# Toward Intelligent, Adaptive Message Routing in the Global Information Grid

*Steve Rowe*
*Douglas Haanpaa*
*Charles J. Cohen, Ph.D.*
Cybernet Systems Corporation
727 Airport Blvd.
Ann Arbor, MI 48108
734-668-2567
srowe@cybernet.com

Keywords:
Global Information Grid, message routing, artificial intelligence

**ABSTRACT**: *The Global Information Grid concept enables commanders to have access to information about every warfighter, vehicle, and weapon system in a particular theater of action. If not properly managed, this amount of information is overwhelming. On one hand, the client may be deluged with unwanted information, and on the other hand a critical system may be under so much electronic scrutiny that is crippled by merely trying to answer information requests. One way to manage the information is to make sure that redundant, irrelevant, and otherwise superfluous information is not propagated. Another is to distribute the server load for hot spots in the network. These techniques can be effectively applied within the routing nodes of the network. For example, Cybernet already uses modular expert systems to limit unnecessary traffic in their HLA implementation.*

*The fluid nature of the battlespace network will require the application of dynamic, adaptive, reactive computing technologies such as Belief Networks, Intelligent Agents, and Learning Systems to the information routing problem. Among the benefits of doing so are the ability for the network to self-heal when a router is lost, reduce bandwidth usage, and provide a higher quality of information to those who need it. There are also substantial technical challenges, including interoperability, security concerns, software deployment logistics, and the potential for vital information loss.*

*This paper explores the benefits and the consequences of applying these technologies to the GIG information routing problem, highlighting promising approaches and proposing solutions to potential problems. We conclude with a list of recommendations.*

## 1 Introduction - The vision of the Global Information Grid

The DoD directive 8100.1 outlines the vision of the Global Information Grid (GIG). There are startling ramifications of this directive, as implied by these excerpts:

> *The GIG shall support all DoD missions with information technology, for national security systems, joint operations, joint task force (JTF), and/or combined-task force commands…"[1]*
>
> *GIG assets shall be interoperable, in accordance with approved requirements documents, and compliant with the operational, system, and technical views of the GIG architecture [2]*
>
> *An enterprise-wide inventory of GIG assets shall be established and maintained.[3]*

To further clarify the meaning of "all GIG assets", one can refer to the Joint Vision 2010 publication of the Joint Chiefs of Staff, which says:

> *The fusion of all-source intelligence with the fluid integration of sensors, platforms, command organizations, and logistic support*

*center will allow a greater number of operational tasks to be performed faster. [4]*

In summary, the vision of the GIG is to have a universal, interoperable, indexed network of all information assets. This presents the implementers with a large number of technical challenges, a few of which are addressed in the next section.

## 2 Some potential problems

In order to limit the scope of this paper, we will assume that the GIG will be implemented in such as way that each node has a unique address (this can be accomplished using IPv6 [5]), that the GIG network is architected in a way similar to the current Internet (i.e., uses routers, domain name services, TCP/IP, etc.), and that each information source has some software for processing information transactions. We also assume an idealized network wherein there are no packet errors; that is, if a request for data is made, it will be honored. We will further limit our discussion to traffic flow within this idealized network.

Even given these assumptions, there are potential problems that must be addressed by the implementers of the GIG, and the application software that it serves:

- *Too much information for the client*. In this case, we refer to the user of some client software, who is presented with so much information that it is impossible for him to discern the facts of interest.
- *Too much traffic on the network*. So much information is being generated that the network infrastructure is saturated such that data delivery is no longer timely.
- *Too much strain on information sources*. Certain nodes in the network may become focal points of interest. These nodes may receive so many information requests that they cannot process them all and still perform their normal function.
- *Inaccurate data.* Data may be considered inaccurate by some information client if it is too old, too new, or misreported. Data may also be *suspect*, for example if it is unconfirmed or in conflict with other data.

The next section discusses possible solutions to mitigate these problems.

## 3 Applying technology to mitigate the problems

### 3.1 Problem: Too much information for the client (user)

This is not a network problem *per se*; it indicates that the network is functioning well, and that all available information is available to the client application. It is instead a problem of presenting the needed information (and no more), as efficiently as possible.

#### *3.1.1 Information Culling*

The first problem is how to decide which information is needed and which is not. Some first approximations can be made based on the purpose of the application (e.g., is it a force deployment application, or a logistics route planning application?). Next, the role of the user can be used to decide what is relevant (e.g., weapon line-of-sight is relevant for a force commander, but probably not to a quartermaster). These two steps are relatively simple ways to discern the *focus of attention*, which is the key to providing exactly the information required. Using focus of attention as the basis of information presentation has the dual benefits of limiting the amount of data that needs to be requested by the client, as well as limiting the amount of processing that needs to be done by the client to present the information.

Other methods of discerning focus of attention can be used to further narrow the amount of information that needs to be fetched and presented. Some techniques require active participation by the user of the system; for example, a user might select some information of interest from a menu. Other techniques are *user-passive*; they do not require any conscious decision or action by the user, and thus do not impose a burden on his cognitive workload. User-passive discernment of focus of intention is thus very important in applications where the user is already near their cognitive workload limit, such as a force commander directing battle operations. Some user-passive techniques that we have employed are eye tracking, situation recognition, and learning systems.

Since the cognitive workload of the user plays a key role in determining the appropriate level of detail to be displayed, reliable methods for determining cognitive workload must be introduced. There are four broad categories of workload assessment techniques that are used to measure cognitive workload: primary-task measures, secondary-task measures of spare capacity, physiological measures, and subjective rating techniques. Physiological measures require somewhat intrusive sensors, while subjective measures are already taken into account if the user is allowed to adjust the level of detail himself, based on how busy he feels he is. However, intelligent client-side software can be used to perform primary and secondary task measurements to automatically adjust the level of detail. Primary task measures entail an examination of the performance on the systems of interest. The idea behind adding a secondary-task to the primary task is that the secondary one is assumed to be inversely proportional to the cognitive needs of the primary one. Therefore, these secondary tasks will reflect the difference in task resource demand that is not reflected in primary-task performance. While secondary-tasks interfere with primary-tasks performance, it is possible to measure the manifestations of operator workload unobtrusively.

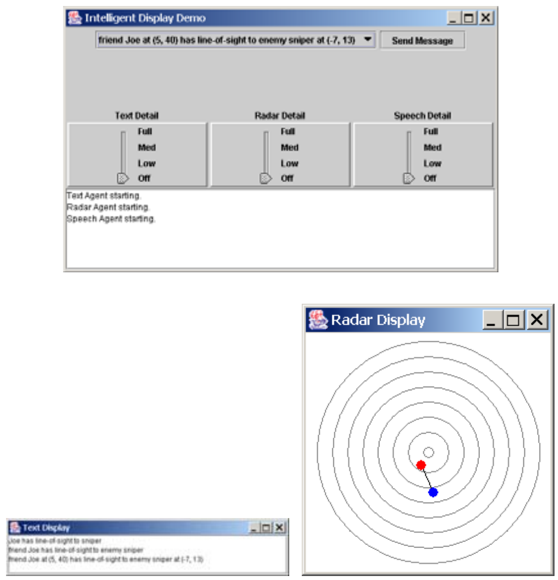


**Figure 1: Intelligent Agent Media Selection Demo**

### *3.1.2 Information Presentation*

Once the correct amount of information has been retrieved, the problem becomes how to most efficiently present that relevant data. We have investigated three approaches for maximizing user understanding.

The first of these approaches is to use multimedia to parallelize presentation. The premise here is that certain media types are good for relaying certain types of information. For example, a map is good at showing geographic location. A bulleted text list is good at displaying inventory. Some information can be shown simultaneously using multiple media types (e.g. the location can be shown on a map, and as textual latitude-longitude values). Cybernet has implemented a prototype intelligent agent-based approach that chooses appropriate interfaces for various informational elements.

The demonstration used a plain-text display, a graphical "radar" display, and computer-generated voice as three example media. By default, all display devices are inactive. The user has three detail sliders to change the level of detail for each of the presentation media. The intelligent agent uses the detail value in combination with the incoming data type and the locally available display capability to choose what user interface (UI) elements to bring up. This concept and the supporting software infrastructure can be easily extended to include modes such as maps, audio players, video players, and 3D renderers.

The second approach is to use a hierarchical or other map-based presentation scheme (map-based here refers to information mapping as opposed to geographic mapping). Similar in some sense to the commonly used cascading menu interface paradigm, this approach extends the idea by extending it into multiple dimensions and generalizing the representation. One possibility is to adopt the so-called *topic map* standard. Topic maps are actually a new ISO standard for describing knowledge structures and associating them with information resources. The topic map standard (ISO 13250) provides a standard and basis for developing technology geared towards knowledge management. This technology allows users to quickly drill down from a gods-eye view of the information space, to the specific information that they are after. This is a very powerful tool especially when the data being accessed is formally categorized, as is often the case with government, corporate, or educational institutions.

The third potential approach is to develop an infrastructure for defining, predicting, and setting per-user preferences. Users should be able to easily specify the information they want. But doing so with current interfaces is typically an explicit process that often increases their cognitive workload. Customizable interfaces just add and remove exactly what the user specifies. What is needed is a system that evaluates a user's preferences (what they want to see and what they don't care about) and intelligently chooses the level of detail for each informational element. Such a system must be able to sacrifice detail for increased clarity and vice versa, and this can only be done if the system has something upon which to base these interface selections. This is another place where the "intelligence" in intelligent agents is valuable. At the outset, such a system would require that users customize their interfaces the old-fashioned manual way. As the system was used, however, an agent-based learning systems would accumulate information from the hundreds or thousands of users that customize their UI layouts. It could incorporate and correlate that UI layout information with the

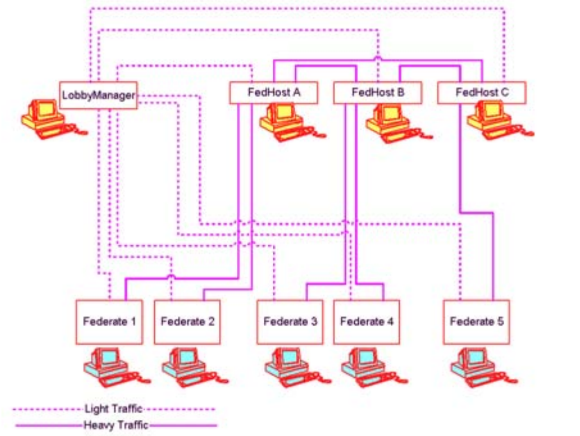


**Figure 2. OpenSkies Network**

subsequent performance of the corresponding users to get a sense of what works best. The system would then anticipate user needs, providing template layouts and suggesting modifications as the system gets smarter.

## 3.2 Too much traffic on the network

### *3.2.1 Intelligent Data Routing*

In order to create a system that provides all the right information to the right people over a potentially enormous network space, the propagation of data must be managed intelligently. In order for such an architecture to scale to large numbers of participants, we must restrict the data flow to only where it is needed.

The OpenSkies™ implementation of its Massive Multiplayer Online Gaming (MMPOG) developer kit, based on the High-Level Architecture (HLA) specification, is Cybernet's attempt to achieve this goal. An RTI implementation employing peer-to-peer communication between N peers will generate network traffic that is proportional to $N^2$. This makes federate scale-up possible only on local area networks and is limited even in this environment.

Cybernet's OpenSkies™ RTI is an HLA implementation that utilizes a network of servers called FedHosts to mediate network message traffic from the game or simulation clients. Each FedHost routes traffic for a subset of the Federates participating in the Federation. Initial contact for each Federate is to a LobbyManager, which immediately assigns the Federate to a FedHost, load balancing the FedHosts in the process. By itself, this does not eliminate the $N^2$ traffic problem. For this, OpenSkies™ implements a concept called culling rules, which are applets inserted into the FedHost that are allowed to decode message packets. Based on simulated entity state and packet contents, these culling rules can control messaging quality of service from each client to other clients. One possible culling implementation is an algorithm that culls based on relative distance. As an example, when objects are far away from each other in conceptual simulation space, data is transferred between them more slowly, or with less quality (or not at all) as compared to objects that are close and must interact more quickly. A subset of this capability is the *Data Distribution Management* mechanism that is a standard part of HLA. OpenSkies™, however, is not limited to this specific culling mechanism. Developers are able, using OpenSkies™, to implement their own culling module, which can take advantage of the attributes of a specific application and its requirements, such as this one. An example in the context of this effort would be to restrict data flow using the media agent intelligence described in the previous section. That way, the decision about what interface component to show a particular user will occur within the networking infrastructure. As a result, the streamed that would feed these interfaces (e.g. streamed audio) will only be sent if the corresponding media display is present and visible/audible within that users interface.

### *3.2.2 Push and Pull interface*

A networking API standard like HLA utilizes a multicasting-based approach, where clients (federates) send information about local entities. Interested parties will then get this information. While this is great for simulation networking it does not apply well to, for example, document retrieval or database lookup. It is important to realize that both mechanisms must be supported.

### *3.2.3 Publish-Subscribe*

Clearly the information content of the various data sources within the GIG must be somehow indexed and made available to the masses. For both push and pull-type of information, clients in the network must be able to tell the system what information they are publishing. I.e. there must be *yellow pages* for the aggregate GIG content. For push types of information, clients must be able to inform the GIG what information in which they are interested. In HLA-speak, this is called subscribing and it ensures that subscribers will receive the information they want if it is being broadcasted by anybody.

### 3.2.4 *Intelligent Queries*

The perfect counter example that demonstrates this point is a web search. Bring up Google and search for very specific information on any topic and you'll likely get back a large amount of hits that only remotely relate to what you are after. A better solution would be to formulate the queries in such a way that all responses are very relevant. While this may be difficult, or even impossible on the public internet, the design of the GIG must take into consideration how queries will be performed. The architecture and interconnectivity of the GIG must be designed with a query language in mind, and vice versa. One such language from which a great deal can be learned is SQL. SQL is an ANSI standard computer language for accessing and manipulating databases. SQL can be used to access, define, and manipulate data in a host of database packages from various vendors. Lessons learned from SQL could and should be applied to this effort.

## 3.3 Too much strain on information sources

This problem relates to the network traffic concern, but is different enough that different solutions apply. Examples of this are everywhere on the public internet, where access to popular websites is notoriously slow or sites that are not accessible during peak hours. On the GIG, such behavior cannot be tolerated. If a tactician needs to see satellite imagery of a battle space, then he/she had better get it. There are a number of approaches that have been applied to this problem. These must be formalized within the GIG. A significant one, ***caching***, is described here:

### 3.3.1 *Cacheing Nodes*

The idea behind caching is that an originator of data can delegate the network dissemination of its data to other hosts. These hosts can contain the same data as the originator, thereby mirroring the originator's information. Clearly, the mirrors must be sent the data to reflect, but the premise that makes caching workable, and that is normally true, is that the original data is updated much less than it is accessed from the outside. As long as the originator can update the mirrors fast enough, the mirrors can service the world. This concept can easily be extended to multiple levels, where there are cache/mirrors for the mirrors.

## 3.4 Inaccurate data

In the context of military operations, inaccurate data is worse than no data. The absence of data can alert command personnel to the need for further intelligence gathering. The presence of inaccurate data may lead to the commander making a bad decision, which can lead to loss of life.

Inaccurate data may result from data mistranslation (e.g., big-endian versus little-endian), network bit-errors, intelligence errors (e.g., a forward spotter mi-identifies a unit), sensor hardware failure, and a host of other reasons. While preventing such errors is the focus of system developers and procedure writers, it is unlikely that we will ever have a system that is completely free of erroneous data. Thus, nodes in the GIG must also focus on *detecting* inaccurate data. Once again, intelligent software techniques can applied to corroborate information. With the universal availability of data, it will often be possible to find supporting (or detracting) evidence for the validity of an particular datum. Belief networks, such as Dempster-Shafer and Bayesian, are a natural choice for technology to implement this type of corroboration system.

A more mundane, but still useful technique, is to understand the time-criticality of information requests so that information can be disqualified if it lies outside a certain temporal window. In the highly dynamic environment of the battlespace, information that is 10 minutes old may be too old to be useful. Therefore, it seems likely that every piece of information in the GIG will need to have some kind of time stamp so that their temporal relevance can be measured.

# 4 Recommendations

## 4.1 Look beyond HLA

When the simulation community thinks about interoperability, it begins with its own standards: DIS and HLA. When looking forward to the GIG, it is vital that we broaden our scope to interoperability with existing commercial standards such as J2EE, existing military standards such as Link 16 and MIL STD 6017, and emerging standards such as the System of Systems Common Operating Environment (SoS-COE) API implementations that will make up the core of the Future Combat Systems (FCS) program. This implies that technologies that promote interoperability, such as the OpenSkies gateway, will be of increasing importance.

### 4.2 Start creating modular mitigating technologies now

The implied capabilities of the GIG and JV2010 place a large burden on implementers and practitioners of military information technology. There are some big potential problems with the very concept of universal interconnectedness of military assets, a very few of which were addressed in this paper. Other concerns include security, need-to-know, prioritization of requests, quality of service, service indexing and lookup, and many others. If we are to reach the promise of the GIG, we must begin immediately to implement and test novel solutions to the potential problems. The application of modern AI in the existing systems that will become the foundation of tomorrows GIG nodes is a promising first step.

## 8. References


[1] Department of Defense Directive 8100.1, September 2002. Section 4.1.

[2] Ibid, Section 4.3.

[3] Ibid, Section 4.9.

[4] Joint Vision 2010, The Office of the United States Joint Chiefs of Staff, p 13.

[5] RFC 2460 - Internet Protocol, Version 6 (IPv6) Specification. http://www.faqs.org/rfcs/rfc2460.html


## Author Biographies

**CHARLES COHEN** has been working in the fields of image processing, robotics, human-computer interaction, and artificial intelligence for over a decade. He is currently the Vice President of Research and Development for Cybernet Systems Corporation. He has been the project manager for many projects for the United States Armed Forces (Air Force, Navy, and Army), National Aeronautics and Space Administration, and other government agencies. Dr. Cohen’s current research interests are in gesture recognition, image processing, estimation theory, system integration, visual communications, machine vision, and perceptually coupled systems.

**DOUGLAS HAANPAA** is currently the technical team manager for Cybernet’s Virtual Reality group. He is responsible for the management of OpenSkies. He contributed much of the design of this product as well as implementation of flight dynamics, collision, and force-feedback algorithms. He also acted as a key designer for many of the OpenSkies subsystems including the terrain parsing/rendering/LOD system, scenegraph, weather model, and parallel thread/timer system.

**STEVE ROWE** is a senior software engineer at Cybernet Systems Corporation. Besides being well versed in simulation environment, intelligent agents, and scenario development work, he has taught in the classroom and also worked in the development of automated teaching materials at KnowledgeNet Corporation.