



Extending Simple Network Management Protocol (SNMP) Beyond Network Management: A MIB Architecture for Network-Centric Services

Title	Extending Simple Network Management Protocol (SNMP) Beyond Network Management: A MIB Architecture for Network-Centric Services
Item Type	Conference Paper
Authors	Gateau, Jamie;Bordetsky, Alex
URI	https://hdl.handle.net/10945/37453
Date Issued	2008
Rights	This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.
Download date	2026-04-14 12:09:03
Link to Item	https://hdl.handle.net/10945/37453

Downloaded from NPS Archive: Calhoun

12TH ICCRTS
"C2 for Complex Endeavors"

Extending Simple Network Management Protocol (SNMP) Beyond Network Management: A MIB Architecture for Network-Centric Services

Topics:

Networks and Networking, C2 Architectures, C2 Concepts, Theory and Policy

Authors:

Jamie Gateau, Naval Postgraduate School (Student)
Dr. Alex Bordetsky, Naval Postgraduate School
Others TBD

Point of Contact:

Jamie Gateau
Center for Network Innovation and Experimentation
Naval Postgraduate School
589 Dyer Road, Room 200A
Monterey, CA 93943-5000
+01 540 455 7579
jbgateau@nps.edu

ABSTRACT

The promise of the Global Information Grid (GIG) includes connecting sensors, shooters and decision-makers who may not be physically co-located in a manner efficient for combat employment, decision-making and information sharing. Current information architecture strategies, such as Network-Centric Enterprise Services have started down one path, requiring the implementation of a Service Oriented Architecture (SOA) and all the requisite underpinnings thereof. These are, for an organization the size of the DoD a very large problem set in and of themselves. An additional unfortunate side effect of choosing a conventional SOA as the backdrop for the GIG is that only those devices capable of running an entire webserver/database stack are able to participate in the architecture, effectively excluding computationally constrained devices. Additionally, the connectivity requirements in a conventional SOA restrict participation by bandwidth-constrained and intermittently connected entities. This paper investigates one possible solution, utilizing SNMP as the language and mechanism for sharing data between disparate systems. Specific decision-support MIBs will be developed to allow transmission of decision-specific information in both push (TRAP/SET) and pull (GET) directions.

I. INTRODUCTION

The Global Information Grid (GIG) and FORCEnet, the Navy's 21st Century construct for network-centric warfare, promise a manifold increase in combat power, fueled by information. In many cases, however, we lack the architecture and the mechanisms for the transfer of the requisite information. We present one information architecture, extending the Simple Network Management Protocol (SNMP), that addresses the necessary underlying requirements of such a network-centric transformation and expanding on the previously introduced concept of Hypernodes.

Hypernodes allow for the description of and exchange of information throughout the battlespace. By taking a service-oriented approach to the functions of the military as a network, Hypernodes expose and allow for sharing of these services, connecting providers with consumers and arbitrating data exchange. This is directly in line with the current move toward a service-oriented architecture for the GIG, but extends the concept of services significantly to include human actors, groups, relationships and decision states as important elements of the network.

Further, the utilization of SNMP allows for a common format for exchanged information that is already accessible to most network-attached elements. It further reduces the complexity to implement a service-oriented architecture by removing the need for specialized software. This has the potential of moving closer to the goal of a fully service-oriented GIG by allowing even computing- and bandwidth-constrained elements to participate fully.

Overall the Hypernode construct and SNMP architecture proposed in this paper suggest an interoperable way to achieve significant impact for network-centric warfare. In this paper we apply this construct to a campaign of experimentation focused on Maritime Interdiction Operations/Maritime Domain Awareness in order to test its applicability. Utilizing a case-study approach and the constant comparative method of theory generation, candidate SNMP Management Information Bases are developed as a first step toward realizing this architecture.

II. HYPERNODES AND THEIR MIBS

A. HYPERNODES

We will start by extending the existing SNMP concepts of *managed devices* and *management stations* with a third class of SNMP-connected nodes: Hypernodes. The original concept of Hyper-nodes derives from Bordetsky and Hayes-Roth (2006), wherein they suggest that the 7-layer OSI model requires an additional, 8th, layer, comprised of network-management-aware functions. Those network nodes which are 8th-layer aware, then, are Hyper-nodes. The 8th layer of Bordetsky and Hayes-Roth, it must be clear, is not the same as the 8th layer discussed in Bauer and Patrick (2004) and adopted into the O-I-T model (Gateau, 2006). Particulars of terminology aside, their fundamental argument is sustained and is extended by this paper.

Hypernodes¹, in this case, consist of three different classes of network devices: those which are network service aware (in the case of Bordetsky and Hayes-Roth, the specific service is network management), those that are subnetwork aware and those that are decision support aware. While these three classes seem widely divergent, in truth, they all share significant commonalities. All will be implemented in an SNMP architecture, and all utilize conceptual extensions to SNMP to facilitate the additional functionality. Additionally, none of these extensions needs to affect the underlying network, SNMP standards or any network nodes which are not, themselves, Hypernodes.

We propose the use of MIB space within the US DoD OID hierarchy. Unlike the Internet portion of the DoD's OID space (familiar to network managers as 1.3.6.1), general DoD applications are allotted space within the 2.16.840.1.101.2 (joint-iso-ccitt (2) country (16) us (840) organization (1) us-government (101) dod (2)) tree. As the first and second children of this tree are already in use for infosec (1) and X.500 (2), we propose to use 2.16.840.1.101.2.3 as the Hypernode space. Unfortunately, we have been unable to receive a grant of space in this portion of the tree. An OID has been applied for in the 1.3.6.1.4.1 space, however, and was granted as: 1.3.6.1.4.1.28291.

1. Network Service Aware Hypernodes

One problem that often vexes network users is the need to find and access network services. We know, for instance, that on the Global Information Grid (GIG) there are a large number of systems providing weather databases. We also might suspect that there are a number of locations where we might go to find the status of the network or where imagery of a certain geographic region might be found. These network services are the heart of the GIG concept, but

¹ We choose to use Hypernodes instead of Hyper-nodes in this paper to differentiate them from the original Bordetsky and Hayes-Roth construct.

quite often remain unknown and unaccessed simply because there is no uniform manner for finding and gaining access to these services.

In a service-oriented architecture built on Web Services², we would implement search via UDDI (Universal Description, Discovery and Integration) and expose the services themselves as web services. This means, though, implementing an entire Web Services architecture 'stack' on every service-providing asset in the network. While this is not a particularly difficult task for many of the "back-office" sorts of nodes found in datacenters ashore and removed from the front lines, it is fairly onerous if one considers that every node is potentially a service provider. These nodes may have very few slack resources to devote to secondary tasks, and the addition of such infrastructure may overwhelm the computing capabilities available on a UAV or to a dismounted infantryman.

Utilizing SNMP, however, will often allow such a web service architecture to be implemented without having to resort to any extra software on the end nodes. As mentioned before, nearly all network-attached devices have or have available SNMP agents. The addition of Hypernode functionality to them requires only the development of a Management Information Base (MIB). These MIBs might be specific to the functionality provided or could be fairly generic in the cases where custom development is uncalled-for. Since the MIB in and SNMP implementation is merely a database schema, the addition of such functionality would represent relatively little difference in the amount static resources (hard disk, for example) required and would only require dynamic resources (RAM, CPU) when the MIB was actually being accessed.

A conventional SNMP-managed node is not aware of the fact that network management is, in fact, ongoing. Nor is its SNMP process aware of any other services that are being performed by the node. There is, though, some ability to report upon those services. Some application MIBs have been developed to allow for remote management and monitoring of those applications. The Oracle-database-MIB is a good example (1.3.6.1.4.1.111.4). With this MIB, it is possible to retrieve variables related to the operation of the Oracle database such as the number of user calls or the number of user commits (1.3.6.1.4.1.111.4.1.1.1.22 and .23 respectively).

This does not, however, tell us *what is in the database* or *how to access it*. Answers to these questions might be expected to be found in the Web Service description, in the case of a conventional SOA, but without such information, a service consumer has no way to make use of the data contained. What is needed is a new extension to the commonly used SNMP MIBs that makes such information available. A network service aware Hypernode meets this requirement by containing within its MIBs an explicit description of the services

² "Web Services" (note the capitalization) specifically refer to the class of Service Oriented Architectures implemented via XML, SOAP and related technologies. When used without capitalization, "web services" refers to the more general class of services provided by networked technologies, regardless of implementation.

provided by the node. The same Oracle database, then, in addition to any generic RDBMS or Oracle-specific SNMP data would expose, at least, the nature of such databases and information on how to gain access.

To continue this example, a node which provides a weather database for a specific geographic region would need to advertise this service somehow. Its MIB should indicate 1) that it is a service provider; 2) that it provides weather data; 3) for which geographic region it has data; 4) how to go about retrieving the data. It might, additionally, include other meta-information about the service it provides, such as the timeliness of updates or, in this case, whether it provides aeronautical, nautical or general weather information.

Bordetsky and Hayes-Roth (2006) suggest this treatment for network management as well. If a node is capable of providing network management capabilities, these should be represented in that device's MIB. This might include simple capabilities, like a wireless access point advertising that it is, in fact, a wireless access point—that is, a network node which provides the service of extending the network wirelessly, or more complex capabilities, such as a node which streams video and is capable of modifying the output data rate (thereby managing the amount of data traversing the network). When all these nodes advertise these services via SNMP and, to the extent possible, allow them to be modified via SNMP, they are network service aware Hypernodes.

a. Extending the Network

To illustrate how such a concept might be realized, consider a simple subnetwork made up of only three nodes. One node is our dismounted infantryman, perhaps a special operator, who is carrying a network-attached sensor capable of multi-spectral imaging. The second node is back at base camp, a standard PC, connected via a robust network link back to the rest of the GIG and being monitored by a local decision-maker in need of the imaging data our SOF operator is acquiring. The third network node is a UAV. As long as the SOF operator is within a reasonable distance of the base camp, he can continue to send imagery and the decision-maker has access to his services. See Figure 1.

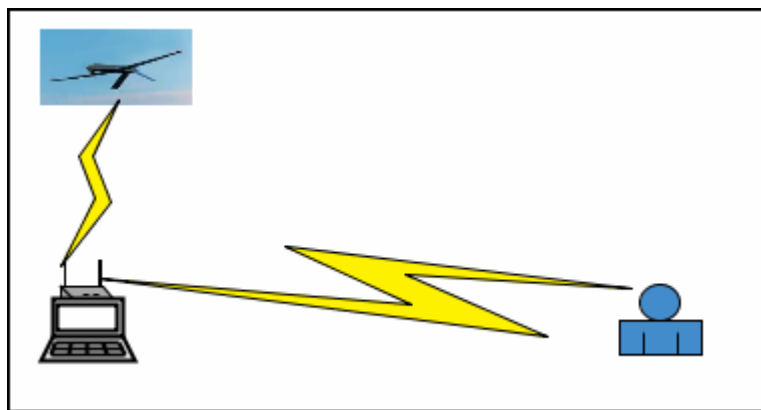


Figure 1. Line of Sight

However, as he moves farther away, the radio can no longer establish communications (or more likely is no longer capable of providing sufficient available capacity to meet the requirements of the imaging). In the case of Figure 2, our network service aware Hypernodes can make this information known to both ends. The SOF operator might see this diminished capacity and attempt to relocate or reduce the demand created by his activity (such as reducing the refresh rate or resolution of his imagery). The decision-maker might attempt to increase his transmit strength (via SNMP commands to the transmitter!) If he can accept degraded video quality, he might make the same attempts to reduce demand as the SOF operator.

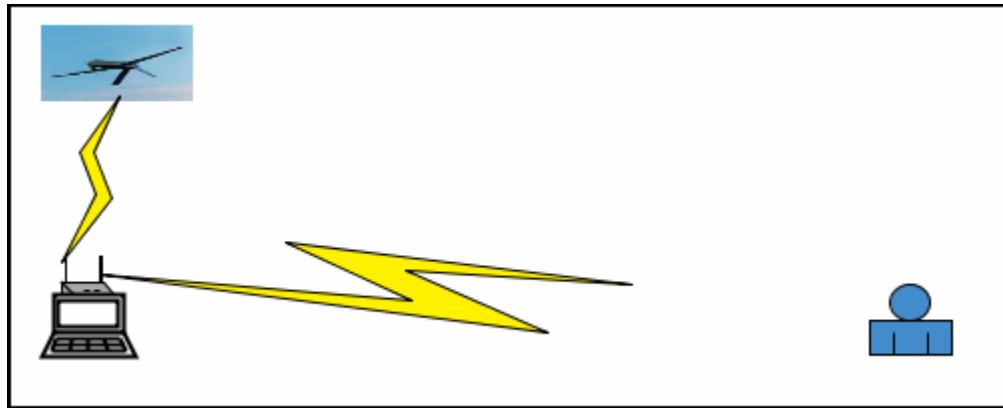


Figure 2. Loss of Line of Sight

This will not always solve all the problems, however. What may be needed in this situation is additional capacity as opposed to reduced demand or a simple repositioning.

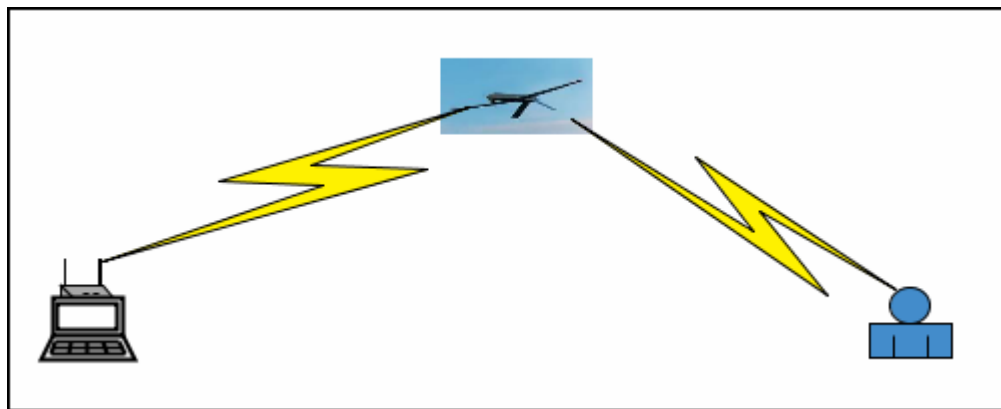


Figure 3. Link Restored

Fortunately, the UAV in the area is also a network service aware Hypernode. This UAV can be used to relay radio communications to the SOF operator. When the decision-maker at base camp searches for a node with the ability to extend his network, he finds the UAV and sends it out toward the SOF operator as shown in Figure 3. If the UAV is particularly sophisticated, it might

be able to process a request such as "maintain 300Kbps on this link," and do what maneuvers are necessary to maintain that as conditions and the location of the SOF operator change. Even without that capability, though, the decision-maker can continue to adjust the location of the UAV to maintain such link capabilities in a human-in-the-loop manner. (As we will see below, we may also wish to model the services provided by that human-in-the-loop.)

A more sophisticated multiple-criteria problem exists, however if that UAV is, unlike this simple case, attempting to maintain several network connections throughout the battlespace. If the UAV is, in fact, a sophisticated Hypernode, as described above, it might attempt to solve the problem itself, alerting human users only when it can no longer communicate. If it is unsophisticated, the humans would need to arbitrate for themselves or at higher headquarters. Here, too, Hypernodes can play a role. Since the base camp is ultimately performing a service—whatever his mission is—this, too, can be entered as a service in the SNMP MIB on his computer. Now, instead of querying all the entities vying for access to the UAV, higher headquarters need only query their Hypernodes.

b. *Hypernodes Representing Humans or Groups*

This is an additional potential for SNMP Hypernodes and should not be overlooked. In the usual fashion, SNMP agents only report on themselves—the computing resources being managed. They return requests for data about the hardware and the software processes running on them. There is no reason, however, that this must be so. If an entity—a command, a unit, a fire team, an external expert—is capable of providing services to the network, they can be represented in a MIB.

In this case, the SNMP agent must still run on some network-attached computer (at least until such time as we are, ourselves, network attached computers). Any computer will do, but only one should be identified as the Hypernode representing a Human or Group (of course, there is no reason that a single computer could not be the Hypernode for more than one). An obvious choice might be for the command's web server to function as the Hypernode for that command. The SNMP MIB for this command would then enumerate the capabilities of the command and point to the Hypernodes for any subordinate commands.

When describing Hypernodes which represent humans and groups, another interesting possibility is suggested. Humans and groups of humans tend to develop relationships with one another. These relationships, themselves, become services that can be accessed on the network. Hence, service aware Hypernodes should also be *relationship aware* in the case of humans. By expressing these relationships in a MIB, it becomes possible to look at a network of Hypernodes connected by relationships as having emergent capabilities themselves. Further, the modification or addition of relationships could be used to reconfigure such a network for other or more varied tasks.

In the next few sections, we will be dealing with one example of a network into which the addition of SNMP Hypernodes is suggested. In that case, there are a number of external experts connected to the network. These non-military entities may be unknown to the participants in the network, but may have crucial services to provide. Establishing a network service aware Hypernode for these experts allows other network members to either query for required services or investigate the services provided by these nodes.

In order to facilitate this kind of search, and the others mentioned in this paper, some subset of a universally agreed-upon taxonomy will be required to name the provided services in meaningful ways and to allow for informed searches. The development of such a taxonomy is beyond the scope of this paper, but an obvious start is provided by the DoD XML registry: <http://metadata.dod.mil>. The same taxonomy used by the DoD to describe XML tags may be useful for our purposes.

2. Subnetwork Aware Hypernodes

The SNMP specification (Case, et al., 1990) specifically allows for the creation of proxy agents. According to the specification, these proxies might be used to either translate SNMP requests and responses to and from a second protocol or to forward messages to nodes that are not addressable using the transport protocol used by the Network Management Software (NMS). While powerful, in the first case allowing us to manage non-SNMP enabled devices with SNMP and in the second case allowing us to traverse transport protocols (perhaps we need to manage devices on an AppleTalk segment from a TCP/IP network), in both cases the proxy agent is expected to merely translate and forward the request to the destination agent.

We will discuss Hypernodes repeatedly as those SNMP nodes which serve specific functions on a network. There need not be anything more unique to Hypernodes than that they implement portions of the Hypernode MIB—there is no other hardware or software that is required for a node to be a Hypernode. As such, we introduce the concept of Hypernodes that are aware of their existence in a network hierarchy. These Hypernodes that are aware of and responsible for connected nodes are subnetwork aware Hypernodes. For the specific case of subnetwork aware Hypernodes, those devices for which they are responsible will be referred to as child nodes. While there is no specific technical reason to enforce such a rule, it is suggested that a network node be a child of only one Hypernode at a time. It is possible, of course, to simply remove duplicate information after aggregation at a higher level, but this requires overuse of transmission resources and adds processing complexity.

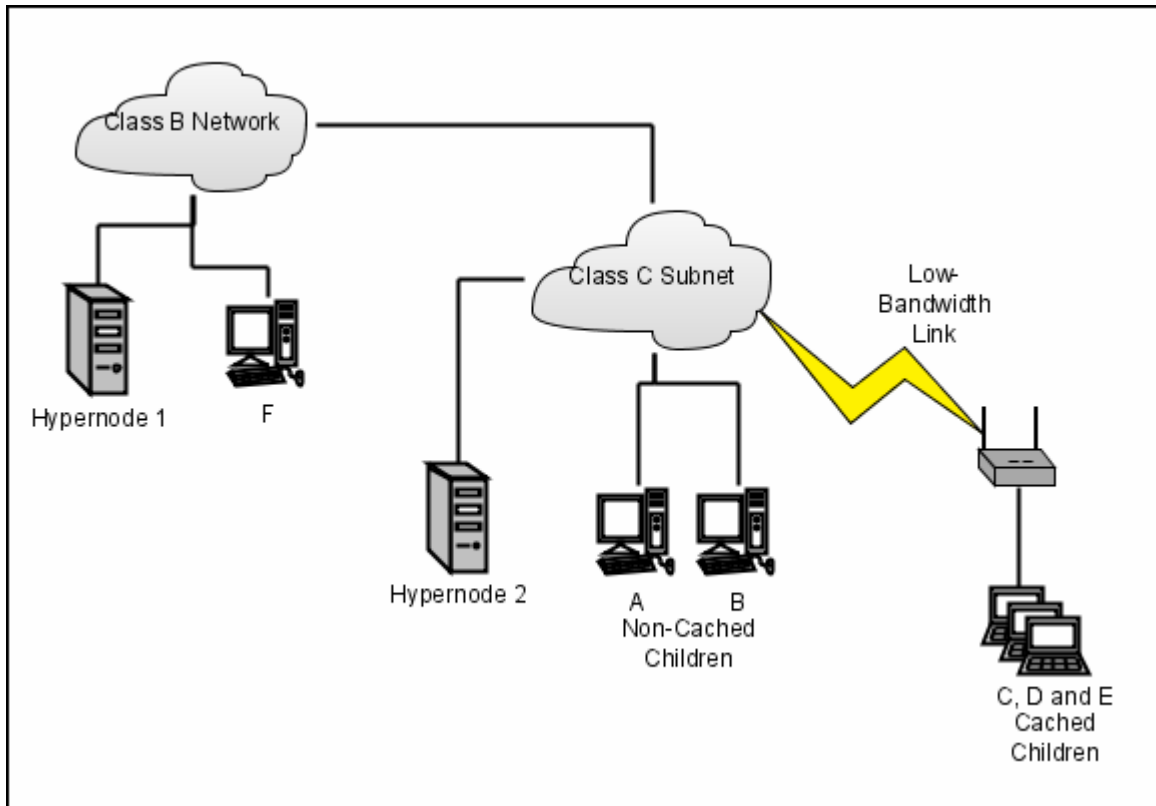


Figure 4. A Network with Hypernodes.

This is not to suggest introduction of a single point of failure in a subnetwork. Contrarily, the computing devices acting as Hypernodes may belong to a cluster, allowing them to share all their collected information and respond on behalf of any node in the cluster in case of a failure or network segmentation. Cluster membership information will be shared with both child nodes and higher-level Hypernodes to allow for redundancy, fault tolerance and automated failover. Figure 4 describes one possible small network with subnetwork aware Hypernodes.

a. Overcoming Bandwidth Constraints

If we consider a network where some of the nodes are on bandwidth constrained links, such as low data rate radios, simply forwarding SNMP requests can quickly saturate the entire capability of a link with just management traffic. What is suggested as a solution in this case is a caching proxy. A subnetwork aware Hypernode is precisely this sort of a proxy. The bandwidth-limited nodes could be configured to only communicate via SNMP trap messages to the Hypernode at some interval appropriate for their communications link. When the Hypernode receives an SNMP request for the subnetwork for which it is responsible, it answers the request itself, replying with data from the cached trap messages.

The details of such an implementation are, in some cases, left for future research, but a few guidelines are proposed. Hypernodes should contain

a table of SNMP MIB values/OID pairs under the Hypernode OID tree. Any MIB data (both network management and services types) that a child node sends to a subnetwork aware Hypernode will be saved into that portion of the Hypernode's MIB. A simple Boolean value will also be included to signify whether the Hypernode should answer on behalf of the child device or not. When a Hypernode is queried, then, it will respond with MIB data on behalf of those children it is instructed to and respond with addresses of those children which desire to be polled directly.

The obvious advantage over other methods is the homogeneity of protocols allowed by SNMP Hypernodes. Currently, such a proxy caching system as described above requires a separate protocol to be utilized for communications between the top-level management application and the proxy nodes. This severely constrains the choices available for network management software. Worse, the protocols implemented by most large management systems (those which have the ability to aggregate and cache) such as Tivoli and OpenView, are not available outside those tools. If we wish to use any of the information available in these subnetwork aware proxies, we must also use their software. When we look to expand beyond network management functions and share that information to arbitrary users in the network, the requirement to use a proprietary protocol is truly onerous.

b. The Large Network Problem

Another concern involves dynamic subnetworks. SNMP, in general, relies on some external method to discover network nodes. One common procedure is as follows:

1. A network operator is alerted that a new device may be on the network, or that a new network segment has been connected.
2. If the address (range) of the device(s) is known, ICMP Echo Requests (pings) are sent to the address range in question. If the address (range) is unknown, ICMP Echo Requests are sent to the entire network.
3. Any device that responds and is not in the current database of managed nodes is then sent a series of SNMP `get` commands to return the contents of the system MIB. In order to do this, some number of community strings must be tried, either based on default configurations, established procedures or specific knowledge.
4. Devices that respond to SNMP are further queried for details the network manager is interested in (the interfaces MIB, for instance) and are added to the management system.
5. Devices that respond to ping, but fail to respond to SNMP are (perhaps) added to the management system as unmanaged hosts. Status will be monitored by using ICMP and looking for responses to indicate an operational status.

This procedure is fairly straightforward and is used in one incarnation or another in many Network Operations Centers (NOCs) every day. Unfortunately it is not very useful in a highly dynamic environment. A few numbers will help to illustrate this situation. If we consider a very small (class C) network of 254 possible nodes, and allow 0.5 seconds per node to respond to the ping and SNMP requests outlined above, it will take a little over 2 minutes to discover and register the entire network. This assumes, of course, that every device responds. If we have to wait for requests to nonexistent or non-SNMP enabled devices to time out (usually about 2 seconds), this can easily stretch toward 8-10 minutes.

Most network discovery tools are capable of issuing multiple requests in parallel, consequently, it is not necessary to wait the full two seconds for one node to timeout before proceeding to the next one, nor must we even wait for the 0.5 seconds that responding nodes require. For the moderately populated TNT network that will be discussed later (about 40% of the addresses are actually active nodes), this network discovery process occurs in about 35 seconds for a single class C network. If we know, then, that a node has been connected in a specific class C, or we know that an entire class C network has been attached, it should only take about a half a minute to discover it and potentially add it to our management system. This is a perfectly manageable timescale for discovery in relatively static networks of this size.

If, however, our network is a Class B (65,534 addresses) and we do not know what address a new device has taken, the prognosis looks more bleak. At the same rates as above, it will now take nearly 2 hours to scan the whole network. Given the fact that the DoD is issued an entire Class A network (16,777,214 nodes) and that many other Class A and B networks are further connected to it via such tools as Network Address Translation (NAT), finding a newly connected node could take an impossibly long time. When networks are changing on a rapid basis, such as with networked aircraft transiting an area of operations, or mobile ground forces moving from one network coverage area to another, such a long lead time to discover new nodes is untenable.

Transition to IPv6 (Internet Protocol version 6) only exacerbates this problem, since the *smallest* network in IPv6 consists of approximately 2^{64} (1.8×10^{19}) addresses, and networks will, by default, be assigned an address space with room for 2^{80} (1.2×10^{24}) addresses. Even if we can find and register 1000 nodes per second, the smaller of those address ranges would require more than *1/2 a billion years* to fully discover³. Clearly our existing procedures for discovering and adding nodes to our management system do not scale well or deal with rapidly changing networks.

³ In case you were wondering, the larger (default) network size could be completely discovered in just over *38 billion years* if we were able to find and register, instead of 1,000, 1 million nodes per second.

Subnetwork aware Hypernodes also address this problem. In addition to caching and allowing proxy retrieval of connected network device information, subnetwork aware Hypernodes contain information about which devices are in their network and for which address range they are responsible. Every device in the subnetwork, further, contains the address of its Hypernode. As a consequence, discovering *any* node in a subnetwork gives enough information to find all the remaining nodes. If we make the simple assumption that we must know (or can easily determine) one address, that of the router interface on that subnetwork, the problem of enumerating this large network vanishes.

It is important to note that the primary problem we are attempting to overcome here is the need to do a blind search in a sparsely populated network. If we have some way of informing our search, or if we know that a subnetwork is, in fact, fully or nearly fully populated, the results from a search are much improved. Unfortunately, in such a case, we are still faced with interrogating every node to determine the state of the network. Aside from offering a solution to the sparse network problem, if our subnetwork aware Hypernodes are caching management data, we need only interrogate one, presumably fast and well-connected, node, although we will need to interrogate it intensively.

The following flowchart attempts to describe this process at a high level. Upon discovering a node in a subnetwork, we first request the MIB variable which identifies the node as a Hypernode. If the result is positive, then we know the Hypernode address and simply ask for all its data. If the result of the query is negative (not a Hypernode) we then request the address of the Hypernode to which this node belongs. We then repeat our "is Hypernode?" query on the presumed Hypernode. Assuming a positive response (it should be) we then retrieve the subnetwork information.

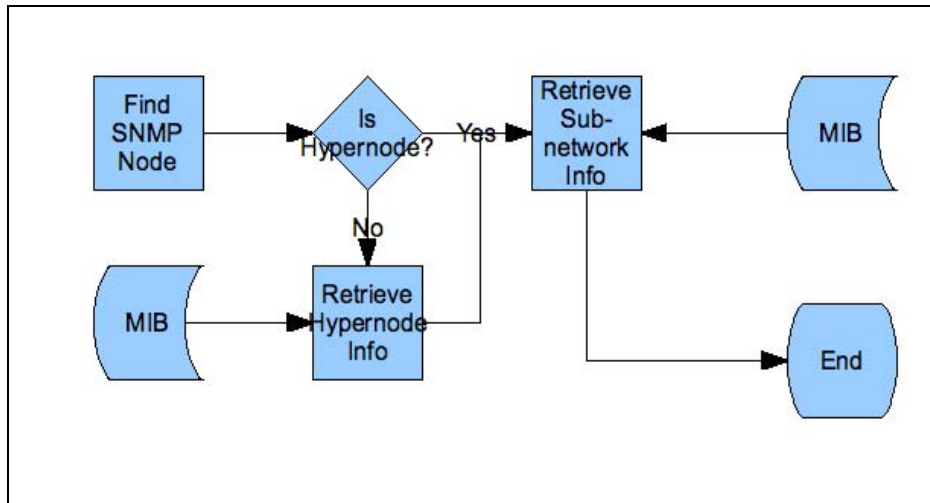


Figure 5. Subnetwork Discovery Flow

New nodes joining a subnetwork (including subnetwork aware Hypernodes whose subnetwork is joining a larger network), then, should first seek out their local subnetwork's Hypernode and register with it. The Hypernode will then forward this information upward as required. Other devices in the network, looking to discover devices or services, need only find Hypernodes (at least initially) within each subnetwork. Given the above process, it can be seen that we have reduced the problem of completely discovering a network address range to a problem of discovering any one SNMP device within it.

With an appropriate hierarchy and forwarding rules, finding all nodes in the entire network is possible by finding any registered node in the entire network. Appropriate forwarding rules ensure that Hypernodes at the root of the network do not have to cache unnecessary large volumes of data, but that bandwidth-constrained nodes need not be concerned with constant polling.

3. Decision Support Aware Hypernodes

The final class of Hypernode is strongly related to the network service aware Hypernode. These Hypernodes represent a special class of network service: decision support. As we will see in the following chapters, one of the primary uses of collaborative technologies we have discovered is to share the results of and request the status of decisions. Unfortunately, making and answering such requests often represents a significant time commitment on the part of all parties involved. By utilizing SNMP MIBs to store these decision states, it is possible to offload much of this work onto computers.

Decision-makers could choose to use SNMP traps to immediately distribute information about a time-sensitive or anticipated decision, while other users might periodically poll, using get commands to retrieve desired information. The implications for the sharing of decisions up and down the chain of command are fairly obvious and could even take advantage of subnetwork aware Hypernodes in order to optimize the use of scarce network resources where necessary in transferring these decisions.

a. *Hypernodes in Edge Organizations*

Potentially even more useful is the ability to look across the organization at decisions that peer or unrelated units have made. If we assume that, moving forward, our military activities are going to require more work with allied, coalition and non-governmental partners, as encompassed by the renewed focus of Military Operations Other than War, (JCS, 1995) the maintenance of information about decision states ceases to be synonymous with military command and control and, instead, becomes an enabler of more Edge-like organizations (Alberts and Hayes, 2003) as groups and individuals anywhere within the organization can immediately access the decision states of all the other involved groups.

Edge organizations, according to Alberts and Hayes allow for leaders to emerge due to their capabilities or expertise in a given situation. Information flow is generally unrestricted, allowing "appropriate interactions between and among any and all members." (2003) This is in sharp contrast to the traditional military hierarchy where information flow is constrained by tradition and by fiat and where leadership is primarily a function of position.⁴ Decision rights, too, are to be distributed to the lowest level possible, allowing individual nodes to make as many decisions locally as possible.

While not a perfect description of the situations created when the military must work with outside agencies, the Edge is a much better description than a military hierarchy, and offers a suggestion on where we should, perhaps, be headed. Hypernodes offer one path for both push and smart-pull information architectures (Hayes-Roth, 2006) that are required by VIRT (Hayes-Roth, 2005) and Edge organizational forms and empower network members with the information they need to make decisions at their level while understanding not just the overall "commander's intent," but also the changing decision states in the network.

As an example, imagine a multi-agency response to a natural disaster. Organizations as varied as FEMA, the Red Cross, the U.S. military, religious aid organizations and local fire and police will, if history serves as any guide, all be involved in the resolution of the disaster. These various organizations will all be making decisions based on the desires of their constituencies or their superiors, and while they are all working toward the same broad goal, the way they conceive of it may be quite different; what the Red Cross expects to do to support a natural disaster may or may not be in line with what the other organizations believe is the appropriate course of action.

We cannot expect these agencies to defer to one another, nor can we expect them to know with whom they should discuss their plans a priori. By utilizing decision support aware Hypernodes, however, these agencies can

⁴ That is not to say that that position was not gained by expertise, or the "sustained superior performance" expected of military leaders. What is suggested is that the specific task, situation or mission may call for leadership skills and expertise not held by such an appointed leader.

simply "post" their decisions and allow others to poll them. When the Red Cross decides to set up a relief supply distribution site, they can enter such a decision, as well as its location and any details they feel germane (types of supplies, what hours it will operate) into their organizational Hypernode for others to find. If, and likely when, they are queried for further information, they can simply update the existing information or choose to include more on their Hypernode.

If the agency believes it is important enough that others need to know the status of a decision, they could, instead choose to broadcast this to all involved parties via an SNMP `trap`. Astute readers will, of course, insist that this could all be handled via other means such as email or web pages. While this is true, the use of Hypernodes is superior in several respects.

No one needs to install/configure/manage web or email servers. This is not a particularly convincing argument in the case of day-to-day operations, but in a case, such as the one above of disaster relief, where the network (both IT and organizational) is being created in an ad hoc manner, simply deciding whose web and email servers to use is non-trivial. Also, this represents excess resource allocations than SNMP, where the agent software is already available to nearly all network devices.

Web and email messages are not particularly machine-friendly. By using Hypernodes to share decision state, SNMP clients can be instructed to take action based on specific updates. Creating the same rules for, especially, web pages is a significantly greater challenge, as the lack of pro forma information on most web sites resolves to a natural language processing problem. Utilization of, for instance, Real Simple Syndication (RSS) would achieve a closer fit to the capabilities of the SNMP model, but at an even higher resource cost.

In the following chapters, we will be dealing with precisely these types of decisions being made by an ad hoc network of actors responding to a Maritime Interdiction Operation. We will see how they interacted with each other and in relation to the various decision states within the network and will make recommendations about the construction of Hypernodes to achieve more efficient information exchange in the future.

III. THE TNT-MIO EXPERIMENTS

A. TACTICAL NETWORK TOPOLOGY (TNT)

Located at the Naval Postgraduate School in Monterey, California, the Center for Network Innovation and Experimentation (CENETIX) has, since late 2004, been involved in a series of experiments collectively known as "TNT." This campaign of experimentation, carried out under the NPS-SOCOM Field Experimentation program involves quarterly field experiments in which a large number of NPS researchers and students investigate various topics related to networking.

TNT is a follow-on to the STAN (Sensor and Targeting Area Network) series of experimentation and is focused on both technologies associated with networking and the human aspects of networked forms of organization. Technologies investigated have included network-controlled UAVs, various forms of wireless networking, a networked Light Reconnaissance Vehicle (LRV), Deployable Network Operations Center (DNOC) architectures and many more. In all of these experiments, the focus has been on both adopting commercially available technologies to military requirements and on investigating the human elements associated with the addition of such technologies to the battlespace.

1. Mission and Objective

The mission and objective of CENTIX, as stated on the center's website (<http://cenetix.nps.edu>) are:

Mission — The mission of CENETIX is to provide students and faculty with opportunities for interdisciplinary study in tactical self-organizing networks, with emphasis on wireless networks, sensors, unmanned vehicles, intelligent agents, and situational awareness platforms.

Objective — CENETIX provides flexible deployable network integration and operating infrastructure for interdisciplinary studies of multiplatform tactical networks, Global Information Grid connectivity, collaborative technologies, situational awareness systems, multi-agent architectures, and management of sensor-unmanned vehicle-decision maker self-organizing environments.

As indicated by the name — Tactical Network Topology — the military requirements under scrutiny are generally of the last mile or tactical nature. As an example, the LRV series of investigations has centered on the construction of a small mobile platform for distributing wireless connectivity from long-haul networks down to the dismounted infantryman. The UAV experiments, meanwhile, have focused on allowing forward operating bases the ability to remotely control simple UAVs over a network to extend their observation range. Various other technologies have also been investigated in order to support the

mission of the U.S. Special Operations Command (USSOCOM), who is a major sponsor.

2. Experimental Network

Each quarter, approximately two-thirds of the experimental time is spent in the field at Camp Roberts, California evaluating these technologies and investigating the human impacts. These field experiments are then connected back to the Network Operations Center (NOC) at NPS via an 802.16 network link. All the network infrastructure involved is operated and maintained by students and is, itself, often the subject of some experimental activity. High-level network views of a few of these sites are given in Figures 6 and 7.

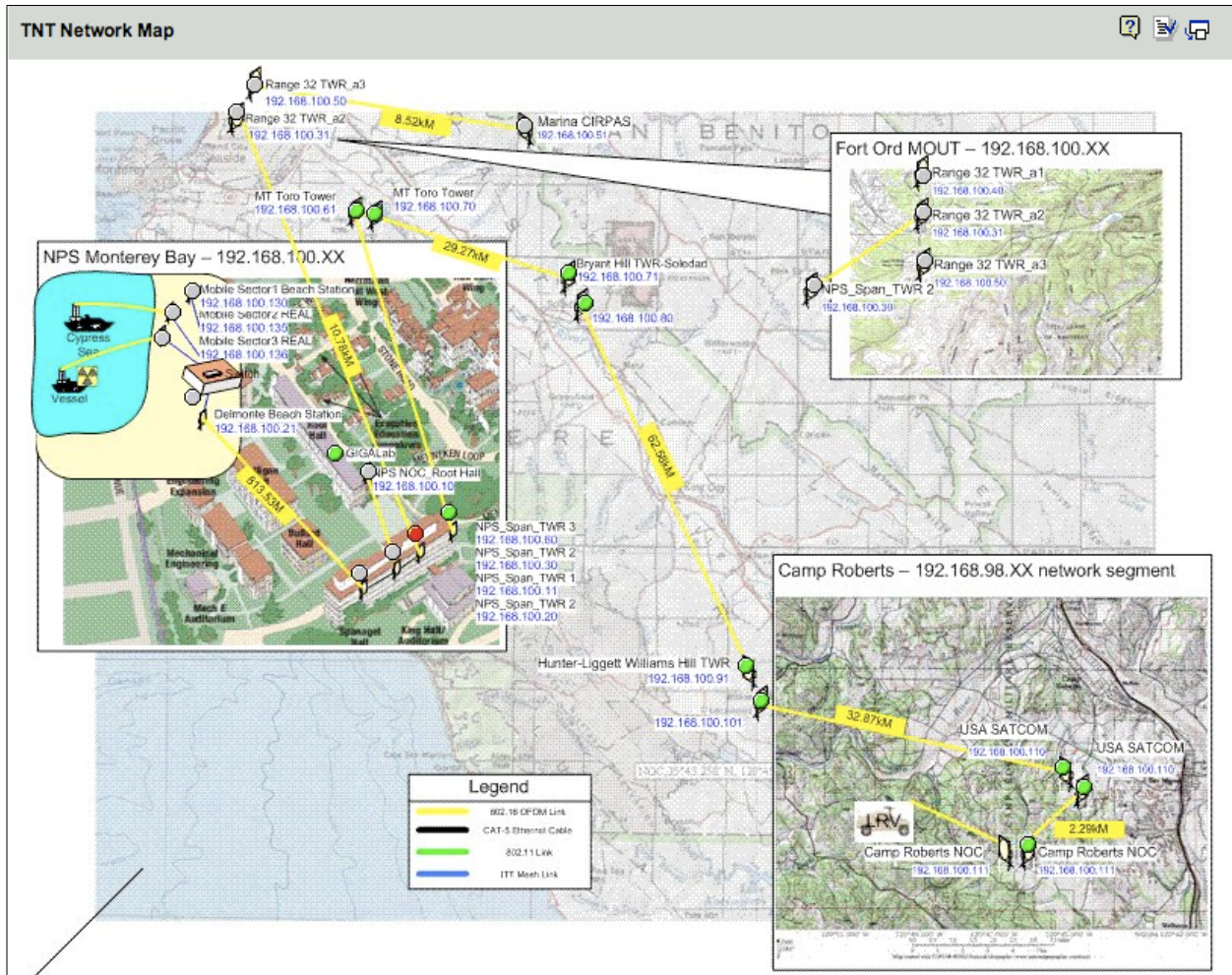


Figure 6. 802.16 Backbone

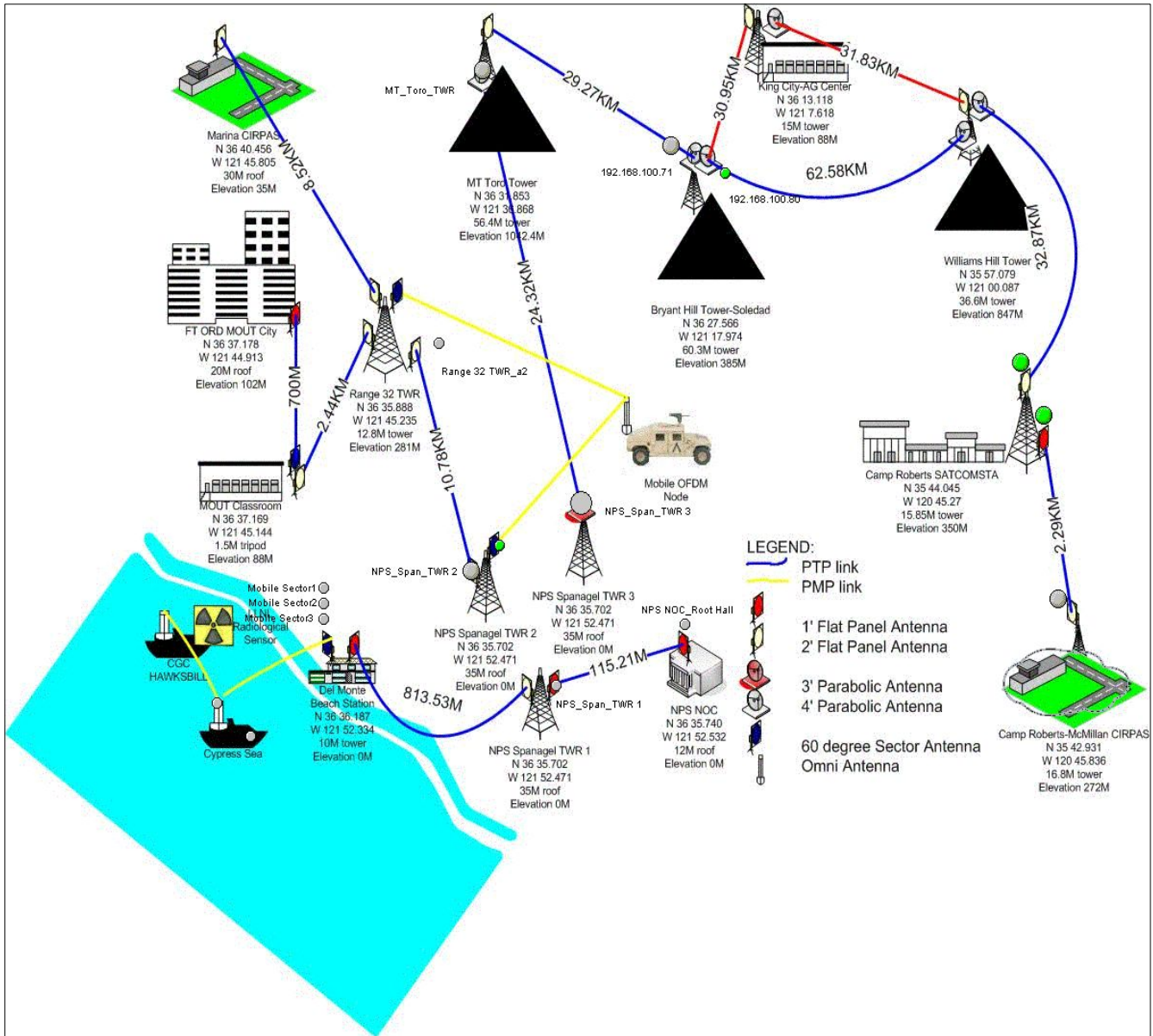


Figure 7. Experimental Network

Figure 6 shows the fixed wireless 802.16 backbone connecting NPS with Camp Roberts. This image is taken directly from one of our network management applications and shows the status of all the nodes in this segment of the network. Figure 7, while several experiments out of date at this point, is a wonderful illustration of the complexity of the TNT network setup. The LRV is indicated as a mobile OFDM⁵ node. Also visible are a number of remote field locations that have been used in past experiments.

⁵ Orthogonal Frequency Division Multiplexing — this is the technology that underlies 802.16.

In addition to the locations at Camp Roberts and NPS, various remote sites are connected to the TNT infrastructure via an ever-changing set of Virtual Private Network (VPN) links, satellite links, iridium phones and other technologies. As such, a large portion of each experiment is concerned with the collaboration and coordination necessary to integrate the large number of sites and interested parties into the ongoing activities. A few of the VPN sites connected to TNT are shown in Figure 8.

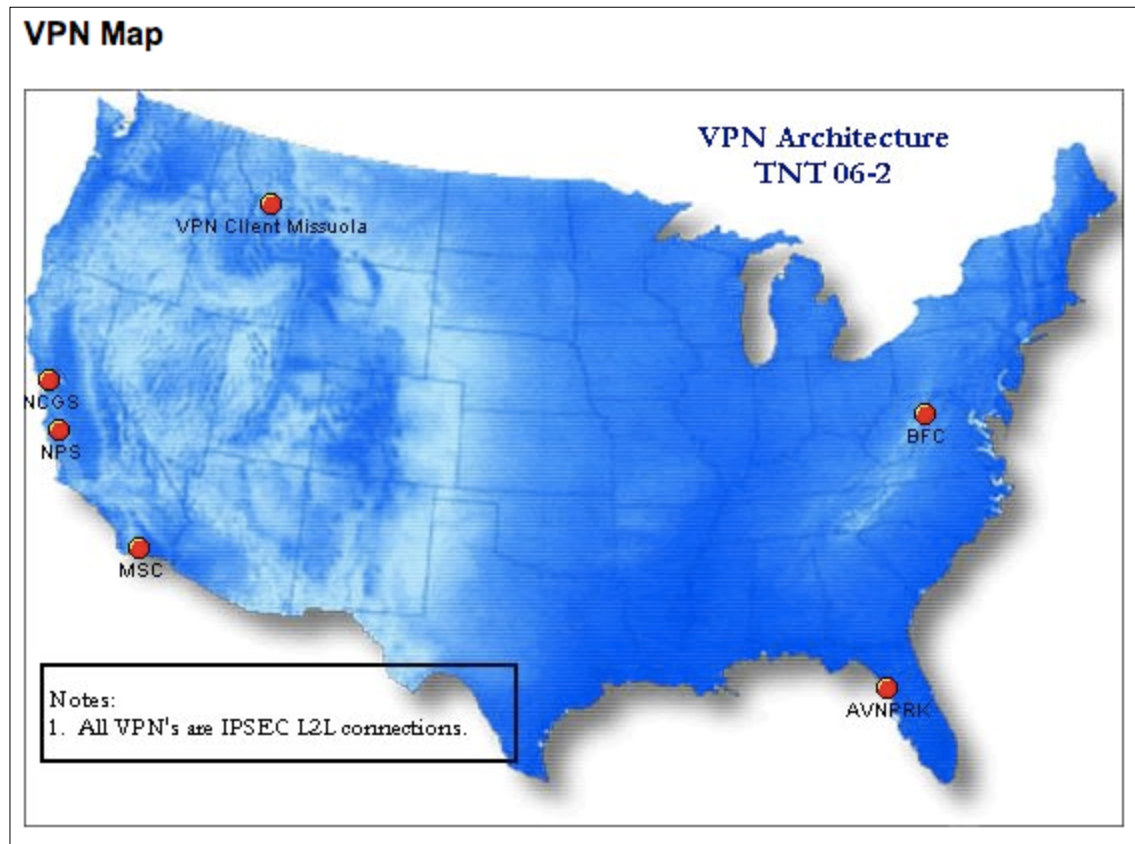


Figure 8. VPN Sites

2. Partners and Sponsors

As they will be involved in the analysis of Section IX, it is worth elaborating on several of the nodes labeled in Figure 8. The Biometrics Fusion Center (BFC), located in West Virginia, has been a member of many of our experiments. They are concerned with our research as a way of connecting remote, tactical field users to biometrics databases removed from the battlefield. In this manner, field agents looking for suspected terrorists can take sensors (fingerprint, facial recognition, etc.) directly to the area of interest while drawing on the full (and likely updated) databases provided by the BFC. Conversely, information gained in the field can be immediately made available to analysts back at headquarters or located in other locations around the world.

The Mission Support Center (MSC) is located in San Diego, California. The Navy Special Operations units involved in the TNT experiments access the network from this location. Additionally, the Stiletto ship, which has participated in a number of experiments, is home ported in San Diego and gains access through this VPN

The remaining approximately one-third of the TNT experimentation not occurring at Camp Roberts is involved specifically with Maritime Interdiction Operations (MIO), usually conducted in the San Francisco Bay area. The network infrastructure that supports the MIO portion of TNT will be covered in the next section, but it, too, is connected to the NPS NOC (and to all the other sites) via a VPN link indicated on Figure 8 as NCGS.

Not visible on this map are VPN links to Austria, Sweden and Singapore. As many nations in the world are concerned with technologies to increase the effectiveness of the Maritime Interdiction Operations, the TNT test bed continues to gain partners who each bring unique capabilities, technologies and operating procedures. All of these serve to enhance both the quality of experiments and the range of variables under investigation.

B. MARITIME INTERDICTION OPERATIONS

As our focus of investigation, we have chosen the Maritime Interdiction Operations portion of the TNT experiments. This highly dynamic, multi-agency series of experiments provides an excellent case study from which many desirable qualities of Hypernodes can be drawn. Further, appropriate evidence is available to suggest utility for all three of the previously identified classes of Hypernodes.

1. What is MIO?

Joint Pub 3-07 defines MIO⁶ as "operations which employ coercive measures to interdict the movement of certain types of designated items into or out of a nation or specified area." (JCS, 1995) Of course, the assumption in the name is that these items are being moved in or out via a maritime avenue (we are not concerned with overland cross-border smuggling, for instance). MIO, then, is primarily comprised of those actions taken to prohibit undesirable shipping to take place. In the current Global War on Terror, this is of obvious concern when we consider the amount of shipping traffic entering and exiting U.S. ports on any given day and the ease with which harmful goods could be brought into the United States.

2. TNT-MIO

The TNT-MIO experiments are focused on several scenarios related to that concern, specifically interdicting the smuggling of chemical/biological and nuclear weapons and nuclear non-proliferation items and identifying known terrorists transiting on container ships. Due to the complexity of this set of tasks and the varied skills required to successfully execute such a mission, our

⁶ Originally, the acronym MIO stood for Maritime Intercept Operations. Recently, the "I" has been repurposed to "Interdiction." This definition was actually written for the former "I," but remains in use.

experiments are focused on creating a collaborative network of experts and linking them with the Navy and Coast Guard operators who are engaged in the actual interdiction. These interdictions take the form of a VBSS (Visit, Board, Search and Seizure).

As the TNT-MIO experiments have progressed, we have moved from simulated boardings of large ships pier side to actual boardings of ships in protected waters (San Francisco Bay). Future experiments will increase the fidelity of the operation by moving the VBSS operation out into the open ocean. If we consider the situation where a potential terrorist is attempting to bring a radiological device into an American port, our ability to interdict him as far away from shore as possible reduces the potential danger to the populace.

A common thread to all of the experiments, however, has been the need to establish network connectivity between the boarding party on board the suspect vessel and the rest of the TNT network (simulating GIG connectivity). Additionally, with a potentially large ship to search and a small boarding team, affording the boarding team connectivity among themselves was also important, allowing each member to reach all the way back to experts located on the other side of the country to help in the accomplishment of the mission.

3. Collaborative Network

The level of expertise required to successfully prosecute a MIO is, unfortunately, more than we can expect of the Navy and Coast Guard members who are actually trained for and tasked with the boarding activities. We can, and do, equip them with appropriate chem/bio and radiation detectors, but even with such advanced capabilities, they lack the skills necessary to interpret the information provided by their detectors.

Conversely, the scarcity of personnel able to interpret such data makes it impossible to simply make sure there is a chemical expert, a biological hazard expert, a radiation expert, a nuclear machine parts expert, etc. on every VBSS undertaken in the MIO environment. By linking these low-density, high-demand experts electronically to the operators engaged in the VBSS, we can significantly increase the level of expertise that can be leveraged against the problem and do it in a timeframe that allows for meaningful action to be taken. Another high-level network schematic follows as Figure 9. This Figure shows the various network nodes in the San Francisco Bay area or connected via VPN during a recent TNT-MIO experiment.

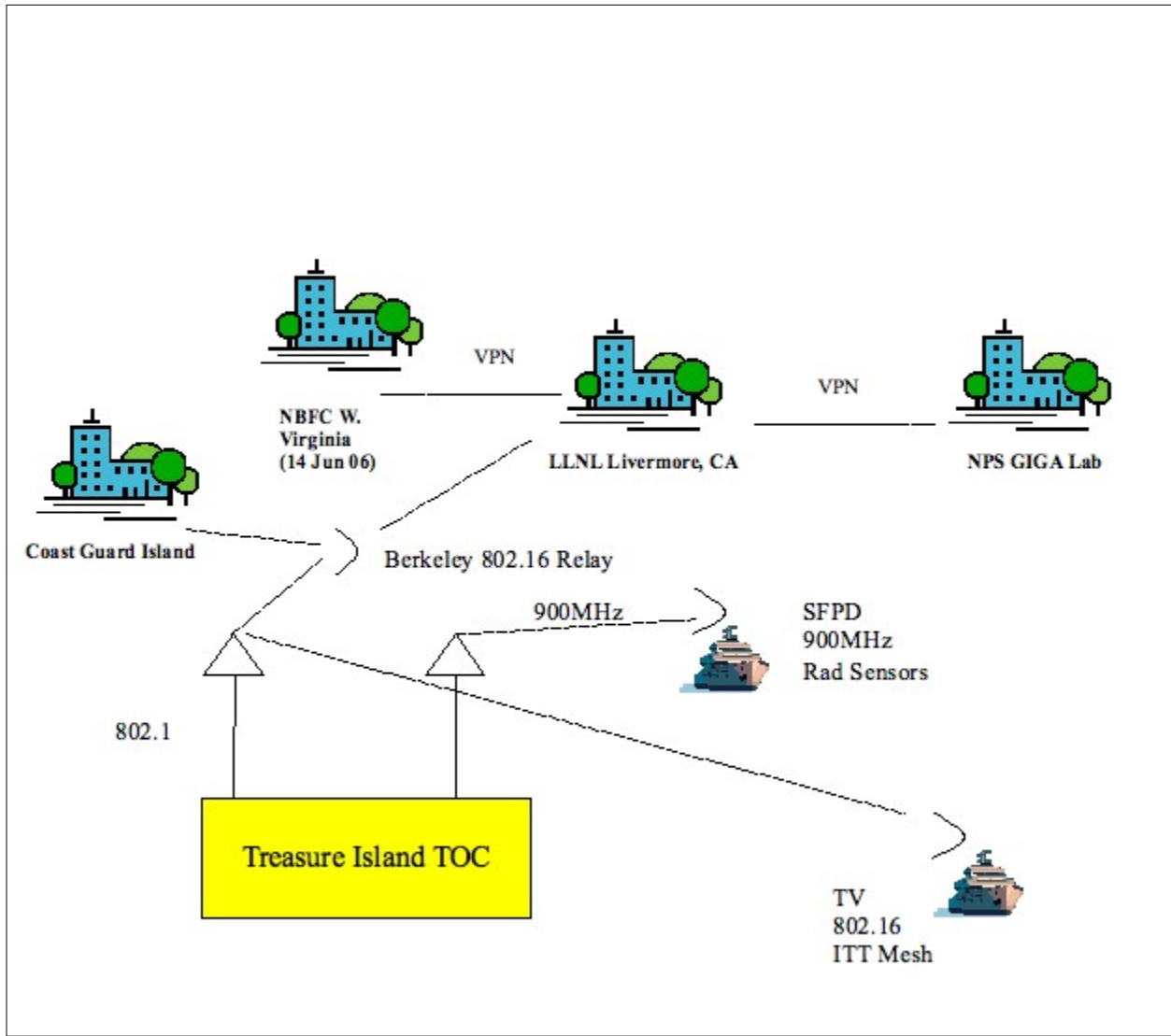


Figure 9. TNT-MIO Sites

In order to allow all the involved parties to communicate, each participant has access to the Microsoft Groove Virtual Office. This peer-to-peer computer mediated communication system allows users to communicate via threaded discussions, shared file spaces, chat and instant message. All parties involved in the MIO simulation were participants in one of several Groove workspaces. In this manner, the boarding party could take pictures of suspected nuclear proliferation materials and immediately make them available to the Defense Threat Reduction Agency (DTRA) agents in the experiment. The DTRA agents could then make a determination on whether the material was problematic or not or if more information was needed.

Groove is not a particularly sophisticated tool, nor is it designed for the kind of operations into which we have pressed it. However, the level of results

that have been achieved in this makeshift fashion are significant enough to suggest that the concept is sound and that continued study is warranted. To understand the power created by linking remote experts with the boarding party in real or near-real time, consider the following: currently, if a boarding party suspects that a crew member or passenger on board a vessel may be a terrorist, they will take fingerprints and pictures of the suspect during the VBSS. If, however, they do not find anything to justify restraining the vessel or arresting the suspect, they will allow the vessel to continue.

Once back ashore or aboard their command vessel, they will submit the information they have in an attempt to identify the suspect. Response to such a request is on the order of hours or days, and that delay only begins after the boarding team has returned to the shore or their command vessel. Often, even if a positive match can be made, the suspect has long since fled, their ship having pulled into port hours or days before the boarding team had any actionable information.

With the use of collaborative technologies and adaptive ad-hoc networking, TNT-MIO experiments have shown the ability to return a positive match within 4 minutes of collecting biometric data. While this is under somewhat controlled experimental conditions, results even within an order of magnitude of this time allow the boarding party to take action while they are still on board the suspect vessel and long before the suspect can evade.

4. Hypernodes and TNT-MIO

As impressive as these results seem, we believe that there is room for improvement and that utilizing Hypernodes will allow us to realize those. For instance, under experimental conditions, the boarding party knows exactly which experts they need to contact in case they require assistance. If they were, instead, faced with a novel situation, how would they discover who to contact? If their network was populated by service aware Hypernodes, they could simply search for one that offered the required service. Finding a radiological source, they would look for a node providing radiation expertise.

Network management, too, is a significant problem during the MIO. The boarding party is unlikely to have, internally, the expertise required to manage and maintain their on-board network or the network link connecting them back to shore or their command vessel. Our current experimental setup attempts to overcome this limitation by utilizing a deployable NOC (DNOC); this gives the boarding party the tools, if not the expertise, to monitor and manage the network themselves. By utilizing subnetwork aware Hypernodes, a centrally located network manager can monitor and maintain even their remote network without severely impacting its performance by constantly polling. Here is an opportunity for improvement.

The decision space is the one where Hypernodes have the opportunity to make the most immediate and obvious impact. Most of the communication in the

Groove workspace is centered on sharing or attempting to gain a shared awareness of the various decision states during the exercise. Has the commander decided there is a need for a more thorough search? Have the results come back from the experts? Do we have orders on how to proceed? Do the experts need more information from the boarding party? Has the boarding party given them the information they need? Capturing and sharing these decision states via Hypernodes can significantly reduce the amount of time and human communication necessary to establish a shared mental model in the collaborative workspace.

5. A Sample Scenario

What follows are excerpts collected from the TNT 06-4 MIO experiments from 29-31 August, 2006. While the names and positions of the specific actors have been removed, as well as many of the purely administrative details, it should serve to illustrate the kind of situations that are indicative of the MIO experiments. The objective of this scenario is:

to continue to evaluate the use of networks, advanced sensors, and collaborative technology for rapid Maritime Interdiction Operations (MIO); specifically, the ability for a Boarding Party to rapidly set-up ship-to-ship communications that permit them to search for radiation and explosive sources while maintaining network connectivity with C2 organizations, and collaborating with remotely located sensor experts.

The experiment extends the number of participating organizations beyond TNT 06-2 MIO to include three international teams in Sweden, Singapore and Austria, Oakland Police and Alameda County Marine Units in the MIO scenario. The networking elements of the experiment are also extended by innovative self-aligning broad band wireless solutions to support boarding and target vessels on-the-move.

The experiment is expected to provide the necessary insight on transforming advanced networking and collaborative technology capabilities into new operational procedures for emerging network-centric MIOs.

The boarding party is faced with finding and identifying a radiation sample on board the target vessel as well as searching for any machine parts that could be a threat (either nuclear technology or bomb parts, etc.) They must also collect biometric data and identify any known terrorists among the crew. Additionally, several overseas participants have been added to increase the realism in the scenario. Information pertaining to the prior identification of some contraband material leaving Austria is available to the boarding party and the remote experts.

As mentioned before, there are some artificialities associated with the experiment, like the remote experts being known to the boarding party and standing by. Aside from that, however, the realism of the experiment is very high. The VBSS is even executed by an actual Coast Guard boarding team while the higher headquarters functions are played by Coast Guard District 11. Due to such realism, and the ability to log all communications in Groove, this provides an excellent source of raw communications data for analysis.

IV. EXPERIMENTAL METHOD AND RESULTS

A. EXPERIMENTAL METHOD

The use of Microsoft Groove Virtual Office as the primary means of communication during the TNT-MIO experiments enables us to capture and analyze, post-hoc, the transfer of information during the experiments. In order, then, to begin developing appropriate MIBs to express the service aware and decision support aware Hypernodes that could support future TNT-MIO experiments and the equivalent real-world operations, we take a qualitative look at this data.

Data are available from three TNT-MIO experiments: TNT 06-3, 06-4 and 07-1. These three experiments took place in June, September and December of 2006 respectively. The experiments are numbered using the federal government's fiscal year system, which starts in October, resulting in the seemingly odd nomenclature for the December experiment.

1. Groove Virtual Office

Groove allows for text-based communication using several tools: chat, discussion boards and instant messaging. Complete text logs are available for the chat and discussion boards from all three experiments, although instant message information is only partial. Because instant message information is purely peer-to-peer, no server-based logging exists of these messages. Where possible, the logs from each participant's software client were obtained, but this results in very spotty data capture.

During each experiment several different "workspaces" were in use. These workspaces (shown in the "launchbar" on the right hand side of Figure 10) were created to insulate the disparate communities of interest within the experiment. The Boarding Party had their own workspace for intragroup communication. The District 11 workspace included the majority of users, the outside experts as well as the Boarding Officer. The TOC and Networking workspace was used as an experiment control channel for the researchers and for network operations communications not related to the experiment.

The utilization of multiple workspaces was due primarily to a need to segregate the experimental control channel (TOC and Networking) from the experiment. A secondary desire was to insulate the Boarding Party's internal communications from the broader group. This is due to a limitation of the Groove software; the only means of access control is at the workspace level, therefore anyone with access to a workspace has access to all the information within that workspace. The Boarding Officer, in charge of the Boarding Party, was a member of both the Boarding Party workspace and the District 11 workspace, thereby allowing filtered communications between the workspaces.

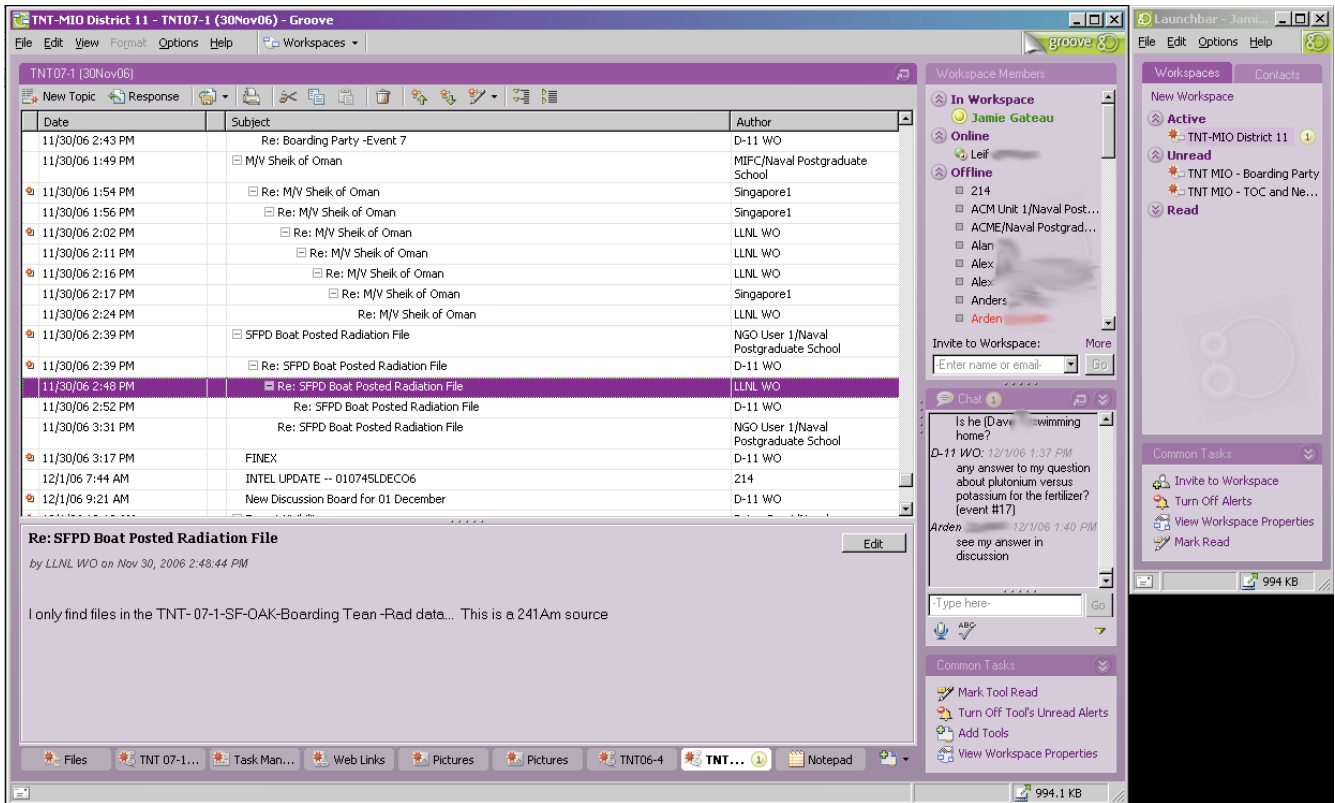


Figure 10. Groove Workspace

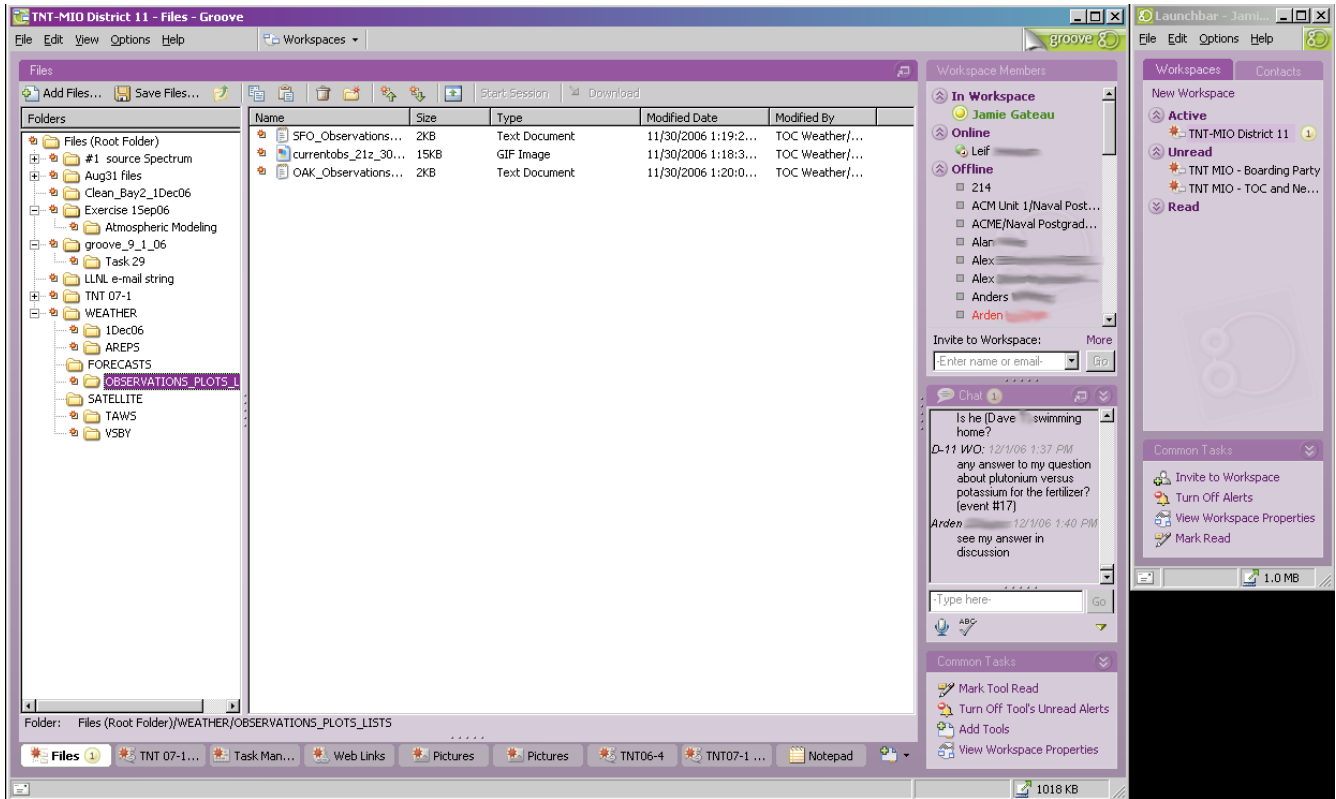


Figure 11. Groove File Sharing

Figure 10 is a screen shot of the Groove software running with the discussion board open in the main window. Discussions are threaded, such that responses to a message appear connected to the original message (see the multiple messages titled re:MV Sheik of Oman). When new messages come in, they are indicated with a red and yellow starburst (several are visible in Figure 20). Because of the threaded nature of these messages, new messages are often not located in temporally related locations; a starburst indication in the tab (bottom of Figure 20) indicates that there is a new message somewhere in the discussion, although it may require some looking around to find.

Figure 11, also a Groove screen shot, shows the file sharing space within the District 11 workspace. As in the previous screen shot, starbursts indicate new files or the existence of new files within a folder, as shown on the left. Figure 12 illustrates the picture utility in Groove. While pictures can be shared in the file area, the picture tool allows users to view the images directly in the Groove software instead of opening them in a separate application. In the normal course of use, both files and pictures are posted in the respective areas and a text message is sent either via chat or the discussion board to alert users.

In all three figures, the chat window is visible on the right hand side of the screen below the user list. This chat tool provides a less structured way for members of the workspace to communicate than the discussion board. A magnified view of the chat tool is shown in Figure 13. This Figure also illustrates the ability of Groove users to resize the various tools to better fit their requirements. Here the chat tool has been expanded at the expense of other tools. Each message in the chat tool is arranged chronologically with the sender identified.

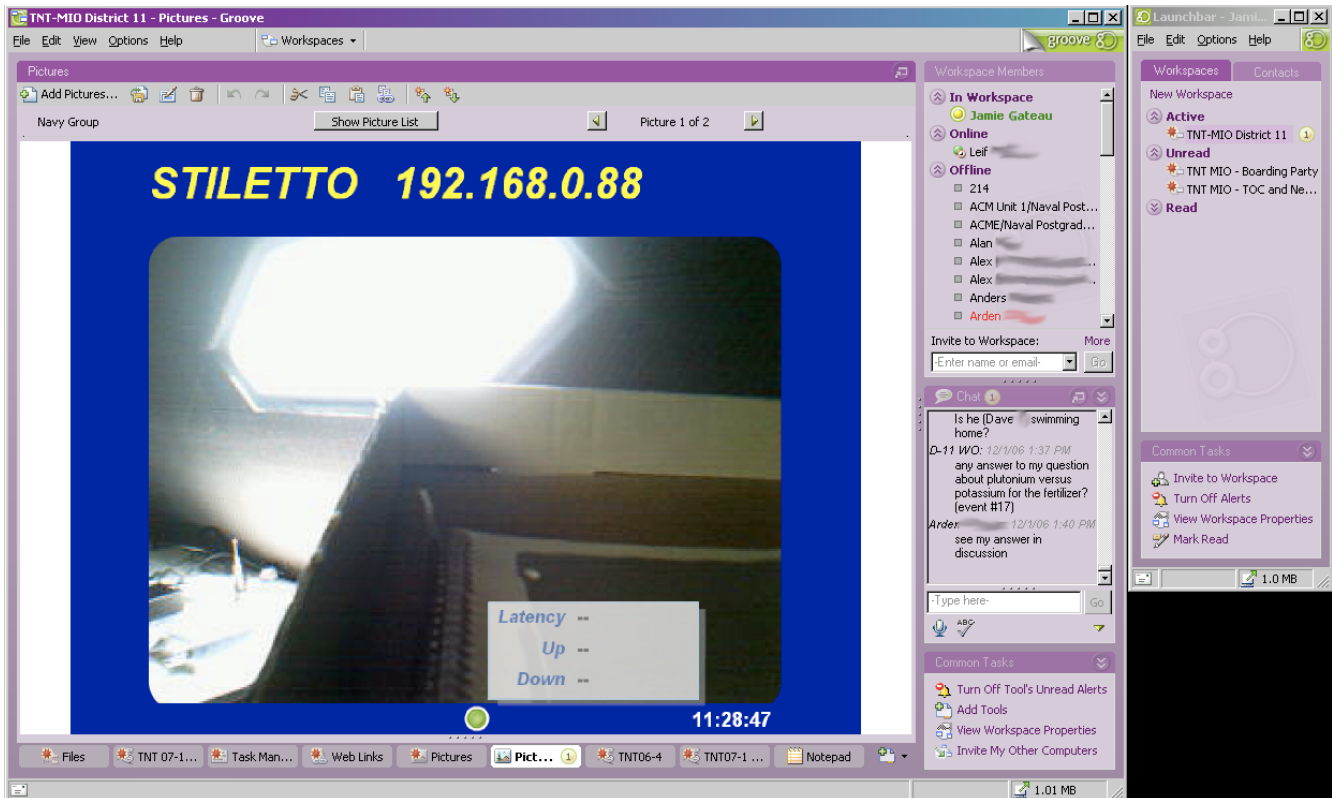


Figure 12. Groove Picture Tool

In addition to the starburst indications, Groove alerts users to new information via pop-up messages in the Windows status bar. Clicking on such a pop-up message takes the user directly to the new message wherever it may be in the Groove workspace. In a busy workspace, such as during the TNT-MIO experiments, it is not unusual for many participants to overlook the arrival of new information, even when it is something they are waiting for.

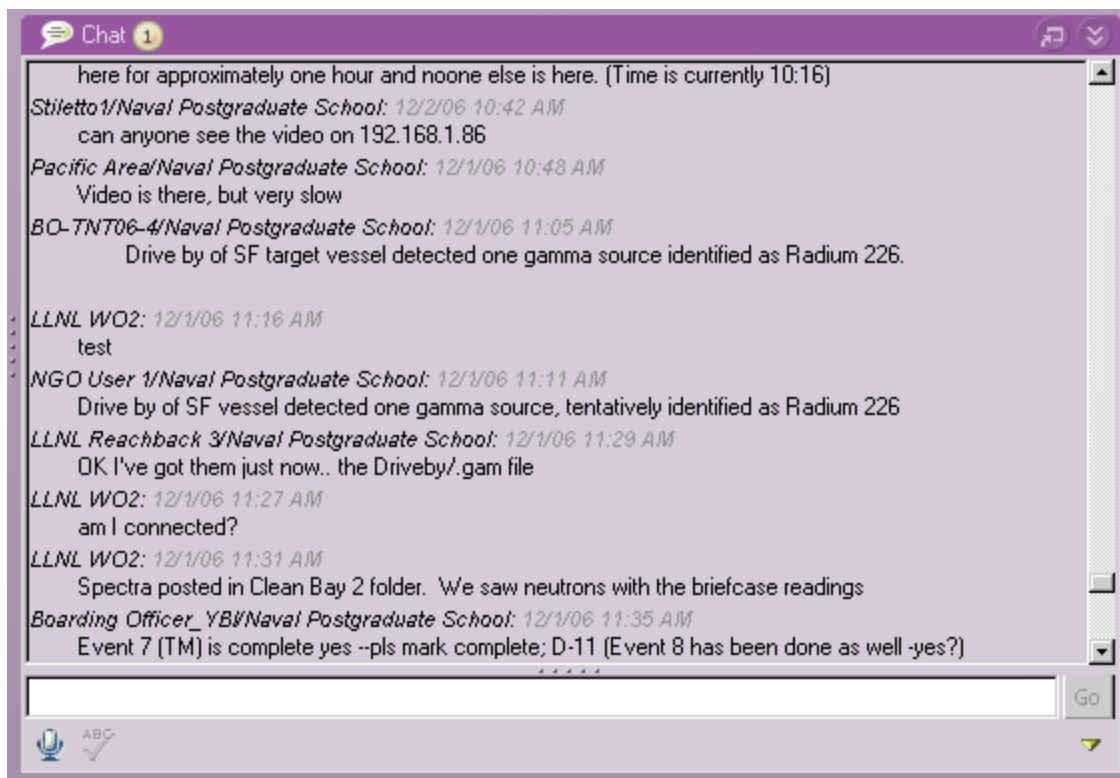


Figure 13. Groove Chat Tool

In recent experiments, we have also tried using multiple discussion threads within a single workspace and relying upon users to contribute only to the correct discussions. This has led to less user confusion, but creates problems with, for instance, the chat logs, since all workspace chat occurs in a single channel. Due to the nature of data analysis undertaken for this paper, none of these should present a detrimental effect to our results.

2. Data Analysis

These data are being treated as three related cases and are analyzed using the case study methodology as suggested in Yin (2003). Additionally, as the nature of communication is generally unconstrained in content, a constant comparison method was used to elicit categorical relationships among the various communiqués (Glazer and Strauss, 1967). Each message was analyzed individually with two foci of investigation: decision states and services.

Since the subnetwork aware Hypernode's MIB is unrelated to the context of the situation, being reflective more of the nature of networking in general and the specific hardware used, our analysis focused on communications which were indicative of the expression of services provided by the network and decision states within the network. The Groove logs for all three experiments were repeatedly inspected, looking for repeated themes. These themes were then collected to determine the nature of information sought by the various participants in the experiments.

When thematic analysis showed a recurring information need (such as querying for a decision state), these needs were considered as potential starting points for MIB development. As an example, from the TNT 06-4 experiment, at time 11:19, there was a request for a confirmation on a radiation detection event. At 11:35, another user was looking for the results of such a confirmation. The original text is as follows:⁷

Request a maritime unit to confirm radiation detection
By Leif on 8/31/06 11:19 AM
Req MU for confirmation

Re: Request a maritime unit to confirm radiation detection
By MIFC/Naval Postgraduate School on 8/31/06 11:35 AM
For ALCO - have you performed the drive-by? If so, have
you posted the radiation files for LLNL reachback?

To understand the format of the messages, note that the first line is the subject of the post. Responses begin with "Re:" and will be indented to increasing levels to show responses to responses, etc. The next line, which begins with "By" identifies the sender of the message. Any following information is the message itself. Because of the relative informality of the message format, some messages have a blank body, as all the information was passed in the subject line.

For coding purposes, the first message above (from Leif) is coded as a "request for services." Leif, in this case, knows that he requires a confirmation on the radiation detection and that a maritime unit is the appropriate element to take action. Had the request, instead been to identify a unit for this confirmation, it would have been coded as a "query for service." The obvious difference is that in one case, we are requesting known services from a known agent. In the other, either the specific service or the agent is unknown.

The reply to this message, from MIFC (Maritime Intelligence Fusion Center) is making two queries. In this case, however, they are coded as "queries for decision state." Sixteen minutes after the initial request for service, at least one participant still does not know the outcome of the activity. Thus, the two-part query is first to determine if the other user has taken an action and second to determine what the result of such an action has been.

A second example from the continuing exchange related to this "drive-by" confirmation follows:

Re: Drive By
By LLNL WO2 on 8/31/06 12:14 PM

⁷ Throughout the next two sections, some minor edits have been made to the original logs, primarily to remove identifying information about the participants.

any info on neutrons?

Re: Drive By
By MIFC/Naval Postgraduate School on 8/31/06 12:17 PM
just talked to the boarding vessel - no info on neutrons - neutron
detector broken

The first question is a "request for information (directed)," as they are looking for more information about the outcome of a previous action and are addressing the request to a specific agent. The second message, however, is a new theme for us, as it actually contains a "response." Specific information which was sought is being offered. This is really a two-part message however, as it both changes the decision state, from unknown to know ("decision announcement"), but also provides the actual response. (It is one thing to know that a decision has been made, quite another to know what the decision is.)

Thus, this qualitative analysis allows us to typify a number of different themes that recur throughout the three experiments. We will then use these themes to inform the creation of MIBs related to them. A treatment of a number of these recurring themes follows. It seems reasonable that future work to develop Hypernode MIBs will need to follow a similar pattern. Understanding the existing information flows within a phenomenon is a prerequisite for developing custom MIB variables if generic ones are insufficient. As this technology develops, of course, the number of exceptional phenomena (those requiring novel, non-standard MIBs) would be expected to decrease.

B. ANALYTICAL RESULTS

The following sections detail the primary themes discovered during analysis. Several of these will be treated further in Section X when we construct some candidate MIBs. Further analysis may, of course, result in more or different categorizations of the interactions that constituted the experimental transcripts. This is only one way of formalizing the interactions and may also not be exhaustive. We have attempted to categorize into as few themes as possible to express the entire range of interactions without losing nuance where such is valuable. Anyone interested in conducting a similar analysis is invited to request the raw log data.

1. Request for Service/Query for Service

Two recurring themes, mentioned above, are the Request for Service and the Query for Service. Requests were very frequent, as participants in this exercise were reasonably well briefed as to the capabilities of the other participants and to whom certain tasks were expected to be directed. It should be obvious, however, that this will not always be the case, and as such, it is postulated that a Query for Service will also be required. This was one of the most frequent exchanges that occurred during the experiments and took many forms.

"LLNL Watch Office: In addition, we need any atmospheric modeling data and any HOPS mapping," and "request drive by to confirm radiation alarm" are two examples of such requests. Another common request was for "reachback." This is a jargon term, understood by the team to mean that external expert assistance was required. They were not always phrased in such obvious ways, however. The statement, "I can't, but LLNL WO may be able to. If there's neutron activation, it's not foundry sand," is an implicit service request to the LLNL WO⁸ to make a determination about a previous statement.

2. Request for Information (Directed)/(Undirected)

Related to the service requests and queries are the requests for information. These came in two distinct flavors: directed and undirected. A directed request for information addressed a specific party (sometimes implicitly) in order to gain some information (again in the Shannon sense (1949)) to reduce ambiguity. These themes occurred as frequently as did the service requests and also took varied forms, from explicit and well-structured, to implicit and ill-structured.

One expansive and well-structured example is,

Got the data. Many questions: Where are the measurements taken? Specific sites? What types of detectors? Gamma? Neutron? Spectroscope? What do the different color curves represent? What was the object that set off the alarm - Cargo, vehicle, individual, etc. Is there an occupancy sensor? This would help me understand background levels better. Is the raw data available? Are there images available?

This message was directed to the agent who had sent a request for service and provided only partial information. These questions, while allowing some freedom in response, clearly delineate the kind of information that would represent an answer. To contrast, "What happened to the drive-by for event 6?" is an ill-structured (and as it turns out, undirected) request for information. This is not a problem for the SNMP MIB or Hypernode constructs, it merely suggests that the agent looking to return an information response may need to ask for amplification or may return data which is not useful as information.

3. Response/Amplification

Obviously, if we are going to make queries and requests, there must be some kind of response. These responses are appropriately numerous in the data and take many forms, mimicking the varied requests that instigated them. A special kind of response also occurred. While the information content is no different from a normal response, an amplification modifies existing data, and, as such, will be treated differently in the MIB.

Amplifications can also be used to modify queries and responses. Since the basic mechanism remains the same—modification of existing MIB

⁸ Lawrence Livermore National Laboratory Watch Officer.

information, there is no difference between these two kinds of amplification in the Hypernode architecture. Thus, "Is this a sodium iodide detector?" (adding specificity to a previous question) and "Target visibility plot updated" (adding information to a previous answer) are equivalently treated.

4. Unsolicited Information

Several times, agents in the experiment offered unsolicited information. This may be in response to a sort of implicit information request that the agent understood to be in effect, or may result from an agent in possession of information that they suspect will be of value to other participants even though it has not be specifically requested. In the SNMP architecture, this is likely best addressed as a trap message, allowing the information source to provide the data to whomever they believe appropriate.

Several messages of the form, "Radiation Alert! File posted in folder Sweden." occurred throughout the experiment. These messages are of apparent importance to multiple actors, since they often resulted in further actions, but were not in response to any specific request. Also, in many cases, an information response to one actor may also be useful to others. Unsolicited information traps, then, allow both smart-push and smart-pull information architectures to be utilized, even both for a single piece of information.

5. Network Registration/Deregistration

Often, as agents came onto the network, they announced their presence. Also, as they left, they made similar announcements. This was often accomplished via the chat function of Groove and suggests these were of lower-priority, or perhaps less important to retain than discussion board messages. This registrations/deregistration keyed other network members to the availability of certain services on the network due to the presence of the member in question. The exchange, "I'm here." "So am I." "We are also here." typifies this kind of announcement.

As a MIB variable, these can be handled as simple registration/deregistration messages. In case of a hierarchical organization, as new members come on line, they simply register with the superior Hypernode. In the case of more Edge-like or flat organizations, network members may choose to register only with a subset of network members, or with all of them. This also suggests another utility for relationship awareness as a network service. Registrations/deregistrations could be propagated through the network as known relationships, allowing late registering nodes to gain awareness of the entire network without requiring the re-registration of all nodes.

6. Service Announcement/Status Announcement

Periodically, nodes found it necessary to announce the existence of new services, such as, "Video feed available at 83.209.68.158." These were generally broadcast-type, undirected announcements, but instead of pertaining to specific information, they reflected the availability of an ongoing capability. Above, we can see that the node with IP address 83.209.68.158 now had

available a video feed. Without more information, however, we do not know of what this might be a video feed.

Fortunately, in this case, the announcement did include some metadata, "(onboard suspect vessel.)" A properly formatted service announcement will need to include enough metadata to be useful to other nodes receiving the announcement. This may be explicit in the service announcement or it may simply refer to some other MIB variables on the servicing node. If, in the previous case, the node had already been identified as being on board the suspect vessel, an announcement of "video feed available" might have been sufficient.

In the case where the availability of a service changed, status announcements were often made. These often took a form such as, "Sorry, that's the best we can do, the identiFiNDER is dead." Here, the actor had to respond to an information request by stating that the information was unavailable, and further that a service the node had previously provided (identification with the identiFiNDER) was no longer available. If such a service later returned, another service announcement would make such a notification to the network.

7. Decision Request

The decision states of the various network actors were of great interest to many other network members. "See ship matching intel description in San Diego Harbor requesting to conduct a drive by for radiation detection," served two information purposes.⁹ First, it offered some unsolicited information (again, there may have been an implicit request for such information, but this was not a response). Second, it requested a decision from higher headquarters: permission to conduct a drive by. It should be clear that this is not a service request, as the service provider is, instead, looking for permission to go ahead.

This is categorized as a "decision request" instead of a request for permission because the underlying mechanism remains the same. Had higher headquarters already decided to execute a drive-by, this decision request would match against that MIB variable and return TRUE. The entity making the request is ultimately asking for the respondent to announce their decision state. There is no need for the decision request to come from a potential executor. It was not uncommon for actors to inquire about decisions made that only laterally affected them. This appeared to be in an effort to better develop awareness of the larger situation.

Other requests, such as, "This is Boarding Officer -based on plume, request guidance on moving the target vessel. Where do you recommend we move?" This one, also, appears to be an information request, but is actually a decision request. The Boarding Officer is looking for someone to decide and then share with him the decision about whether the ship should be moved.

⁹ It is interesting that this entire message was sent as the subject line of the message with no amplifying information.

These requests are often found with either information responses/amplifications or with unsolicited information attached.

8. Decision Announcement/Task Assignment

Although a decision in response to a decision request is merely handled as a response message, in much the same way that actors often made unsolicited information announcements, they often also made decision announcements and status announcements. These are analogous in format, but differ in content from the unsolicited information announcements. In much the same manner, too, a single decision may result in a response to the requester and a decision announcement to one or more other actors in order to disseminate understanding to the entire network.

"I have tasked the boarding team with performing this data collection," is an excellent example. This actor had already tasked the boarding team, effectively passing the status on the data collection decision, but also decided to make the announcement to the remainder of the team. Decision announcements should also set a local MIB variable so that later decision requests immediately receive the decision status from the Hypernode without intervention. Task assignments are also a type of decision announcement as they may or may not result from a specific decision request.

9. SITREP

The boarding party, especially, heavily utilized a specialized information response called a SITREP (Situation Report). These, unlike unsolicited information, are specifically expected by higher headquarters. Like unsolicited information, though, they do not have a one-to-one relationship with an information request. One example is, "sitrep: #4 bio sample sent at 1328 - negative response at 1331 - Done with crew members bio samples - ALL NEGATIVE. Waiting for response on #1 and 2 radiation sources."

Such an information announcement allows all members of the network to maintain awareness of the ongoing situation without a need to positively query the boarding team. Only if more or amplifying information was required, did specific requests get made. Utilizing a SITREP construct in this network reduces the amount of information requests that need to occur. It is anticipated that other expected, implicit, pro forma information announcements will be required in other domains.

V. CRAFTING MIBS

The three candidate MIBs developed for the TNT-MIO scenario we will discuss in this section are available separately. Only individual items of interest will be duplicated here in the interest of space and readability. These MIBs borrow heavily from the syntax and organization offered by RFC-1155 and RFC-1213, so similarities noticed between these and those are not accidental. It should also be noted that there is really only one way to format the data according to specification, the differences among all MIBs stem only from what data you choose to share and how it is organized.

An interesting problem was identified in the construction of these MIBs. Unfortunately, tables in MIBs cannot be nested. This means, ultimately, that table entries can only be leaf nodes and not other tables! Especially in the subnetwork aware Hypernode, this reduces some of the elegance of the solution, but does not result in a loss of capability. This does mean, for instance, that the childMibTable, discussed later, cannot be organized beyond a single layer, placing all MIB values for all child nodes in the same table.

As mentioned in Section VI, this does not mean that the data storage must occur in the same flat table. This MIB, especially, is a good example of a MIB which would benefit from the implementation of a relational database as the storage mechanism for the actual data values. As such, there is an opportunity to introduce another layer of abstraction and better organize the data being held by this Hypernode.

None of the MIBs described below should be considered as exhaustive. They provide only a starting point for future research and examples of how the analysis described in Section IV can be applied to actual MIBs. They are, however, complete and should be implementable as-is. Figure 24 displays these MIBs as children of a yet-to-be-assigned NPS node on the enterprises branch of the Internet (1.3.6.1) tree. A service aware (service-hyper-mib), subnetwork aware (subnet-hyper-mib) and decision support aware (ds-hyper-mib) MIB are each described below.

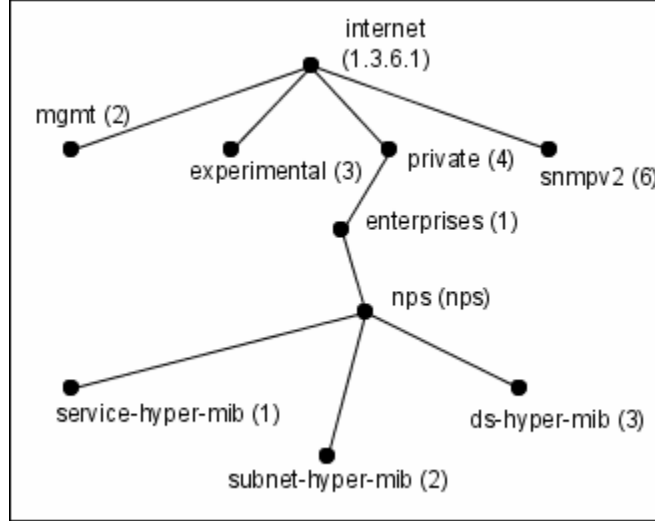


Figure 14. Hypernode MIB Tree

Aside from the three Hypernode MIBs discussed below, a fourth MIB is also offered, describing the highest level of the NPS tree. Currently, this contains only one variable of import to us, an INTEGER of `nodelsHypernode`. If the node in question is a Hypernode, this value will be set using the same formula as `childsHypernode` below. A zero value or the absence of this variable indicates that the node in question is not a Hypernode.

A. A SERVICE AWARE HYPERNODE MIB

The network service aware MIB provided in Appendix D is very generic in its implementation. It seems very likely that as further research is conducted, the details of the provided services table (`proServTable`) will call for a significant increase in the number of variables included. In its current form, this table allows only for querying to find the services provided by a given Hypernode. Any other interactions must occur via a different mechanism. The assumption made by the table is that a uniform resource locator (URL) can be utilized to access the service. This is understood to be a naive assumption at best.

Name	Syntax	Access	Description	OID
<code>servIndex</code>	INTEGER	read-only	A unique value for each service.	<code>proServEntry 1</code>
<code>servName</code>	DisplayString	read-only	A descriptive name of the service provided.	<code>proServEntry 2</code>
<code>servReference</code>	DisplayString	read-only	A unique reference for this service.	<code>proServEntry 3</code>
<code>servDescr</code>	DisplayString	read-only	A free-text description of this service. In the absence of other metadata, this description should be as complete as possible to allow other users to make decisions about the use of this service.	<code>proServEntry 4</code>
<code>servIsMachine</code>	BOOLEAN	read-only	TRUE if this service is automated.	<code>proServEntry 5</code>
<code>servIsAvail</code>	BOOLEAN	read-only	TRUE if this service is currently available. Additionally, a <code>myServDown</code> or <code>myServUp</code> trap should be sent to appropriate users when the Avail status changes.	<code>proServEntry 6</code>
<code>servUrl</code>	DisplayString	read-only	The Uniform Resource Locator where the service may be accessed.	<code>proServEntry 7</code>

Table 1. `proServTable`

Table 1 illustrates the entries in the proServTable in a more human readable format. This Table should be easily understood at this point. There are at least two comments worth making. The servIndex variable should remain as static as possible on a given system. Since the myServUp and myServDown traps reference the index, changes of indexes will cause confusion for other network users. The servIsMachine variable indicates whether this is an automated service or a human-provided service.

Two SNMP traps are also defined in this MIB. These allow announcements analogous to the service announcement/status announcement themes discovered in Section IX. When fired, these traps simply return the OID value for the index of the service in question. If, for instance, a service previously active fails, a myServDown trap is fired to interested parties. We should be concurrently setting the servIsAvail variable to FALSE if this is the case. Similarly, when the service returns to operation, the myServUp trap is fired. In both cases, the payload of the trap is the index of the affected service.

This is also useful if we are bringing a new service on line. When a new service is registered on the Hypernode, the generation of a myServUp trap serves to announce the new service to the network. The fact that we are not actually changing the status from down is immaterial. It may help, in fact, to think of it as changing the status from none to up to understand why this is the case.

Queries for service can be handled by requesting all the proServTables on the network and simply performing a simple search. This is particularly inelegant and we may wish to, instead, have a single Hypernode which does this and then offers "Service Search" as a service itself. This sort of construct will likely require the addition of another high-level npsMib variable so that the search for a "Service Search" node does not regress to the previous problem.

B. A SUBNETWORK AWARE HYPERNODE MIB

The subnetwork aware Hypernode MIB is brutally simple in its implementation, but that is a reflection of the purpose of such a MIB. The primary purposes of the subnetwork aware Hypernode are to enumerate the remaining nodes of the subnetwork and to cache MIB information for any of the nodes that request it. In this first incarnation, the information to be passed is also fairly rudimentary.

Name	Syntax	Access	Description	OID
childIndex	INTEGER	read-create	A unique value for each child. Its value ranges between 1 and the value of childNumber.	childEntry 1
childName	DisplayString	read-create	An administratively-assigned name for this managed node. By convention, this is the node's fully-qualified domain name.	childEntry 2
childIp	IpAddress	read-create	The IP address of this child on this subnet. In cases where there is more than one IP per subnet, the child node will determine which IP to advertise.	childEntry 3
childDescr	DisplayString	read-create	A textual description of the child. This value should include the full name and version identification of the system's hardware type, software operating-system, and networking software.	childEntry 4
childLocation	DisplayString	read-create	The physical location of this node (e.g., `telephone closet, 3rd floor').	childEntry 5
childCached	BOOLEAN	read-create	TRUE if we are caching MIB data for this node.	childEntry 6
childIsHypernode	INTEGER	read-create	A value which indicates the class of Hypernode this child represents. ¹⁰	childEntry 7

Table 2. childTable

For basic information, there is a table of "child nodes," called childTable, and simple count of total children, childNumber. Because of the one-to-one relationship of children to table entries, this means that there are only as many rows in the childTable as the value of childNumber. Table 7 shows the entries in childTable and the salient features of each one. All MIB variables have a STATUS of MANDATORY, therefore, the STATUS information is excluded from Table 2.

Most of the information in this Table should be self-evident. Name, location and description information can be copied directly from the system MIB of the child node if desired. The childCached value reflects whether or not this Hypernode contains MIB information for this child node. Currently, this is only a Boolean value of TRUE or FALSE, meaning that we are either caching all MIB information for the child or none.

¹⁰ This integer value is based on a formula as given in the MIB and Appendix D.

The childIsHypernode entry allows us to identify whether the child node is, itself, also a Hypernode and what kind of Hypernode it is. The integer value of this variable identifies which of the three kinds of Hypernodes currently devised it serves as. If new classes of Hypernodes are identified, the formula which defines this value is easily extensible and is explained in the full MIB. Readers familiar with the Unix attrib command will recognize this formula immediately.

All values of this table are set as read-create with the understanding that child nodes, when registering with the Hypernode, will simply edit the appropriate MIB values in the table. Using read-create, instead of read-write, allows for new entries to be added to the table. It is expected that appropriate security measures will be implemented, either via SNMPv3 or some other means, to restrict access to these variables. If this is not possible, it will be sufficient to override the read-create value with read-only and make changes to this table on from the Hypernode via a programmatic method.

The other table defined in this MIB is called childMibTable, and contains the MIB values for all child nodes for which this Hypernode caches. As currently conceived, this is, quite literally, a table with every single MIB value on the subnetwork in it. As mentioned above, this may not be the most efficient way of actually storing the data, but it does make for a very simple representation. Table 8 describes the childMibTable.

This table represents a completely brute-force method of caching MIB values, but allows for an arbitrary nesting of cached nodes. Since the childMibTable is, itself, a number of MIB values, if one subnetwork aware Hypernode is caching for another each of the child MIB values is also a Hypernode MIB value. If the Hypernode is also cached at a higher level, all the child MIB values of which it is aware will also be passed up to the caching Hypernode.

Name	Syntax	Access	Description	OID
childMibIndex	INTEGER	read-create	A unique value for each cached MIB entry.	childMibEntry 1
childIp	IpAddress	read-create	The IP address of this child on this subnet. In cases where there is more than one IP per subnet, the child node will determine which IP to advertise.	childMibEntry 2
childMibOid	DisplayString	read-create	The OID being cached.	childMibEntry 3
childMibValue	DisplayString	read-create	A copy of the MIB value from the child node. Since we can't be sure of the syntax for the cached information, we use a display string.	childMibEntry 4
childMibDescr	DisplayString	read-create	A copy of the MIB description from the child node. This creates overhead in the case that this is a standard MIB value, but allows us to be self-describing even for previously unknown OIDs.	childMibEntry 5

Table 3. childMibTable

This construct also creates a one-to-one relationship between Hypernode MIB values and child MIB values. For instance, if the System Description MIB variable of a cached node is located in 1.3.6.1.4.1.28291.2.1.3.1.1, retrieving 1.3.6.1.4.1.28291.2.1.3.1.1.4 on the Hypernode returns the same information as retrieving 1.3.6.1.2.1.1.1 on the cached node. Thus, we no longer must query end nodes directly to return their MIB information.

C. A DECISION SUPPORT AWARE HYPERNODE MIB

The decision support aware Hypernode MIB takes much the same form as the service aware MIB. This is not surprising when we consider that decision support could have been implemented as a specific kind of service provided by the Hypernode. Such a decision may be useful in the future, but we have chosen to differentiate them at this stage because they fill significantly different spaces in the TNT-MIO domain.

Table 4 details the basic decision state table. This table, madeDecisionTable, has a slightly misleading title. While this table does contain finalized or made decisions, it also contains new decisions and those in progress. As it is expected that closed decisions will remain in the table for much longer than the decision-making process itself, we expect that the majority of entries in this table will have been previously made decisions at most times.

In keeping with the previous MIBs, most of the information should not require explanation. Two comments are noteworthy, however. The decisionIndex variable should remain unchanged for at least the lifecycle of a decision. If, for instance, a given node makes decisions with a three-day timeline (an ATO cycle, for instance), index numbers should not be recycled inside this frequency. Since the upper bound on integers is extremely large in human terms (232) it will likely be worthwhile to never recycle decision numbers until the counter “rolls over.” It is expected that maintaining unique decisionNames is likely to pose more of a problem than running out of index space.

Name	Syntax	Access	Description	OID
decisionIndex	INTEGER	read-only	A unique value for each decision.	madeDecisionEntry 1
decisionName	DisplayString	read-only	A descriptive name of the decision. This should be a short but unique identifier of the decision.	madeDecisionEntry 2
decisionDescr	DisplayString	read-only	A free-text description of this decision. In the absence of other metadata, this description should be as complete as possible to allow other users to take action based on this description.	madeDecisionEntry 3
decisionIsOpen	INTEGER	read-only	other(0), new(1), open(2), awaiting-amplification(3), closed-undecided(4), closed-decided(5)	madeDecisionEntry 4

Table 4. madeDecisionTable

The decisionIsOpen variable is an INTEGER, but takes an enumerated list as possible values. New decisions, those on which the decision maker has taken no action toward a decision, are classified as 1. Once the decision maker or his delegate has begun to work on a decision, it should be set to 2, open. If the decision maker requires amplification or information and therefore cannot make a decision, the value should be set to 3.

If this decision maker chooses to transfer, abdicate or simply table a decision without a resolution, it should be set to 4 so that other network members know not to expect any further change on this decision from this node. A closed-decided decision (5) indicates that a decision has been made, with appropriate description information in decisionDescr. The value of 0 (other) should be used when no other value is appropriate. In this case, amplifying information should be placed in decisionDescr.

The existence of the madeDecisionTable allows other network members to issue SNMP get requests in order to determine a decision state without needing to directly contact anyone responsible for the decision. This alone would have significantly reduced the amount of discussion board and chat traffic for the TNT-MIO experiment, and when coupled with the ability to issue a trap when decision states changed, could reduce the manual information flow to purely exceptional issues.

This MIB also defines two traps, the second of which applies to this table. The myDecisionChangeState trap is issued any time the decision maker changes the decisionIsOpen variable, indicating to network members that the decision whose index is in the trap has changed state. Unfortunately, due to the nature of the SNMP trap, that single value is all that is available. Interested parties must still then execute an SNMP get to determine the new value of the variable.

Finally, Table 5 describes a vehicle for allowing nodes to ask for decisions. The ds-hyper-MIB also implements a table to support such requests called dsInboxTable. Where the variables in the prior table were all read-only, these are all available to be modified or created by other nodes. It is expected that a separate mechanism will be used to transfer these submitted decision requests into the madeDecisionTable.

The primary difference between this table and Table 9, aside from name changes is the introduction of the dsRequester value. This value contains the IP address of the node requesting the decision.

The remaining trap defined by this MIB, myInboxAccepted, simply notifies the original requester of a decision that it has been accepted for action by the decision maker. This message includes the inboxIndex of the submitted request. After creating the entry in the madeDecisionTable for the new entry, the decision maker's agent should also send a myDecisionChangeState notification, since the creation of a new madeDecisionEntry, necessarily, involves a change in state—from none to, most probably, new or open. In this way, the requester knows both that his decision has been accepted and what the OID is for the index to the decision so that they might monitor it later.

Name	Syntax	Access	Description	OID
inboxIndex	INTEGER	read- create	A unique value for each decision.	dsInboxEntry 1
dsInboxName	DisplayString	read- create	A descriptive name of the decision. This should be a short but unique identifier of the decision requested.	dsInboxEntry 2
dsInboxDescr	DisplayString	read- create	A free-text description of this decision. In the absence of other metadata, this description should be as complete as possible to allow the decision make to take this for action.	dsInboxEntry 3
dsRequester	IpAddress	read- create	The IP address of the node requesting the decision.	dsInboxEntry 4

Table 5. dsInboxTable

These three candidate MIBs go a long way to fulfilling the needs identified during the thematic analysis of the TNT-MIO data. They should still not be seen as a complete solution to a Hypernode implementation, even within this limited domain. They are, however, intended as both a proof of concept and template by which others may improve the capabilities of the Hypernode architecture, and even in this rough form, they offer significant capabilities to address identified requirements within the TNT-MIO experiments.

VI. EXTENSIONS AND FURTHER RESEARCH

Throughout this paper, we have tried to develop an architecture for information exchange based on the Simple Network Management Protocol (SNMP). We first reviewed the context of Network Centric Warfare provided by the Global Information Grid and FORCEnet concepts. These concepts, along with the O-I-T multi-level model inform the direction we have taken with that architecture, attempting to address the information exchange problem at multiple levels in a service-oriented fashion.

After discussing the technologies which underlie our architecture, namely SNMP and ASN.1, we developed three classes of Hypernodes which we propose to address the current deficiencies of information exchange. These three classes were further explored with case study investigation of the TNT-MIO experiment. From this analysis, candidate MIBs were developed and explained as an initial step in the realization of this architecture.

In order to appropriately develop the theoretical background for a work of this size, however, it was necessary to sacrifice a number of the products originally planned for this paper. What remains, we contend, will provide not only a thorough treatment of the subject matter as it pertains to the TNT-MIO experimental domain, but will also serve as an invaluable starting point for significant future research. A number of the originally planned products are obvious candidates for future work.

A. WEB SERVICES

Throughout this paper, we have discussed SNMP as the primary means we wish to utilize to transfer data. This decision was made from a twofold desire to minimize the number of information formats in use on the network and to ensure that even disadvantaged nodes have the ability to participate with or as Hypernodes. One direction of future research should include the investigation of using Web Services in conjunction with or as a wrapper for the SNMP data.

At least one architecture is suggested wherein a number of SNMP Hypernodes also serve as Web Services gateways. These gateways would allow for translation from SNMP to, for instance, the WS-Management framework and back, allowing SNMP-enabled nodes to access information on non SNMP nodes and vice versa.

B. SNMP-SNMP GATEWAY

The subnetwork aware Hypernode offers a quantum leap in capability for management of large, distributed networks in a purely SNMP fashion. In the TNT network, alone, this could reduce by a factor of 2-3 the amount of network traffic consumed by management information on the fringes of the network where capacity is the smallest already. The tools currently in use by CENETIX do not, however, have an ability to directly monitor custom MIB variables. In our experience, this is a significant limitation on nearly all the toolsets available in the network management space.

An obvious extension of this work, then, would be the creation of an SNMP-SNMP gateway serving much the same function as the Web Services gateway above. In this manner, a normal SNMP request to a cached node would be intercepted by its Hypernode which would answer for it. Unfortunately, without such capability, non-Hypernode aware network management tools will not be able to take advantage of this advance.

C. COMPILED MIBS

The process for compiling the ASN.1 format MIBs into executable computer code turned out to be more complicated than originally expected, possibly equal in scope to an entire thesis itself. An obvious extension to this research would translate the ASN.1 MIBs into C/C++ code and compile them into a usable format.

D. USER AND MACHINE INTERFACES

Even with usable custom MIBs, the manual effort required to issue SNMP commands for the custom variables would overwhelm any positive effects of the Hypernodes themselves. A possible parallel activity to the MIB compilation would be development of user applications for interfacing with Hypernodes. These would, necessarily, both include user-centric and machine-centric applications. A user-centric application might allow a user to query for services using a web form and return nodes offering matching or similar services.

Machine-centric applications would also be required. As mentioned in Section V, a number of changes to MIB variables should result in traps being sent throughout the network. This requires a helper application to watch the change of one variable in order to send such traps automatically. Database connectors also must be developed such that SNMP requests can be answered from robust data sources.

E. TESTING WITHIN THE TNT-MIO DOMAIN

Once the prerequisite MIBs have been compiled and deployed into the TNT network, testing should occur to determine both the usability and efficacy of the MIBs. The extent to which applications have been developed for this use will likely determine the means and method of testing. Involvement of users early in the program may also be useful for informing the development of the user interfaces themselves and for capturing previously unseen requirements.

F. OTHER DOMAINS/OTHER MIBS

It should be clear that the methods here are not designed purely for the TNT-MIO domain. It is our hope, in fact, that this architecture is appropriately flexible and powerful to be used as the information infrastructure in a wide range of domains. Consequently, future research designed to expand the applicability of the Hypernode concept to other domains is welcomed and encouraged.

There are likely other MIBs or extensions to the proposed MIBs both within the TNT-MIO domain and within others that will need to be developed. Hopefully, the method proposed by this paper can also guide such work.

G. OTHER DECISION MODELS/DATA MINING

In early drafts of this paper, it was pointed out that the decision information captured in a decision support aware Hypernode may be useful as a data source for later study. The results of and processes used for these decisions could inform later decisions. As currently structured, the decision support aware Hypernode MIB appropriately supports a case-based decision-making methodology.

Further research into the specific applicability of this decision information to later users should be investigated from the point of view of multiple decision-making methodologies. With this information, the decision support aware MIB should be modified or extended to give this capability to users of this class of Hypernode.

H. OTHER CLASSES OF HYPERNODES?

This paper proposes three classes of Hypernodes, each filling a different requirement in the information space. These classes were chosen due to their applicability to the GIG and FORCEnet concepts. They were designed in order to maximize the number of levels in the O-I-T framework to which they made contributions. It is unlikely, however, that we have succeeded in exhausting the classes of Hypernodes available for exploration. It may even turn out that the Decision Support Aware Hypernode is, in fact, merely a special kind of service, collapsing the proposed three Hypernode types to only two. We welcome such criticism and invite future research along these lines.

LIST OF REFERENCES

Alberts, D., J. Garstka, Richard Hayes and David Signori. Understanding Information Age Warfare. Vienna, Va.: CCRP, 2001.

Gateau, J. "Designing Command and Control." 2006 International Command and Control Research and Technology Symposium: Coalition Command and Control in the Networked Era, Cambridge, England, US Department of Defense, Command and Control Research Program (CCRP), 2006

Glazer, Barney and Anselm Strauss. The Discovery of Grounded Theory: Strategies for Qualitative Research, Aldine de Gruyter, 1967.

Hayes-Roth, F.. "Model-based Communication Networks and VIRT: Filtering Information by Value to Improve Collaborative Decision-Making". 10th International Command and Control Research and Technology Symposium: The Future of C2, McLean, VA, US Department of Defense, Command and Control Research Program (CCRP), 2005.

Hayes-Roth, F. "Two Theories of Process Design for Information Superiority: Smart Pull vs. Smart Push." 2006 Command and Control Research and Technology Symposium: The State of the Art and the State of the Practice, San Diego, CA, US Department of Defense, Command and Control Research Program (CCRP), 2006..

Huber, George. "A Theory of the Effects of Advanced Information Technologies on Organizational Design, Intelligence and Decision Making." The Academy of Management Review, Vol. 15, No. 1 (Jan 1990), pp. 47-71.

Joint Chiefs of Staff. Joint Doctrine for Military Operations Other than War. Joint Publication 3-07. 1995

"NetOps 100 Student Guide v1.2.2," Presented by JTF-GNO to The Naval Network and Space Operations Command, Dahlgren, VA, Jan 2005.

Network Working Group Request for Comments: 1157, J. Case, M. Fedor, M. Schoffstall, J. Davin May 1990

Network Working Group K. McCloghrie and M. Rose (eds) Request for Comments: 1213, Mar 1991

Wolfowitz, Paul. Department of Defense Instruction 8100.1 19 Sep 2002. Global Information Grid (GIG) Overarching Policy.

Yin, Robert. Case Study Research: Design and Methods, 3rd ed., Sage Publications, 2003.