Gateways using the full accuracy region and cell sets, respectively. Notice that an Application Gateway's full accuracy region changes as both local and remote entities move across cells. Since the Application Gateway supports an instantiated IM coupled with a DOR which changes, this is the first system surveyed with an instantiated IM with a dynamic DOR.

### 4.6 STOW ED-1

Synthetic Theater of War Engineering Demo 1 (STOW ED-1) [21, 5, 4], October 1995, was the first in a series of engineering demonstrations with increasing levels of functionality planned for the next STOW system, scheduled to complete in 1997. STOW '97 is designed to improve joint services training. This system is planned to eventually support 10,000 entities at 50 sites. It uses DIS PDUs, but is not DIS-compliant. It is also scheduled to be the first system to use the High Level Architecture. ED-1 encompassed 7 sites, 62 ModSAFs, and 3,000 to 5,000 entities. ED-1's filtering capability was provided by an upgraded version of Realtime Information Transfer and Networking.

ED-1 uses an extrinsic, two-level, grid-based filtering scheme with low- and high-frequency grids (referred to as multiple fidelity/uncertainty channels) supported by multicast. This is similar to version 1.4 of ModSAF in which partial data is received by subscribing to one grid and complete data is received by subscribing to both. Several logical multicast groups were multiplexed onto a single physical group when the number of logical groups exceeded the hardware limit of approximately 1000. These groups were static. By 1997 hardware support for approximately 5000 dynamic groups may be available.

Each node in ED-1 supports a local IM (Subscription Principal) responsible for aggregating IEs for entities on the node and forwarding them to a Subscription Agent. An Agent Host at the LAN-WAN boundary (similar to the Application Gateway in STOW-E) supports a Subscription Agent whose function is similar to the Subscription Principal's, but at the level of the LAN. The Subscription Agent collects the IEs which have been aggregated at the Subscription Principals on the nodes on the LAN. The Subscription Agent evaluates all the IEs and joins the multicast groups which coincide with the IEs. Likewise, when an entity's behavior necessitates deleting an IE, this information is passed to the Subscription Principal which forwards it to the Subscription Agent. The Subscription Agent leaves the associated multicast group if no other entity on the LAN has an outstanding IE which still requires membership in the group. Data received at the Agent Host via the multicast groups is broadcast back to the LAN where the Subscription Principals retrieve just the data coinciding with the IEs of local entities. Like CCTT, STOW ED-1 demonstrates some dynamism by joining and leaving multicast groups, but is classified as having static DOR since the multicast groups are assigned to grids whose location and scope are fixed.

### 4.7 Proximity Detection

Unlike the other systems surveyed, this system had its genesis in parallel, optimistic simulation. It began as a non-DIS, constructive, ground war simulation executed on a Hypercube running the Time Warp Operating System [19, 20]. The same technique was applied to an aircraft simulation of 1,000 to 2,000 entities on a cluster of SGIs running SPEEDES. Both implementations used optimistic scheduling protocols and assumed reliable message passing. This system is included because it demonstrates the fundamental characteristics of Interest Management.

Proximity Detection uses an extrinsic, two-level, hierarchical filtering scheme where the coarse level is provided by active grid entities and the fine level is provided by the entities (sensors) themselves. As entities move they check in and out of the grid cells they occupy. The grids maintain lists of resident entities and manage the process of keeping all resident entities cognizant of each other. Since the grids are the IMs and their location and scope are fixed, their DORs are static. Entities use point-to-point unicast to send all their messages to all other entities in the current grid. The receiving entity performs more detailed filtering based on its own sensor model to determine if it can truly "sense" the other entity.

## 5. CLASSIFICATION

Table 2, Implementation Classification, presents the classification of the seven systems surveyed according to the preceding taxonomy.

**Table 2. Implementation Classification**

| System | Communication Model | Filtering Focus | IM DOR |
|---|---|---|---|
| ModSAF | Broadcast/ Multicast | Extrinsic | Static |
| JPSD | Unicast | Intrinsic | Dynamic |
| CCTT | Multicast | Extrinsic | Static |
| NPSNET | Multicast | Extrinsic | Static |
| STOW-E | Broadcast | Extrinsic | Dynamic |
| STOW ED-1 | Multicast | Extrinsic | Static |
| Proximity Detection | Unicast | Extrinsic | Static |

Based on the preceding survey and classification we can make some observations about the nature and current state of Interest Management.

- Intrinsic filtering inherently requires the IM's DOR to be dynamic. Since entities are filtering directly on the intrinsic characteristics of other entities, as the interested entity moves or changes attributes, its DOI implicitly changes.
- To date, multicast groups have been used to support extrinsic filtering. This is primarily driven by the time required to reconfigure multicast groups. Since intrinsic filtering requires dynamic DORs, the multicast groups would have to be reconfigured on the fly.
- Extrinsic focus solves the problem of interest management to a lesser degree than intrinsic filtering. Since extrinsic filtering results in a coarser filter, entities still receive a larger set of messages than they actually need. However, even intrinsic filtering is not necessarily perfect. The final decision about what data is relevant to a simulation can only be made based on detailed semantic information only available within the simulation itself. This issue touches on the debate over source-based vs. destination-based filtering[4]. Clearly, filtering data closer to its source reduces network traffic. But, the cost of filtering must be paid somewhere and sending very detailed filters over the network just shifts that cost and may offset the gains achieved by source-based filtering.
- None of the systems implemented thus far have an intrinsic/dynamic implementation with the concept of the DOR as an actual object. In JPSD the DORs can be viewed as the union of the IEs of the constituent entities, but

[4] I don't see the distinction between source-based and destination-based filtering as binary. The use of multicast groups is difficult to classify in one category or the other. And systems such as STOW ED-1 filter at more than one level in the architecture.

this grouping is implicit rather than explicit. RITN for STOW-E has the most dynamic DOR concept with clustering, but its filtering focus is extrinsic.

- Only Proximity Detection implements an extrinsic IM as an actual object in the system.

## 6. CONCLUSIONS

As the scope of these systems grow, Interest Management techniques will have to deliver ever more precise results in response to entities' IEs. Even extrinsic filtering using multicast groups may not be able to keep up, given the current and projected limitations of multicast switch hardware. It is widely held in the Interest Management community that the solution to the problem is truly dynamic Interest Management.

My research focuses on developing such Interest Management systems using autonomous objects. Autonomous objects execute in a distributed environment and differ from traditional remote procedure call and object paradigms in that autonomous objects are mobile within the network and carry their own behavior with them in the form of a program. This enables them to navigate freely in the underlying network, communicate with one another, and invoke node-resident functions. They can coordinate the invocation of and the exchange of data among the various functions distributed throughout the network in both time and space. They can also coordinate their behaviors among themselves.

I am investigating several approaches to Interest Management using the autonomous objects system MESSENGERS [3]. MESSENGERS has applicability to both intrinsic and extrinsic filtering. In an intrinsic filtering approach, IEs may be instantiated as Messengers which migrate to remote nodes where they determine the relevance of locally generated PDUs to their originator. This approach is somewhat similar to the one used in JPSD. However, because MESSENGERS supports a significant subset of C, the IEs can be arbitrarily complex. This functionality can be added to existing simulations with only slight modification to redirect output PDUs, as opposed to rewriting significant portions of the simulations to incorporate the functionality directly in the simulations. As an added benefit of this decoupling, at any time during an exercise an entity or simulation can inject a new IE into the system. The new IE can be arbitrarily different than any previous ones injected by the entity. This can have particular benefit in exercises which typically run non-stop for several days. If a player in the exercise sees that IEs defined prior to the exercise are not as effective as hoped, a new IE can be constructed and injected in real time. In systems in which the IEs are compiled into the simulators, it's not feasible in the middle of a exercise to take a simulation off line, change the IEs, recompile, and bring the simulation back on line.

MESSENGERS may be applied to extrinsic filtering in a manner which is analogous to load balancing the filtering responsibility for groups of entities. Initially an IM implemented as a Messenger can be assigned responsibility for a domain in the parameter space. As more entities move into that space, the IM may become

overloaded and need to divide its DOR with another IM. Using Messengers' capability to autonomously replcate themselves and migrate the copy to another node, the initial IM may create another IM to accept the overload and migrate the new IM to a less loaded node. "Adjacent" IMs may also coalesce when they become under loaded, thereby reducing unnecessary overhead of maintaining extraneous IMs. Clearly, this scheme requires testing and analysis to optimize scope, location, division, aggregation, and migration algorithms of IMs.

Finally, there is work in progress in the LSDS community to migrate simple objects, such as missiles, closer to entities with which they are interacting to improve performance. As autonomous objects, the Messenger IMs may migrate through the network to be in closer proximity to the entities they are managing. The challenge of this approach is to decide in real time if the cost of migrating is mitigated by improved proximity. Making such a decision is complicated by the fact that it will be the case very infrequently that all the managed entities are conveniently collocated in the same region of the network.

## REFERENCES

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

[2] Steve Benford, Lennart E. Fahlen, and John Bowers. Supporting Social Communication Skills in Multi-Actor Artificial Realities. In *Fourth International Conference on Aritficial Reality and Tele-Existence*, pages 205-226, July 1994.

[3] Lubomir Bic, Munehiro Fukuda, and Michael Dillencourt. Distributed Computing using Autonomous Objects. *IEEE Computer*, 29(8), August 1996.

[4] James O. Calvin, Carol J. Chiang, and Daniel J. Van Hook. Data Subscription. In *12th Workshop on Standards for the Interoperability of Distributed Simulations*, pages 807-813. March 1995.

[5] James O. Calvin, David P. Cebula, Carol J. Chiang, Steven J. Rak, and Daniel J. Van Hook. Data Subscription in Support of Multicast Group Allocation. In *13th Workshop on Standards for the Interoperability of Distributed Simulations*, pages 367-369. September 1995.

[6] James O. Calvin, Duncan C. Miller, Johsua Seeger, Gregory Troxel, and Daniel J. Van Hook. Application Control Techniques System Architecture. Technical Report RITN-1001-00, MIT - Lincoln Labs, February 1995.

[7] DIS Steering Committee. Standard for Information Technology - Protocols for Distributed Interactive Simulation, March 1994. IEEE Standard 1278.

[8] DIS Steering Committee. The DIS Vision: A Map to the Future of Distributed Simulation. Version 1, May 1994.

[9] Michael Macedonia, Michael Zyda, David Pratt, and Paul Barham. Exploiting Reality with Multicast Groups: a Newtork Architecture for Large Scale Virtual Environments. In *Virtual Reality Annual International Symposium '95*, pages 2-10, March 1995.

[10] Thomas W. Mastaglio and Robert Callahan. A Large-Scale Complex Virtual Environment for Team Training. *IEEE Computer*, 28(7), pages 49-56, July 1995.

[11] Larry Mellon. STOW Interest Management. Presentation to the HLA Filter Tiger Team, January 1996.

[12] Edward T. Powell, Larry Mellon, James F. Watson, and Glenn H. Tarbox. Joint Precision Strike Demonstration (JPSD) Simulations Architecture. In *14th Workshop on Standards for the Interoperability of Distributed Simulations*, pages 807-810, March 1996.

[13] RDT&E Division. Engineering Demonstration #1 Bandwidth Analysis. Naval Command, Control and Ocean Surveillance Center, February 1996.

[14] RDT&E Division and Advanced Telecommunications Inc. Synthetic Theater of War-Europe Technical Analysis. Naval Command, Control and Ocean Surveillance Center, May 1995.

[15] Dennis Rogers. STOW-E Lessons Learned: Focused on the 3 Primary Army STOW-E sites. In *12th Workshop on Standards for the Interoperability of Distributed Simulations*, pages 143-147, March 1995.

[16] Steven J. Rak and Daniel J. Van Hook. Evaluation of Grid-Based Relevance Filtering for Multicast Group Assignment. In *14th Workshop on Standards for the Interoperability of Distributed Simulations*, pages 739-747, March 1996.

[17] Kevin L. Russo, Lawrence C. Shuette, Joshua E. Smith, and Matthew E. McGuire. Effectiveness of Various New Bandwidth Reduction Techniques in ModSAF. In *13th Workshop on Standards for the Interoperability of Distributed Simulations*, pages 587-591, September 1995.

[18] Joshua Smith, Kevin L. Russo, and Lawrence C. Schuette. Prototype Multicast IP Implementation in ModSAF. In *12th Workshop on Standards for the Interoperability of Distributed Simulations*, pages 175-178, March 1995.

[19] Jeff S. Steinman. Proximity Detection Interest Management. Personal Communication, February 1996.

[20] Jeff S. Steinman and Frederick Wieland. Parallel Proximity Detection and the Distribution List Algorithm. In *Proceedings of the 1994 Workshop on Parallel and Distributed Simulation*, pages 3-11, July 1994.

[21] Daniel J. Van Hook. RITN IM and IM History. Personal Communication, January 1996.

[22] Daniel J. Van Hook and James O. Calvin. AGENTS: An Architectural Construct to Support Distributed Simulation. In *11th Workshop on Standards for the Interoperability of Distributed Simulations*, pages 357-362, September 1994.

[23] Daniel J. Van Hook, James O. Calvin, Michael K. Newton, and David A. Fusco. An Approach to DIS Scaleability. In *11th Workshop on Standards for the Interoperability of Distributed Simulations*, pages 347-356, September 1994.

[24] Daniel J. Van Hook, Steven J. Rak and James O. Calvin. Approaches to Relevance Filtering. In *11th Workshop on Standards for the Interoperability of Distributed Simulations*, pages 367-369, September 1994.

[25] Rob Vrablik. ModSAF Interest Management Overview. Personal Communication, November 1995.

[26] James F. Watson. JPSD Interest Management Overview. Personal Communication, November 1995.

[27] Webster's Ninth New Collegiate Dictionary. Merriam-Webster Inc., Springfield MA, 1984.