From the Desktop to the World: An Introduction to Developing and Hosting a High-Availability ArcIMS® Application

An ESRI ® Technical Paper • January 2004

ESRI 380 New York St., Redlands, CA 92373-8100, USA • TEL 909-793-2853 • FAX 909-793-5953 • E-MAIL info@esri.com • WEB www.esri.com

J-####

From the Desktop to the World: An Introduction to Developing and Hosting a High-Availability ArcIMS Application

An ESRI Technical Paper

Contents

Introduction 2
Getting started 3
   Take inventory 3
Application design 4
   Data 5
   Functionality 5
   Audience 6
   Availability requirements 7
Application development 12
Sizing considerations 14
Performance considerations 18
   ArcIMS configuration 18
   ArcIMS service tuning 23
   Data storage 26
Staging 28
Production and maintenance 29
Conclusion 30
Support 30

From the Desktop to the World: An Introduction to Developing and Hosting a High-Availability ArcIMS Application

ArcIMS® gives GIS professionals the ability to distribute geographic data and mapping services to a wide audience via the Internet. The world is becoming more dependent on the rapid, reliable exchange of information through the Internet, a trend that is undeniably here to stay. ArcIMS allows GIS administrators to create Internet GIS applications that will meet their respective data-sharing goals.

Introduction

The purpose of this paper is to outline the most important considerations for hosting a high-performance ArcIMS application that is reliable and consistently available to all potential users. The discussion will cover application design, development, staging, and production, as well as sizing and performance considerations. Readers should be familiar with developing and storing GIS data, and should have a basic knowledge of ArcIMS, networks, and the Internet.

This paper is structured around the following definition:

A high-availability ArcIMS application is one that is capable of serving every anticipated incoming request successfully and in a timely manner.

In other words, the goal to keep in mind when building your ArcIMS application is that no user should get a blank page or an error when they try to access the Web site and browse the map service; nor should they have to wait an unreasonable amount of time for any page to load.

The key factors to achieving this goal are:

- Developing clear, detailed project objectives and using those as guidelines for design
- Awareness of the highest potential user demand
- Efficient use of the resources that support the application
- Proper data preparation and storage
- Proper configuration of all supporting hardware and software for maximum availability and performance
- Effective communication between all staff involved in the creation and preparation of the application, data, network, hardware, and software

Because the scope of this paper tends to be broad, specific instructions cannot always be provided; however, several links to more detailed information have been supplied for those who wish to investigate a particular topic more thoroughly.

Getting started

When implementing a high-availability ArcIMS Web site, it is helpful to keep the following philosophy in mind: You are not designing a GIS application that will be available on the Internet; rather, you are designing a high-availability Internet application that happens to contain GIS content. Although the distinction may seem subtle, the focus on the Internet application is what will determine whether your GIS Web site is a success or a failure. Even the most spectacular GIS content is worthless if it cannot be accessed.

Perhaps you have no idea what is involved in developing and hosting a busy Web site. That's OK—this paper is designed to demystify the process. While not a step-by-step guide to implementing an ArcIMS application, it should enable you to approach the project with confidence. Knowing the right questions to ask is half the battle.

Take inventory

Before jumping right into design, it is recommended that you first assess the resources that are available to you.

- *What kind of Internet connection do you have?*
  It's important to know what kind of Internet connection is available for your ArcIMS application. Examples of different connection types are given in Figure 4 on page 17. If deemed necessary, would an upgrade be possible?

- *What hardware resources do you have access to?*
  Find out how much machinery you will be able to devote to the project. Identify and note the make, model, processor speed, RAM, and operating system of all available machines. This is critical information that is necessary to determine if the hardware meets your requirements.

- *Does your organization already host a Web site?*
  If so, try to gather information about the lessons already learned by your IT department about designing and maintaining this site. You may find a wealth of Web application talent right under your own roof. Form a good working relationship with those involved, and reach an agreement to secure support for your new ArcIMS application.

  If you don't have the advantage of a functioning Web site already produced by your organization, don't be discouraged. Of course this means more responsibility for you in the design and implementation of your application, but you can still help speed up the process by taking inventory of what you have available.

- *How are you currently handling availability requirements?*
  Learn the systems administrator's strategy for handling the availability requirements for your organization's current Web site if there is one. Gather information about the number of page views that the site receives, as well as information about peak access times. If your production Web site accesses a relational database, there should be an availability plan in place for that as well.

If your organization already employs an availability strategy that works, go ahead and use it—just adapt the strategy to your ArcIMS application.

- *How is the current Web site hosted?*
  Chances are, an existing external site will be hosted within a perimeter network with some degree of security implemented. Spend some time finding out if the site administrator has anything to say about hosting your ArcIMS site. Find out if there are any set strategies in place that you can use when you're ready to roll your ArcIMS application into production. Gather information about security strategies for external Web sites. Maybe your organization has a policy for exclusively using one type of Web server software. Find out as much as you can.

- *Who can help you?*
  Take the time to establish good working relationships with those who can help you on the road to success: Web administrators, GIS professionals, database administrators, system administrators, and others. If you already have resources available to you, it is imperative that you communicate and collaborate with the appropriate people to bring these resources to your disposal. Do you have the staff to develop, implement, and maintain your site? Try to identify the strengths of your current organization and the areas that might need some additional staffing or resources.

- *What shape is your GIS data in?*
  Do you already have the data you want to display on your site, or do you need to obtain it? Is it easily obtainable? How and when will it be updated or maintained? Does the data include metadata? Data is a key component; consider it at the beginning.

The purpose of this initial evaluation is to identify your starting resources and determine what else you need for your project.

Application design

One of the classic colloquialisms in the world of information technology is "If you don't know where you are heading, you can't expect to get there." In other words, it's important to determine your objectives and devise an appropriate plan for meeting them.

The design phase is the most important aspect of Internet application development. Producing a fast, highly available Web site requires considerable resources from a variety of areas, and good planning will help ensure that resources are properly identified and efficiently used. It takes time and effort to design an Internet application well, but good design can save time, money, and frustration over the life of the project.

Many times it's difficult to plan everything down to the last detail. The audience may be relatively unknown going into production, or the functional requirements of the site may change over time. This is to be expected to some extent. Small changes, such as an improvement to the aesthetic experience of the Zoom In tool or the fields to be returned on an identify request, will have a minimal impact on the overall expense and complexity of the project. Large changes though, such as an eleventh-hour decision to incorporate the county aerial photography database with 2 terabytes of imagery into the application, can have a huge overall impact.

In the four sections that follow, we will look at the most critical areas to consider when designing an ArcIMS application that is capable of serving every anticipated incoming request successfully and in a timely manner: data, functionality, audience, and availability requirements.

Data

The data used within an ArcIMS application represents the *purpose* of your Web site. Without GIS data, there can be no Internet GIS application. It is the data that drives demand for your application.

It's important to have a good understanding of the data that you want to publish on your site. The size, type, and detail of the data will need to be identified, along with the projection and coordinate system you want to use. Keep in mind that large amounts of data, the use of imagery, or the use of metadata can significantly affect the design and support requirements of the application.

Data decisions should be made as early as possible in the design phase. Large-scale changes in the source data requirements can result in problems that will ripple through the entire project and directly affect the final overall cost and time frame for your application. Carefully think about the objectives of the Web site, and decide what data is essential in order to fulfill those objectives.

Not only is it important to gauge your end users' data needs, but also to prepare the data for high availability. This means paying careful attention to data storage issues—you should consider generalizing your data if possible, or increasing raster cell size. You should also get rid of any features or attribute fields that are unnecessary. These steps will result in faster load times for your maps. Finally, you should determine whether you plan to use ArcSDE® to manage your data, an option with significant benefits over using file-based data. The use of ArcSDE will provide a GIS database administrator with many powerful tools that can help prepare a GIS database for rapid access and retrieval within an ArcIMS application. We will say more about data storage later, in the section on performance considerations.

Functionality

Once it is known what data will be served, the next point to address is the functionality that will be required within the application. ArcIMS offers many different GIS-specific functions to manipulate and present the data that you wish to share. This includes the option to zoom in and out, identify feature attributes, query the data, perform analysis, or anything else your users need to do. (A complete discussion of all the possibilities for client interaction with an ArcIMS Web site is beyond the scope of this paper.) The functionality provided by your Web site is a major contributing factor to user experience. Your data might be nice to look at, but if a user cannot interact with it, it's not very helpful.

You should also think about how frequently users will want to access the tools provided by your site. One of the main reasons for establishing functionality guidelines at this point is to begin estimating the number of transactions that can be expected for a single user session. ArcIMS is fundamentally a transaction server. A user sends a request to an ArcIMS service and then gets a response. Every time a new map is requested, a transaction is initialized. Every time a feature is identified, a transaction is initialized. Zooming in and out, panning, and zooming to full extent all result in a request for new service information—a transaction. The exact number and types of transactions can be difficult to gauge going into a project, but an approximation can be made based on the tools that are available to the users. In short, each transaction can be considered a hit to the ArcIMS server. The number of hits will not be the same as the number of page views—one page view can have many hits. For instance, initial access to an ArcIMS application will result in a recorded page view for the Web server software, but the user might send many requests or hits to the ArcIMS server as they pan and zoom and otherwise interact with the map.

More sophisticated functionality such as adding layers on the fly, geocoding, routing, and layer extraction can not only mean more transactions, but more importantly, those transactions are typically more resource-intensive. There will always be a trade-off between functionality and performance in any Internet application. So, if it's not mandatory functionality according to your project objectives, it's best to do without it.

If you want to give users the ability to browse Metadata Services as a gateway to other ArcIMS services or for downloading purposes, keep in mind this will result in additional application overhead. The overhead involved in managing, storing, and serving a large metadata database is significantly greater than that for a traditional ArcIMS service application. Support for XML style sheets must be written into the client application, and additional ArcIMS and ArcSDE server resources will be required to support the Metadata Service. Consult the ArcIMS Metadata Server documentation[1] or the ESRI Support Center (http://support.esri.com) for specific details.

It is understood that there is significant demand for the ArcIMS Metadata Server within organizations that intend to share large amounts of data on their own or through the National Spatial Data Infrastructure. Metadata is vital for identifying useful datasets within a GIS data clearinghouse.[2] This is a perfectly logical requirement for an ArcIMS application, and the functionality is built into the ArcIMS architecture. However, it is important to keep the added overhead inherent with the Metadata Server in mind as you consider your strategy for a high-availability application.

Audience

Once data and functionality requirements have been considered, you can start making some educated estimates about the amount of traffic that can be expected on your site. The number to be concerned about is not page views, but rather the number of transactions, or hits. As mentioned above, each request for information counts as a transaction.

There are many strategies for estimating the number of requests that will come through a particular Web application. Most will attempt to estimate the traffic on a site in terms of peak transactions per day, hour, or minute—in other words, the highest number of transactions that can be expected during a given time frame.

Web sites are almost never viewed on a consistent basis throughout their life cycles. Web site administrators do not have the luxury of a steady, consistent flow of requests over a period of days, hours, or minutes. In practice, the flow of requests will be unpredictable. The application could go for several hours without a single request, and then receive 10,000 requests within 15 minutes. Logically, it is this peak period that you should

[1] *Creating and Using Metadata Services* installs as a documentation option with ArcIMS 4.0 and higher. Be sure to consult this document for additional information about the ArcIMS Metadata Server.

[2] The ESRI Geography Network^{SM} and the National Geospatial One-Stop are good examples of large spatial data portals designed and built with the ArcIMS Metadata Server extension.

prepare for.[1] If you can handle the load during this time, you will easily handle anything less.

In addition to peak requirement calculations, other estimates of requests over time may be formulated to help size the application and plan for maintenance or other scheduled downtime when the application is relatively unused. Statistical calculations, such as a Poisson distribution, can be used to estimate the flow of requests over time with sizable accuracy.[2]

As an example, let's say that a site is being designed to inform the public about wild fires on public lands. Under normal conditions, this site would only be accessed by those interested in the damages caused by occasional fire events. Load on the application can remain relatively light for long periods of time. If, however, there are major fires on public land that threaten large metropolitan areas, your audience has just increased significantly. Residents of those densely populated areas and their relatives are going to want reliable information about the fires, and suddenly your site could be inundated with requests. It is this most extreme scenario that must be considered and planned for.

An understanding of your audience is crucial for determining your availability requirements, which should be the primary influence in the development of a high-availability ArcIMS application.

For the purposes of this paper, we will assume a peak estimate of 100,000 hits per hour. This value represents the busiest hour that we can anticipate having to support within our application.

Availability requirements

In the introduction, a basic definition of a high-availability ArcIMS application was proposed: one that is capable of serving every anticipated incoming request successfully and in a timely manner.

It is a vague definition by design. ArcIMS project leaders should come up with a more detailed definition that is tailored to their specific availability requirements.

One of the universal elements of a detailed high-availability definition is a reference to a set of redundant systems that takes over when the primary system becomes unavailable for some reason. By allowing one or more parts of a configuration to be offline, we establish a backup plan, ensuring that our users will be able to access the application when they need it. *High availability equals reliability.*

Taking a step back from the world of information technology, it's easy to find applications of high availability in many industries. For example, architects and engineers incorporate high availability in almost every project. Bridges and airplanes, for example,

[1] This is especially true if the application will be hosting mission-critical data. If the data is somewhat less important, then this guideline does not carry as much weight.

[2] ESRI uses such calculations for the Geography Network to estimate traffic and schedule downtime and maintenance of portions of the network. These statistical calculations can deliver results that indicate that 80 percent of requests will arrive in 20 percent of the time, etc. More information about statistical Internet traffic calculation is available in print and on the Internet; try searching on "Poisson Internet statistics" using your preferred search engine.

are designed and built to meet not only a high standard of performance, but also reliability.

The real key to reliability is to have a plan in place for continued operation during scheduled (or unscheduled) downtime of the primary system. A four-lane bridge can be repaved two lanes at a time instead of all at once. The bridge is still operational but at a reduced capacity. Airplanes are always built with multiple engines, so that if one fails, the plane can still operate successfully.

The need for redundant systems will increase with the level of importance of the device or system. If a bridge has to be shut down because of high winds, commuters will have to find an alternative route. It's an inconvenience, but it's not too serious.

Airplanes, however, carry hundreds of passengers thousands of feet in the air. If an aircraft fails and crashes, it results in horrible consequences. Lives are lost, and public confidence in the reliability of air travel and in that particular airline is shaken. Such dire potential consequences demand extensive safety precautions.

Computer systems can likewise have different degrees of availability requirements. A Web site with only a few users that are accessing data of limited value can afford to have quite a bit of downtime as components fail, software is rebooted, and so on. However, a Web site that directly affects the income for a company must stay available as much as possible. Failures and updates must be accounted for at all times, and redundant components must be available instantly. The company could potentially lose customers and a good deal of money if the application goes down for any reason, so downtime must be minimized.

Most ArcIMS administrators will find that their availability requirements fall somewhere in the middle of this spectrum. The application will have to function reliably, but a minimum amount of downtime will be acceptable. Recognizing where your specific application will fall within this spectrum is perhaps the largest challenge of implementing a high-availability Web application.

Calculations of availability, downtime, and recovery for an Internet application can be extremely complex. It is not expected that an ArcIMS administrator would have expert knowledge of high-availability application and system design, but there are some key points to cover and some key terms to recognize.

First, a fundamental discussion of network availability is required. The information covered here will help lay the groundwork for estimating the availability requirements for an ArcIMS application. Networks—composed of computers, switches, hubs, and routers—form the skeleton of any ArcIMS application, so it makes sense to start with this general discussion and work toward the other components of the application. Once we have an understanding of network availability, we can address software, data storage, application design, and other considerations of overall availability requirements.

Network availability is predominantly stated in two ways: the percentage method and the defects per million (DPM) method. The percentage method is generally used to state the predicted availability of a network based on the overall network design and assumed availability of its components. The DPM method, on the other hand, is generally used to provide a metric of availability performance for existing networks. It is better suited to actual statistics as opposed to estimations.

As its name implies, the **percentage method** states the percentage of time that a network is predicted to be available. Specifically, the percentage will represent the amount of time within one year that the network is estimated to be up and running properly.

It is not uncommon to hear network availability expressed as “five nines” or something similar. This simply means that a network has been designed to provide service 99.999 percent of the critical operation time in an average year.

There is a good deal of flexibility involved in the expression of network availability as a number of nines. For instance, if a network has an hour of scheduled maintenance every night, then the scheduled downtime might not be factored into the calculation even though the network is unavailable for this time. The availability calculations will be based on the scheduled availability only, not the absolute availability.

The following chart illustrates the use of the percentage method to calculate downtime in minutes for one year. All calculations are based on a figure of 525,960 minutes per year, which accounts for leap years.

Number of 9s – Uptime and Downtime

Number of Nines	Availability Percentage	Minutes of Uptime per Year	Minutes of Downtime per Year	Annual Downtime
1	90.000%	473,364	52,596	36.5 days
2	99.000%	520,700.4	5259.6	3.5 days
3	99.900%	525,434.0	525.96	8.5 hours
4	99.990%	525,907.4	52.596	1 hour
5	99.999%	525,954.7	5.2596	5 minutes
6	99.9999%	525,959.5	0.52596	32 seconds

Figure 1—Quantitative analysis of uptime and downtime for percentage method calculations. Taken from *High Availability Network Fundamentals* by Chris Oggerino.

As you can see, the amount of downtime significantly drops with every additional nine. One of the more common sayings in networking circles is that after the second nine, each subsequent 9 costs twice as much to achieve! In other words, if you want your predicted availability to increase from three 9s to four 9s, the cost of the network will double. This might seem discouraging, but you have to keep in mind that for the extra 100 percent in cost, you have gained 1000 percent in availability.

The **defects per million method** can be useful to state the actual availability performance of a network. Simply put, this method measures the number of failures per million for a device, network, Web server, or other components of an application. This method can be better suited than the percentage method to track the availability performance within an existing network. Many different statistics—such as the number of hours that a network has been available, the hours of operation of all of the devices that comprise the network, and even the number of hours that users are accessing the network—can be calculated in DPM.[1]

Both of these methods can take into account the reliability ratings of your network equipment. Since this section primarily deals with the *planning* of a high-availability application configuration, the percentage method is going to be used for the sample calculations.

Most network and server devices, such as switches and routers, are advertised with some sort of a reliability rating. Most of the time this rating is measured in Mean Time Between Failures (MTBF) or Mean Time To Failure (MTTF). These values are often confused and muddled within the industry, but as will be shown, it really doesn't matter for the calculations demonstrated here. The important fact is that there is a way to roughly estimate when these devices are expected to fail and to plan a network accordingly.

By far the most important metric for planning the availability of a network is the Mean Time To Restore (MTTR). This approximates the average amount of time that it takes for the network to be restored to proper working order after a failure. In essence, this factor represents the downtime of the network, or the time that our application is unavailable.

Once we have some values for these numbers, we can plug them into the availability equation. This equation calculates a percentage of expected availability in the form of a decimal value (for example, .998943). You will have to multiply the availability value by 100 to get a percentage or "number of nines."

AVAILABILITY = MTBF / (MTBF + MTTR)

Alternatively:

AVAILABILITY = MTTF / (MTTF + MTTR)

For example, consider a system that can go an average of 10,000 minutes between failures, and when it fails, it takes five minutes to restore it. In that case,

AVAILABILITY= 10000 / (10000 + 5)

The equation results in a calculation of .999500, or 99.95 percent.

So it becomes clear that the confusion of the values for MTTF and MTBF really doesn't mean much for these calculations. It also becomes clear that the most important availability factor by far is going to be the time that it takes to get the network up and running again after a failure, or MTTR. This is obviously where most resources should be leveraged to increase availability ratings.

With this in mind, it can be concluded that a successful server configuration should be able to withstand and account for failure of any single node or maybe even a number of nodes depending on the availability specifications. If handling mission-critical information, then the network will have to keep the Web application available at all

[1] Additional information about the DPM method is available from many sources. Chris Oggerino's *High Availability Network Fundamentals* from Cisco Press and numerous Internet sources contain more details.

times. Less important data might be able to get by with a lower availability rating. At any rate, it is necessary to quantify the amount of time that it takes a redundant system component to step in and take over for a failed component. It is impossible to obtain useful availability calculations without an accurate measurement of MTTR.

The availability equation and discussion above has only introduced a very simple mathematical view of availability. These simple calculations can become quite a bit more complicated when all of the applicable downtime factors are calculated. Factors to be considered include not only switches, hubs, and routers, but also hardware, software, applications, Internet service providers, etc. All of these things must be factored into the availability calculations if the application is to meet the target availability requirements.

It's worth repeating—the prime factor for achieving high availability is the ability to recover from failure as quickly as possible. Of course, in order to do this effectively, it's necessary to identify all possible areas where a failure can occur and design redundant systems that can take over in the event of an unforeseen failure. Once this information is known, the simple availability equation can be applied to all of the anticipated areas of failure, and finally an overall availability rating can be calculated.

There are many contributors to downtime for an Internet application. Some of the most important are:

- Hardware
- Software
- Environmental considerations
- Human error
- Network design
- Network operations

A failure in any one of these areas can cause an application to go offline for a period of time.

Hardware failures are the easiest of the factors to deal with. Anyone who has contemplated implementing a high-availability network configuration has most likely accounted for hardware failure at some level. Many network devices have some metric of expected failure stamped right on the box. While these values are largely marketing tools, they are representative of the importance of hardware availability in any network availability calculation.

One often-overlooked factor of network availability is the availability of the software in use on the network that supports an application. What if firewall software crashes, and the external gateway router has to be rebooted? This could cause the loss of the Internet connection for the period of time that it takes the router to come back online. What if there is a problem with the operating system on one of the Web servers and it must be rebooted or repaired? There has to be another Web server that can reliably serve the application at such a time.

Environmental concerns could be anything from a major earthquake to a fire to a prolonged power outage. The ability to factor these unpredictable events into availability calculations can be difficult. However, it is not uncommon for mission-critical applications to be supported by server farms in multiple locations or to have backup generator systems to account for these environmental factors. Some detailed cost-benefit analysis is obviously required to justify such drastic availability measures. Even if it is decided that the cost of such measures is prohibitive, environmental factors should be considered in your availability calculations if possible. Perhaps MTTR metrics for your local power company, based on various environmental scenarios, can be obtained. This can help formulate a more accurate picture of potential downtime.

Human errors, such as tripping over a power cord or updating software on the wrong server, can also cause significant downtime. Network design and operation factors, such as bandwidth limitations, software updates, and equipment upgrades, can affect downtime dramatically if something goes wrong or mistakes are not accounted for. It is very important to formulate upgrade and maintenance plans that work within the availability requirements of the application.

It is the project manager's responsibility to account for all of the above factors when designing and implementing a high-availability application. A lack of planning for any factor that can bring down an application could itself be considered the cause of failure. For example, unanticipated downtime can occur *because there was no redundant system in line to serve as a backup*. It is important to formulate as complete a plan as possible, even if all downtime factors cannot be accounted for. At least the potential downtime can be identified and factored into predicted availability calculations.

The importance of accurately quantifying availability requirements should not be underestimated. If too much is invested in trying to achieve perfect availability for an application that does not require it, then there will be problems with rising costs and inefficient resource allocation. Likewise, if too little attention is paid to availability and a mission-critical application fails regularly, there will definitely be problems in the form of lost revenues or decreased production. Plan to spend some time developing and quantifying the specific availability requirements for your application. It is equally important to consider having a redundant system in place. ArcIMS configuration possibilities that incorporate redundant systems will be explored later, in the section on performance considerations.

Clearly, most organizations that are interested in implementing a highly available application of any kind should have at least one person who specializes in performing availability calculations and formulating contingency plans for every conceivable situation. The more critical the system, the more critical the planning will be.

This section should have provided ArcIMS and/or GIS administrators with the basic knowledge needed to work with such a person and to be closely involved with the design of a high-availability network to host an ArcIMS application. In addition, a foundation for availability requirements should now be in place that can be built upon as the application begins to take shape.

Application development

In general, the out-of-the-box viewers created by ArcIMS Designer will probably not be well suited to most high-availability requirements. The HTML, Java™ Standard, and Java Custom viewers are a showcase of the functionality available for an ArcIMS client and serve as a way for new users to get up and running quickly with an Internet GIS application.

High-availability applications should be written to support only the specific functionality requirements of the audience and should avoid the inclusion of extra code that does not serve the application's primary objectives. As an example, the ArcIMS HTML viewer includes the ability to geocode and extract layers from a service, among other things. If you do not plan to use this functionality within your application, there really is no reason to require that your users download the additional HTML and JavaScript code that facilitates it. This would add useless overhead to the application.

It will likely be in your best interest to develop a custom viewer for the target audience. ArcIMS supports a number of application development options through the various Application Server connectors. Normal HTML and JavaScript communication is handled through the servlet connector, but this is by no means the only option that developers have to design and implement custom ArcIMS applications.

The Java Connector, ActiveX® Connector, and ColdFusion® Connector allow developers to design GIS-centric Web sites with a wide array of Internet technology and also allow for ArcIMS content to be included in existing Web sites that have been developed with these technologies. For example, the Java Connector can be used within server-side JavaBeans™ or a custom servlet to handle all ArcIMS content and transactions within a Java/JSP application.[1]

Your functionality requirements are a factor in determining the best application development options for your project. If database access or updates are required, then you'll want to investigate the server-side solutions for ArcIMS that allows for ODBC, OLE, or JDBC database connections. If the need exists to handle complex client–server transactions through multiple servlets, the Java Connector might be used to develop a custom servlet specific to ArcIMS transactions. The real key here is to be informed of the functionality that each option offers and to weigh that against the needs of your application and the resources at your disposal.

To find out more about the different options available with the different connectors, take a look at the ArcIMS samples. Each of the Application Server connectors within ArcIMS comes with a suite of samples and templates that install as an option. The samples illustrate functionality ranging from the simplest tools for displaying attribute information to robust applications that incorporate a number of complex GIS tools and detailed user interaction. Alternatively, you can explore the ESRI Support Center to download additional viewer samples.

Above and beyond being an excellent showcase of the functionality of the different Application Server connectors, the samples can serve as a starting point for your own application. You might find it much more efficient to get started with one of the samples than with a blank project in your favorite Internet Development Environment.

No matter which option is chosen, performance and reliability of the application should always be on the minds of the developers. Remember the goal that was set forth in the introduction: an application that is capable of serving every anticipated incoming request successfully and in a timely manner.

[1] Consult the *Customizing ArcIMS* series of documentation—available from the ArcIMS media CD—for specific information about the individual Application Server connectors and their options.

The potential limitations of your audience should be considered, such as their level of GIS knowledge or the anticipated download speeds with which they will be accessing the application. For example, if an application user is limited to a 56K modem and a dial-up Internet connection, they will not be able to download a large HTML and JavaScript application very quickly. You might want to further evaluate server-side scripting options to minimize the amount of information that has to travel over such a slow connection. You will also want to keep the size of the output images to a minimum for ArcIMS Image Services to make sure that these users are not waiting a long time for a map image to refresh.

As you develop your application, keep in mind it is best to minimize the number of transactions between the client and the server. Simple choices such as the use of stored map and legend images, for example, can reduce the amount of requests that will result from a user accessing the site for the first time.

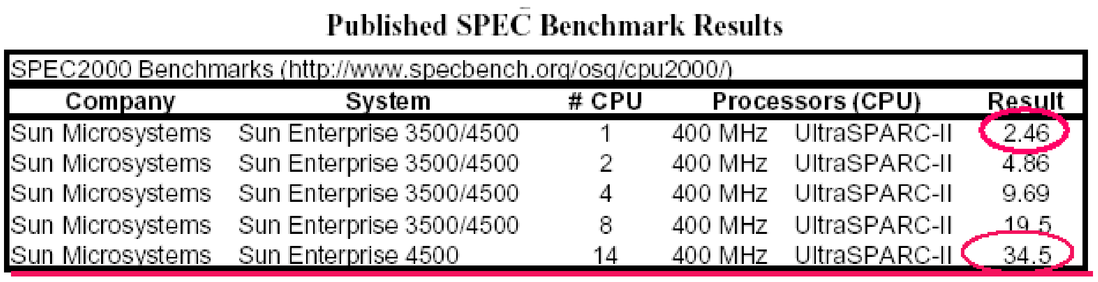
No matter what choices you make for your application, it is important that everyone on the team is involved and informed during its creation. The development staff will have to let the appropriate network administration department know if they will be utilizing custom servlets or complex database access so that these factors can be incorporated into the overall system design. Remember, communication is a key factor to a smooth rollout of the finished product.

Sizing considerations

The ESRI Systems Integration team has assembled a wealth of information about hardware sizing within the *System Design Strategies* white paper, which can be accessed from http://www.esri.com/systemsint/index.html. The charts and graphs in this paper can help a system administrator make an informed decision about the hardware resources needed to host a high-availability application.

Fortunately, GIS administrators are not alone in the quest to limit the costs of supporting a busy application. The Standard Performance Evaluation Corporation, or SPEC (http://www.spec.org), has taken much of the guesswork out of choosing server hardware by assigning a metric to server performance. The metrics reported by SPEC are based on benchmark testing tailored to specific areas of interest. These comprehensive tests measure specific machine traits—throughput for floating-point calculations, for example—and allow you to compare different machines on a numeric scale based on the results.

Server hardware differs from manufacturer to manufacturer, and it can be difficult to settle on the best choice when all you have to go by is the word of the salesperson. The information available from the SPEC test results can aid in evaluating the best-suited hardware among many candidates that might seem very similar on the surface. With an informed comparison, one machine might stand out from the rest or offer better performance at a more reasonable price.



Published SPEC Benchmark Results

SPEC2000 Benchmarks (http://www.specbench.org/osg/cpu2000/)

Company	System	# CPU	Processors (CPU)	Result
Sun Microsystems	Sun Enterprise 3500/4500	1	400 MHz UltraSPARC-II	2.46
Sun Microsystems	Sun Enterprise 3500/4500	2	400 MHz UltraSPARC-II	4.86
Sun Microsystems	Sun Enterprise 3500/4500	4	400 MHz UltraSPARC-II	9.69
Sun Microsystems	Sun Enterprise 3500/4500	8	400 MHz UltraSPARC-II	19.5
Sun Microsystems	Sun Enterprise 4500	14	400 MHz UltraSPARC-II	34.5

Figure 2—SPEC Performance Benchmarks assign a scaled metric of performance to different server machines by name. Taken from *System Design Strategies*.

As has been stressed throughout this paper, ArcIMS is a transaction server. The ArcIMS Spatial Server is optimized to serve spatial and attribute data on a request-by-request basis and to handle many such requests at once. The metrics of throughput available from SPEC are used to create ESRI system design charts such as the following:

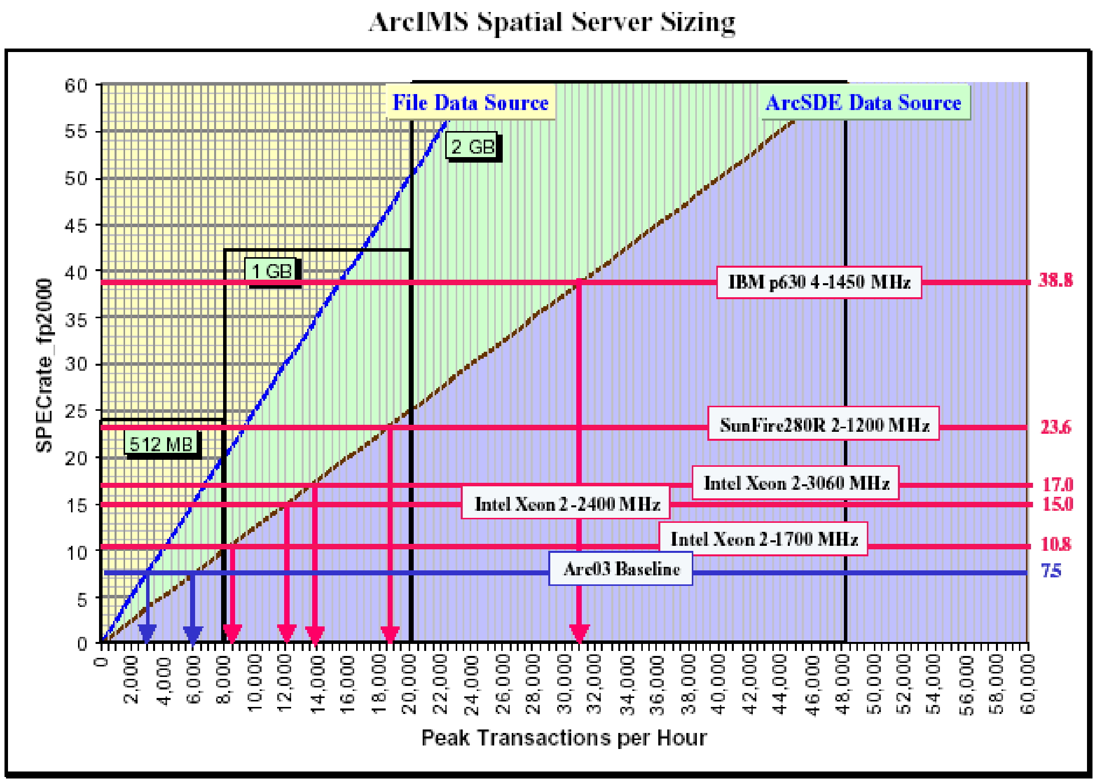


Figure 3—Illustration from *System Design Strategies* showing the relationship between SPEC performance metric and peak transactions per hour for ArcIMS Spatial Server hardware.

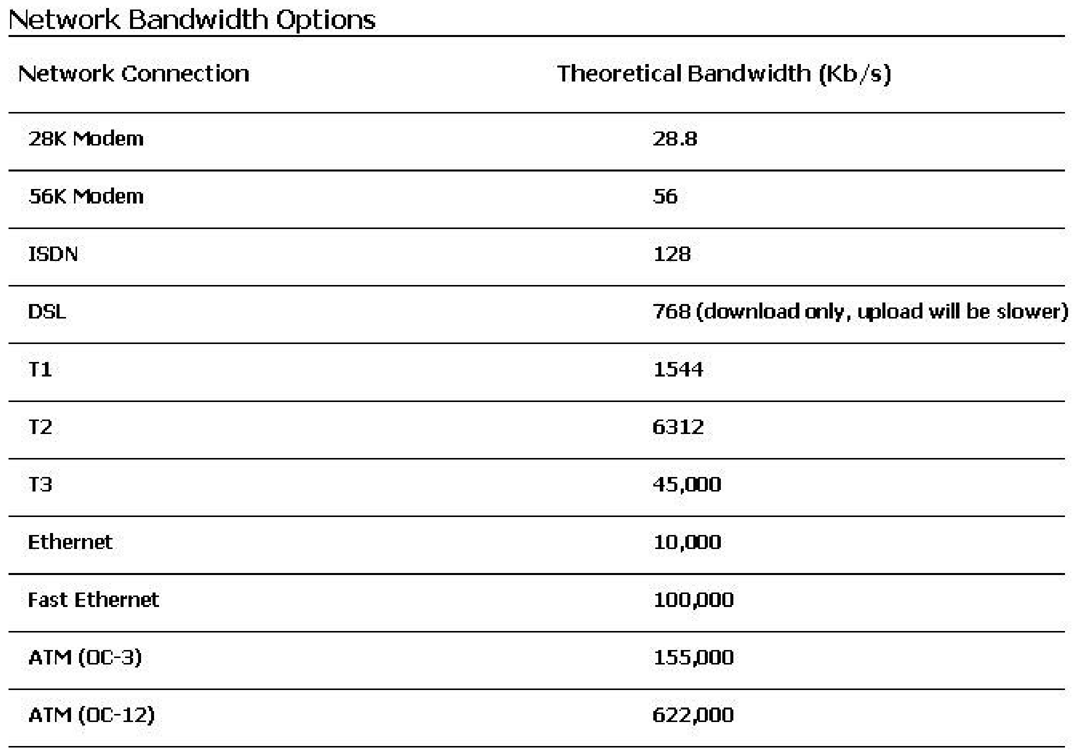
Using knowledge about the site requirements, and assuming a calculated need to serve 100,000 peak requests per hour, this chart can be used to estimate that, since one Spatial Server with a SPEC rate of 25 can handle an estimated 20,000 transactions per hour, five such Spatial Servers (and an ArcSDE data source) will be required to achieve the target peak rate of 100,000 transactions per hour.

Of course, this estimate can change based on the data to be served, the size of the output images, etc.[1]

Once decisions have been made about the hardware requirements of our application, it is time to consider how to make all of this hardware available to our audience. What is the Internet connection or bandwidth required for our users to get the desired experience?

Throughout the life of an ArcIMS application, there is an excellent chance that the cost of the bandwidth required to serve it to the Internet will be significantly higher than the cost of the hardware and software used to host it. This is important to note, because the cost of bandwidth is often overlooked. Bandwidth will be crucially important to the application. After all, how do the application and the ArcIMS output images travel from your server to the client's machine?

Bandwidth is measured in bits per second. One kilobit is one thousand bits. Note the word bits and remember that one bit equals 1/8 of a byte. This is an important thing to understand. It would be difficult to explain that your application could not support as many users as estimated because you only purchased access to 1/8 of the bandwidth that you really needed! This is an easy trap to fall into since output images and Internet application file sizes are measured in kilobytes. Try to avoid confusion as much as possible—files in kilobytes, bandwidth in kilobits.



Network Bandwidth Options

Network Connection	Theoretical Bandwidth (Kb/s)
28K Modem	28.8
56K Modem	56
ISDN	128
DSL	768 (download only, upload will be slower)
T1	1544
T2	6312
T3	45,000
Ethernet	10,000
Fast Ethernet	100,000
ATM (OC-3)	155,000
ATM (OC-12)	622,000

Figure 4—Network bandwidth connection types

[1] See Chapter 7 of the *Systems Design Strategies* white paper for more details.

There are several ways to illustrate the vital importance of bandwidth to an Internet application. One of the best is to use a little math to better understand the requirements from both ends—the client and the server.

On the server side, enough bandwidth will have to be available to support all of the users that are estimated to be accessing the application during the peak viewing period. Based on an estimate of 100,000 hits per hour, we know that we need to be able to potentially send 100,000 output images per hour across our upload Internet connection. If we can assume that each image will be about 75 kilobytes in size, then we can set up a quick formula:

75 kilobytes = 75000 bytes
75000 bytes ∗ 8 bits/1 byte = **600,000 bits/hit**

1 hour = 3600 seconds.
100,000 hits/hour ∗ 1 hour/3600 seconds = **27.8 hits/second**

27.8 hits/second ∗ 600,000 bits/hit = **16,680,000 bits/second**

16,680,000 bits/second ∗ 1 kilobit/1000 bits = **16,680 kilobits/second**

Knowing this value, and looking at Figure 4, we might conclude that a T2 connection (or, for a LAN, an Ethernet connection) is a suitable option for hosting this high-availability Web application if we want to meet the availability goals set forth in this paper.[1]

However, don't forget to factor the actual application into bandwidth calculations. The example that we just calculated does not take into account the size of the HTML and JavaScript files that will make up the client-side portion of our Web application. Let's say that we have a Web application that consists of 200K of HTML/JavaScript files and images that users will have to download. Using calculations similar to those above, it is shown that an additional 1.6 million bits (1600 kilobits) of information will need to be sent across our connection every time someone accesses our page for the first time. This would result in a download time of 28.57 seconds for a user accessing the site with a 56K modem on a dial-up Internet connection. If this is unacceptable, a smaller application footprint will be required to ensure that modem users are able to access the application in a timely manner.

These estimates do not take into account other factors beyond our control that can eat away at available bandwidth, not to mention other Internet applications that might be served through the same connection.[2] It's probably a good idea to overestimate the bandwidth calculations to be on the safe side.

[1] Contact your telecommunications provider for specific information about availability and cost of the required bandwidth connection.

[2] See Chapter 4 of *System Design Strategies* for further details.

Performance considerations

No discussion about high availability and ArcIMS would be complete without dealing with performance tuning. It's important to be aware of the specific options, tools, and settings within ArcIMS (and ArcSDE) that can be leveraged to increase overall performance and availability.

There are three primary areas to concentrate on when performance tuning an ArcIMS application: the ArcIMS configuration, the ArcIMS service, and how the data is stored. The following sections will examine each of these areas.

ArcIMS configuration

ArcIMS is a fully scalable GIS mapping application. This is due in large part to the separation of the product into specific components, each of which performs a specific task. All of these components communicate with each other via TCP/IP networking, and as such, can be loaded onto the same machine, or on two or more different machines.

A high-availability configuration will, by definition, reside on more than one machine. As you formulate the needs for security, availability, and performance, this configuration should begin to take a rudimentary shape. The configuration can be fine-tuned as the application enters the testing phase and any bugs are worked out.

In general terms, a true high-availability configuration will allow for the failure of one or many hosts and still keep the primary application up and running. The possibilities for such configurations are endless and beyond the scope of this discussion.[1]

In addition to the many configurations that are proposed in detail within the *System Design Strategies* document, there has been an ongoing effort between ESRI and the GIS user community to incorporate the use of network load balancing and clustering software within an ArcIMS configuration.

In a nutshell, network load balancing allows for a number of different Web servers to host a single virtual IP address on the Internet. There are a number of software and hardware packages available that allow for such a design. By allowing multiple Web servers to share the load of incoming transaction requests, you can create a redundant system that distributes transactions among all available ArcIMS applications. A typical network load balancing configuration is the "stovepipe" approach, where each node on the IP address cluster is set up as an individual ArcIMS application. Network load balancing is one of several steps in designing more integrated and complex fail-over configurations.

The ESRI Systems Integration group has formulated a strategy for installing multiple Monitor Services on a single machine to manage multiple Spatial Server processes.[2] The primary advantage of this configuration is that it overcomes one of the limitations of ArcIMS by allowing multiple Spatial Server processes on one machine to communicate with individual Application Server processes on other machines. Under normal ArcIMS installation conditions, a single Monitor Service (which manages all Spatial Server containers on a single machine) will only be able to communicate with a single Application Server (specified on installation). This new strategy allows for the

[1] More information can be obtained from the ESRI Systems Integration group (http://www.esri.com/systemsint/index.html) and the *Systems Design Strategies* white paper.

[2] ArcIMS 9.x users should consult the Install Guide Step 3a, which includes instructions on installing additional Monitor Services. ArcIMS 4.0.1 users may contact ESRI Technical Support to request document QFE-IMS-401-CQ00192545, which provides the same instructions.

installation of individual monitor processes to manage each Spatial Server container on a machine. The end result is totally independent Spatial Server processes running on the same physical machine, connecting to individual Application Server processes on other machines.

This is an important breakthrough from an availability standpoint. Now ArcIMS administrators are not limited to the stovepipe approach to high availability. This new option opens up some additional, potentially elegant solutions. Note the configuration represented by the following diagram:

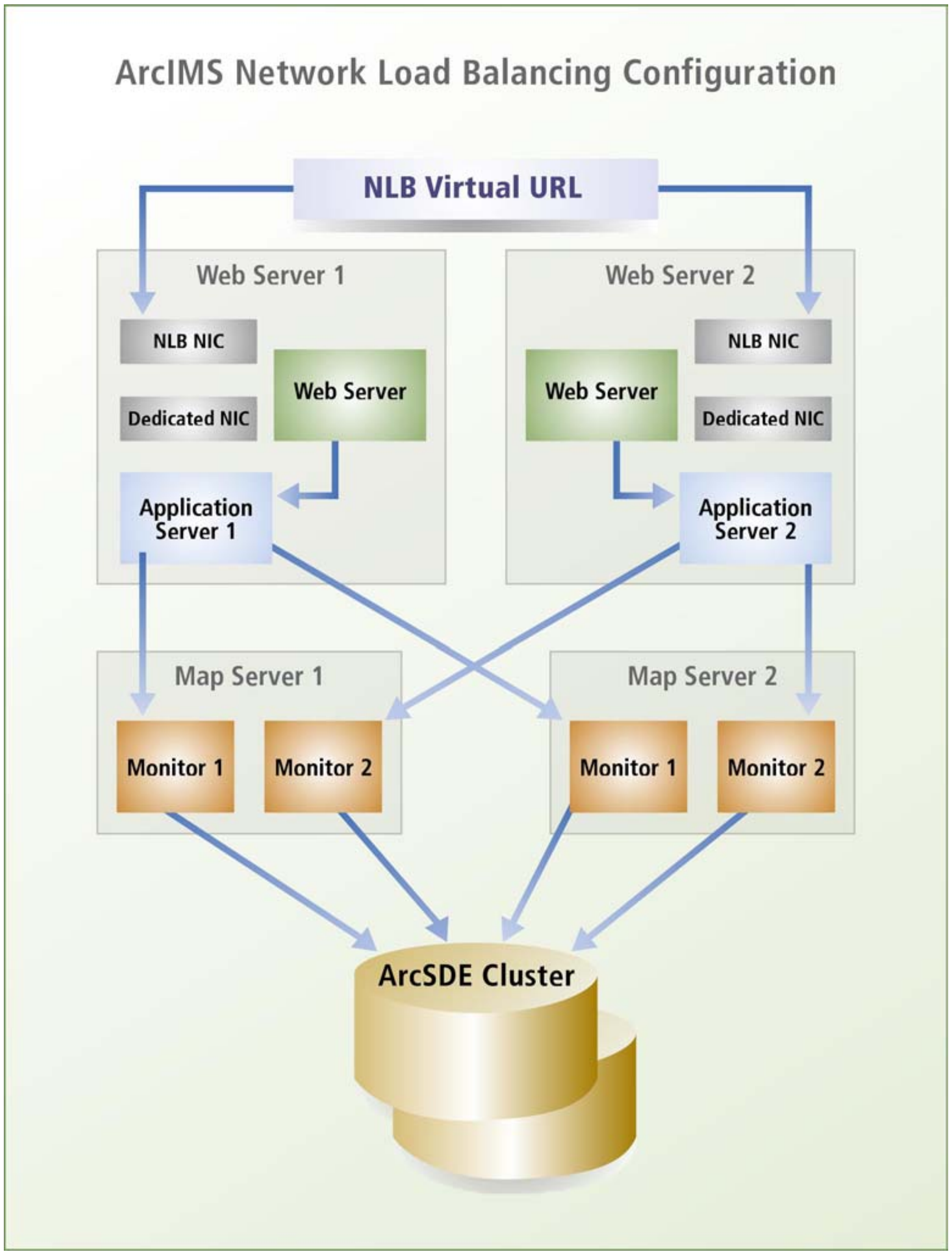


Figure 5—Example of Network Load Balancing architecture utilizing multiple monitor processes, as described within an ESRI Systems Integration Test Report (see http://www.esri.com/systemsint/kbase/testrep.html). Similar configurations are discussed within the *System Design Strategies* white paper.

This diagram illustrates the lines of communication available within the network load balancing environment. Note that there are two of each host—the Web server/Application Servers, Map (Spatial) Servers, and Database Servers. The two Web/Application Server machines would be hosting the same virtual IP address via the NLB switch or hub. Each of these Web servers also hosts the ArcIMS Servlet Connector and the ArcIMS Application Server. In addition to these two machines, a pair of Spatial Server hosts and a clustered set of database hosts are also configured.

Multiple Monitor Services are used on the Spatial Server machines to allow each of the Spatial Server processes on that particular machine to communicate with independent Application Servers.[1] In addition, we have clustered ArcSDE/database servers that can handle data requests from any number of Spatial Server machines.

This diagram represents the smallest scale of what could be considered a truly redundant system. This configuration allows for the failure of any machine within the application. If any such failure were to occur, the other machines within the configuration would be able to continue serving the content and application while the failed machine is repaired. This would result in diminished overall performance, but not in a failure in availability of the entire application.

The drop in performance could be remedied by extending this design to include fail-over systems that would come online as soon as a failure was detected within the system. In other words, the above configuration can simply be a starting point. Fail-over and redundant systems can be added to further account for system failures at any level.

Regardless of the ArcIMS configuration that is implemented, it is important to understand the different areas of concern for administration of the configuration and to keep track of these areas with vigilance. One tool available to ArcIMS Administrators is information logging.

ArcIMS includes a wide array of options for logging information that will be helpful to any site administrator. Logs for the ArcIMS Application Server, Spatial Server, and even individual Virtual Servers can provide a great deal of information about the performance and reliability of the configuration, and can record valuable information in the event of a failure. For more information about enabling logging for ArcIMS, see technical article 20844 at the ESRI Support Center.

Once you have enabled the desired logging for the ArcIMS configuration, you will notice a tremendous amount of information being written to the logs. For instance, the ArcIMS Application Server will record all communication between the client application and the server for all ArcIMS connectors. Information about the status of ArcIMS Spatial Servers being managed by that particular Application Server will also be logged. This can provide useful information about possible bugs in ArcIMS client applications or about the events leading up to the failure of the ArcIMS configuration.

[1] This level of detail is shown more clearly in Systems Integration's technical brief titled *Identifying Firewall TCP Server Ports in an Enterprise ArcIMS Configuration* (see Figure 2). This and other technical briefs are available at http://www.esri.com/systemsint/kbase/techbriefs.html

Arguably the most useful log files created by ArcIMS are those created by the ArcIMS Spatial Server. These log files contain information about the status of the Spatial Server, as well as the activity and performance of all of the Virtual Servers that are associated with it.

These log files contain a huge amount of information when fully enabled. Generating the output for these log files will consume some CPU and disk i/o resources, so enable the various logs only as needed. It can prove daunting to wade through all of the information stored in the logs; in fact, ESRI has dealt with the same problem hosting large ArcIMS sites such as the Geography Network and Realtor.com. To help with the task of digesting the logs, a Perl-based log file analyzer has been written that parses out the most useful information from the log files and writes it out to an intuitive HTML page or Microsoft® Excel spreadsheet. This log file analyzer (aimslogs.pl) is available to the public through the ESRI Support Center.

Information gained from the log files can certainly prove valuable in tuning your ArcIMS site, but there are still some areas that are so abstract and dynamic that it is difficult to quantify them in a way that we can log. One of these areas is the number of instances that are assigned to the Spatial Server process (or processes) for each Virtual Server.

Virtual Servers, Spatial Servers, and map services are administered within the standalone ArcIMS Administrator application, through the ArcIMS Service Administrator (a Java JSP application that installs as an option with ArcIMS), or from the command line (see technical article 19396 at the ESRI Support Center). At first glance, the default settings within Administrator appear to be adequate for most ArcIMS sites. However, small changes to these settings can have a huge overall impact on the performance of the application and the viewing experience of its users.

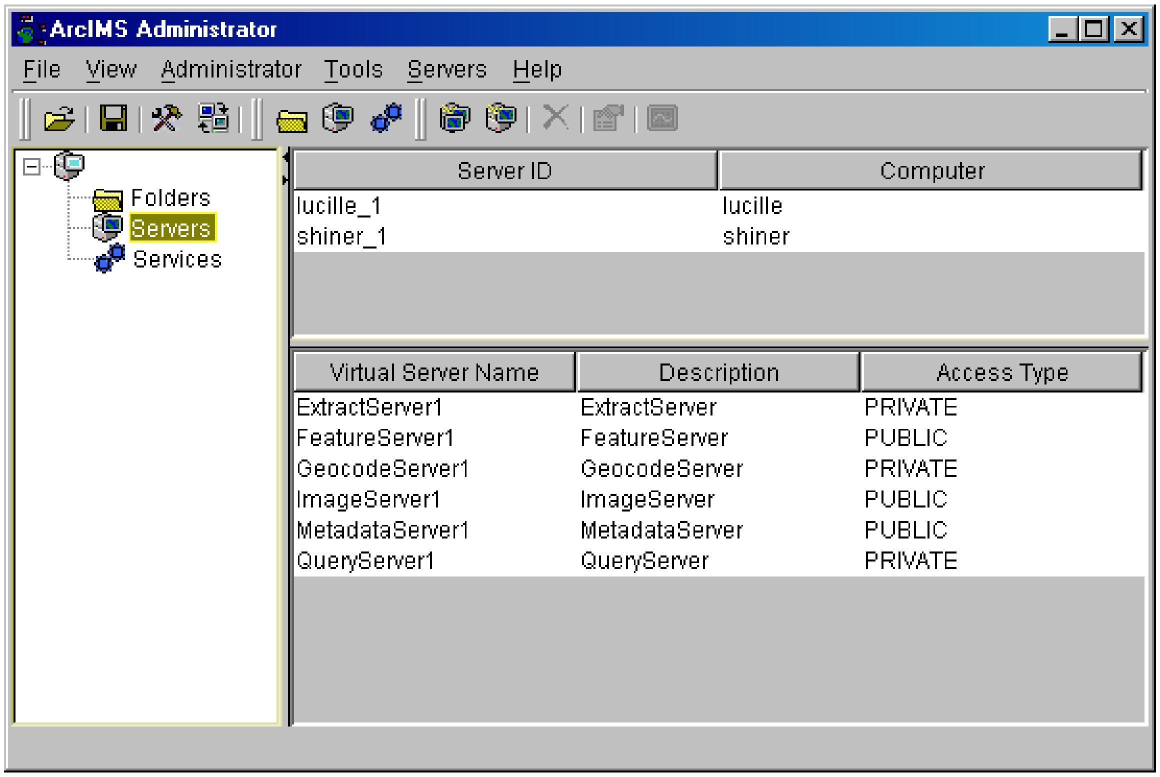
There are settings within ArcIMS Administrator to help project administrators get the most out of the available hardware. Most ArcIMS users are aware that the creation of a map service requires the specification of a Virtual Server. Many may not be aware that the Virtual Server allows the power to delegate site resources to a particular map service through configuration properties.

This delegation of resources is made available through the Servers dialog box. The settings in this dialog follow a few simple rules:

- For each map service, only one Virtual Server may be assigned
- For each Virtual Server, multiple Spatial Servers can be assigned
- For each Spatial Server, a number of instances can be set

A complete explanation of Virtual Server grouping and instances is beyond the scope of this paper; however, there are many sources of information available to you.[1]

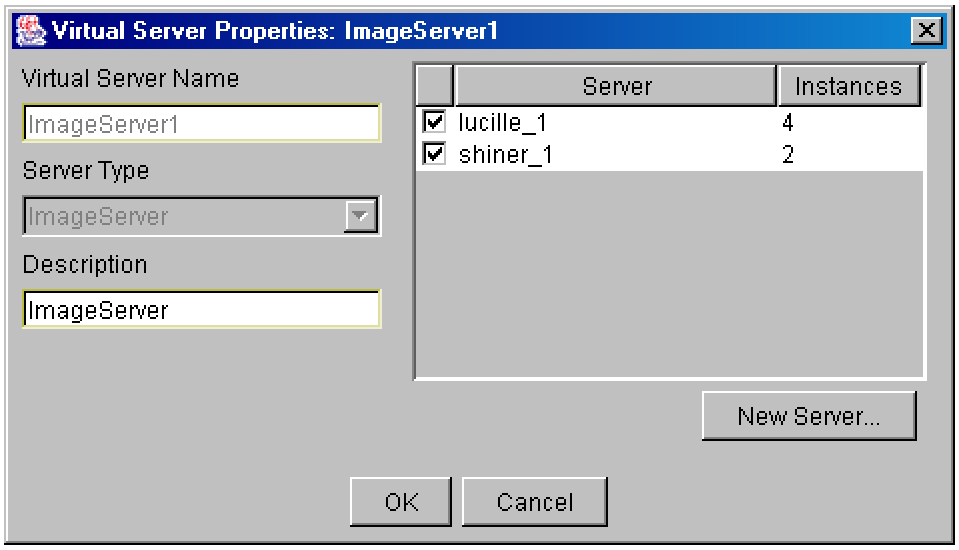
[1] More information is available from the *Using ArcIMS* book and the Introduction to ArcIMS and ArcIMS Administration courses. ESRI Systems Integration Services has also provided a technical brief called *ArcIMS Configuration Performance Factors*, available from http://www.esri.com/systemsint/kbase/techbriefs.html.



Server ID	Computer
lucille_1	lucille
shiner_1	shiner

Virtual Server Name	Description	Access Type
ExtractServer1	ExtractServer	PRIVATE
FeatureServer1	FeatureServer	PUBLIC
GeocodeServer1	GeocodeServer	PRIVATE
ImageServer1	ImageServer	PUBLIC
MetadataServer1	MetadataServer	PUBLIC
QueryServer1	QueryServer	PRIVATE

Figure 6—Servers dialog box of ArcIMS Administrator application showing existence of multiple Spatial Server hosts and all available Virtual Servers.



Virtual Server Name: ImageServer1
Server Type: ImageServer
Description: ImageServer

	Server	Instances
✓	lucille_1	4
✓	shiner_1	2

Figure 7—ArcIMS Administrator dialog box for setting Virtual Server properties. Note that instances for a single Virtual Server can be balanced across all available Spatial Server processes.

In essence, ArcIMS administrators should experiment to find the smallest number of instances that permits 100 percent utilization of the hardware and software resources when under peak load. At that point, performance will be in balance with the ability to serve many simultaneous requests. An instance setting of 3 is a good starting point, but larger and faster sites might want to try a higher number. Use the Virtual Server log files and Performance Monitor tool (available from the Servers toolbar in ArcIMS Administrator) to identify two key factors for a service: average response time and number of requests queued. These two factors will have to be in balance for ArcIMS to perform at maximum capacity.

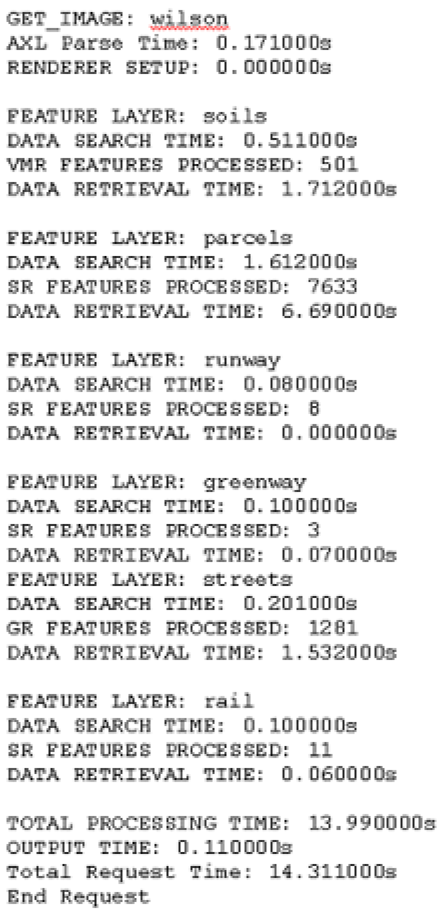
This topic is worth considering in detail. As the number of instances is increased, the Spatial Server(s) will be able to work on more requests simultaneously (good!). Beyond a certain point, however, there will be a drop in overall performance because all instances still have to share the same resources (CPU), resulting in a higher average response time when under load (bad!). Alternatively, as the number of instances is decreased, there will be an increase in the average performance of the service through lower response times (good!), but all of the available hardware resources may not be utilized. Under load, users are forced to wait in the Application Server queue for an instance to become available, even though CPU and other resource utilization may be well under 100 percent (bad!). Experiment to find the right number of instances that balances these two factors, and you'll achieve optimal utilization of your resources as well as an optimal user experience.

ArcIMS service tuning

Tuning your ArcIMS map services is one of the cheapest and easiest ways to make huge gains in performance. It is very expensive to purchase new equipment, bandwidth, and staff to increase performance and availability. It is very inexpensive to tune map services to be as efficient as possible.

In the previous section, the use of log files was recommended as a valuable tool for tuning the overall performance of an ArcIMS application and to pinpoint possible problems with communication between the various components of ArcIMS. Additionally, the ArcIMS log files can help you tune map services for maximum performance.

Specifically, the ArcIMS Virtual Server log files contain a wealth of information specific to the ArcIMS services running on the system. This information includes the overall response time for a request, the type of request, the time of the request, the process ID, the time it took to read the ArcXML request, the layers requested, the type of rendering for each layer, the number of features requested for that layer, and so on.



```
GET_IMAGE: wilson
AXL Parse Time: 0.171000s
RENDERER SETUP: 0.000000s

FEATURE LAYER: soils
DATA SEARCH TIME: 0.511000s
VMR FEATURES PROCESSED: 501
DATA RETRIEVAL TIME: 1.712000s

FEATURE LAYER: parcels
DATA SEARCH TIME: 1.612000s
SR FEATURES PROCESSED: 7633
DATA RETRIEVAL TIME: 6.690000s

FEATURE LAYER: runway
DATA SEARCH TIME: 0.080000s
SR FEATURES PROCESSED: 8
DATA RETRIEVAL TIME: 0.000000s

FEATURE LAYER: greenway
DATA SEARCH TIME: 0.100000s
SR FEATURES PROCESSED: 3
DATA RETRIEVAL TIME: 0.070000s
FEATURE LAYER: streets
DATA SEARCH TIME: 0.201000s
GR FEATURES PROCESSED: 1281
DATA RETRIEVAL TIME: 1.532000s

FEATURE LAYER: rail
DATA SEARCH TIME: 0.100000s
SR FEATURES PROCESSED: 11
DATA RETRIEVAL TIME: 0.060000s

TOTAL PROCESSING TIME: 13.990000s
OUTPUT TIME: 0.110000s
Total Request Time: 14.311000s
End Request
```

Figure 8—Simplified ArcIMS Image Virtual Server log file output. Total request time for this individual request is more than 14 seconds, longer than most users may be willing to wait. Note the "data retrieval time" logged for each layer of the request. Is there a single culprit for the performance bottleneck? How could it be addressed?

The information in the Virtual Server logs is crucial to identifying potential bottlenecks within an ArcIMS service. The information can help an ArcIMS administrator identify the slower areas of a service and address those areas with more efficient use of scale factors, rendering, or labeling, either within the map configuration file or in the individual requests coming from the client application. As pointed out on page 22, the ArcIMS log file analyzer can help an administrator make more sense of these files by parsing and writing specific information from the log files into an HTML document or an Excel spreadsheet. This can allow for more detailed statistical analysis of the log file information.

The information that is written to the log files can help you design a more efficient application. You could use the information in the log files, along with a simplified client viewer like the SendArcXML Viewer, to simulate different client requests and see how the service responds. This can be especially valuable for testing advanced functionality, such as feature extraction and geocoding.

Also keep in mind that the use of ArcMap™ Image Services within ArcIMS will have a significant negative impact on performance when compared to a traditional ArcIMS

Image Service, simply because the increased functionality available in ArcMap Image Services has a higher processing cost.

ArcMap Image Services can be very desirable due to the additional functionality that they provide. The ArcMap Server extension offers ArcIMS administrators the ability to create detailed Image Services based on the information in an ArcGIS map exchange document (.mxd) or an ArcReader™ published map file (.pmf). However, the additional functionality of complex rendering, geodatabase versioning, annotation, and CAD files also results in additional overhead. This additional overhead must be accounted for when weighing the suitability of using the ArcMap Image Services within a high-availability application.

Also keep in mind that there will be additional administration requirements specific to the ArcMap Image Virtual Servers. In order for these Virtual Servers to function reliably and efficiently, a recycling routine must be set up within ArcIMS Administrator. This recycling routine will stop and start the Spatial Server processes allocated to the ArcMap Image Server on a specified schedule to free up system resources and keep service information up to date.[1]

In most cases, the services used within a high-availability application will be of the traditional image type. These services are designed to send output images to users based on a combination of parameters set within the map configuration file and the client application. Images are created on a request-by-request basis and exist on the server for only a short time.

When a request comes in from a client application for a map image, the ArcIMS Spatial Server must first retrieve the data for the visible layers within the extent of the request and then render these features into an output image that can be sent to the client's browser via a URL. Once again, the use of the ArcIMS Virtual Server log files can provide a wealth of information about the efficiency of the service.

Many factors can slow down the response time for an image request. In short, any additional functionality that you include in a service will increase overhead and diminish overall performance. Some of the things to limit or avoid include:

- Projecting layers on the fly
- Complex labeling
- Complex rendering
- Dynamic layers and acetate layers
- Displaying many features at a small scale

Perhaps the most important of these is the last. It is good practice to avoid displaying too many features for a layer at a small viewing scale (i.e., when zoomed out). After all, if you are creating an 800 by 600 pixel map image, is the user going to be able to clearly see 45,000 parcel polygons displayed all at once? Probably not. Use scale factors within

[1] Additional information about the specific administration needs for the ArcMap Image Server is available from the ArcIMS Help documentation.

the map configuration file to control the visibility of layers at appropriate extents, and you may easily gain in performance at no cost.[1]

In addition to these points, there are other specific areas in which to tune a map configuration file to optimize performance. The general rule is: The simpler the map, the faster it will be created. It is important to know the different options available for your ArcIMS service. Having access to all of the information is key. Consult the ArcXML Programmer's Reference Guide for specific information about the options available for both map configuration files and handling requests.

The only way to know what works best for your application is to test out the functionality that you want and gauge the effect on overall performance. You may find that a seemingly small change in a map configuration file or in the requests generated by your ArcIMS service can have a large impact on performance.

Data storage

Data storage plays a major role in the performance and availability of an ArcIMS application. Inefficient data storage techniques can make all other performance tuning useless. Culprits such as disk contention, inefficient database tuning, and a general lack of attention can bring an otherwise robust application to a crawl.

In general terms, ArcIMS is designed to publish static data to the Internet. It is worthwhile to prepare this data for publication prior to including it in an ArcIMS service. The following items should be considered:

- Data inclusion and generalization
- Coordinate system and projection information
- Database and file access tuning
- Using ArcSDE

One of the simplest things that can be done to make an ArcIMS application more efficient is to only serve the data that is actually *needed by* the application. This may sound obvious, but it can be tempting to throw the same data layers that you use in other GIS applications into an ArcIMS application without considering the effect on performance. A difference of a tenth of a second in data search time can really add up if a thousand people are simultaneously viewing your Web site.

Eliminate from your data any unnecessary features or attributes. This will save valuable processing time when the Spatial Server is gathering information to formulate a response. You might want to generalize your data using one of several utilities within ESRI Desktop applications; for example, the Generalize tool can be found on the Advanced Editing toolbar in ArcGIS 8.3 or higher.

Although useful, the ArcIMS projection engine requires significant system resources to project data on-the-fly. It is most efficient to choose a coordinate system, projection, and units during the design phase of development, and then project all your data into the same coordinate system before using it in an ArcIMS map service. If you must use data in different coordinate systems, make sure to define the coordinate system of each and every dataset. The COORDSYS tag is probably the most efficient method for file-based data;

[1] The use of scale factors is essential for ArcIMS performance tuning. There are a number of options available, including using several copies of the same dataset as different layers in a map service. These layers can be alternated based on the visible extent that is requested from the user. The more generalized data will be displayed when zoomed out, and the more detailed data will be displayed as the user zooms in past a certain scale factor.

see the Projection Elements section of the *ArcXML Programmer's Reference Guide* for more information.

Database and file access tuning is a unique subject unto itself. Many people structure their lives around perfecting the art and science of effective data access. One of the common buzzwords in recent years has been the storage area network, or SAN. Indeed, it has been said that the SAN is to data access what LAN and WAN (wide area network) are to networking.

The science of designing and implementing storage area networks is nothing new, and there is a wide array of material on the subject. The essential thing to realize is that all of the ideas and strategies that you will see discussed for high-speed, reliable data access can be directly applied to a high-availability ArcIMS application. After all, the need for SANs came about as a result to store and serve large amounts of data to a dynamic client base. Many of the same problems that had to be overcome for large e-commerce sites are the same problems that ArcIMS administrators face when trying to serve a large amount of GIS data. Strategies for hosting, updating, and sharing the data must be formulated as a part of the application.

The use of ArcSDE can help bring the power and flexibility of a relational database to your application, thereby increasing overall performance and scalability. Large, enterprise database management systems, such as Oracle®, SQLServer™, IBM®, DB2®, and Informix®, allow virtually unlimited options for storing, updating, and preparing data for publication. The ability to incorporate automated backup routines, monitor user access, and cluster multiple servers to serve a single database provides a very extensive toolbox for ArcIMS application developers. The database management system can handle much of the dirty work that ArcIMS would otherwise have to handle with file-based data.

ArcSDE adds GIS components to a relational database, allowing it to leverage its power with GIS datasets, including the ability to create large, seamless imagery databases or to group features of a feature layer together for faster display. ArcSDE also has the ability to create and store reduced-resolution copies of data layers to further increase display performance. These are known as pyramid layers for raster datasets and grouped layers for feature datasets. The use of pyramid and grouped layers can significantly improve performance within an application with only a slight cost in additional storage.

If you don't choose to incorporate ArcSDE into your application, file-based data is a perfectly acceptable data source, but the absence of the RDBMS and ArcSDE mean that you must prepare your GIS database for high availability in other ways. Disk storage options and system configurations that stress speed and reliability are key. Storage area design strategies that will avoid disk contention and other bottlenecks should be investigated. After all, if you have 300 people accessing your Web site at the same time, and all are looking at the same file at the same location, data access limitations of the storage area might slow the entire application down. If the data cannot be retrieved as fast as the application is requesting it, the storage device or network has just become a performance bottleneck that affects all users, reducing the likelihood that they will receive their responses in a timely manner. The proper use of fast storage devices (such as RAID) or efficient storage area network design can be leveraged to avoid this if file-based data is a must for your application.[1]

When weighing the decision between an ArcSDE and a file-based data source, it is very important to be as informed as possible. Make sure to consider all the various factors such as hardware, software, network, and staffing.

◆ **ArcSDE vs. File Sharing**

◆ **Compare amount of data transferred over network for a request**

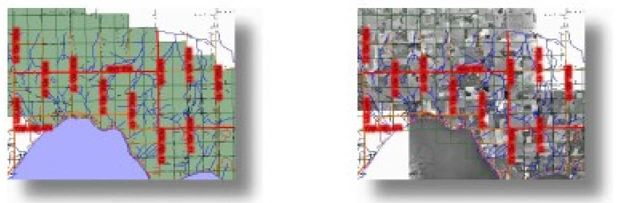


Image 1 (vector only) **Image 2** (vector and image)

	Image 1	Image 2
ArcSDE Application Server	137 KB	353 KB
ArcSDE Direct Connect	253 KB	499 KB
UNC file sharing	1124 KB	116095 KB

Figure 9—Illustration showing the difference in network traffic for requests to identical services: One service created using data from an ArcSDE database and the other created using file-based data. This is just one area in which ArcSDE can represent a significant advantage over file-based data.

More information about the performance-enhancing features of ArcSDE is available from the *ArcSDE Configuration and Tuning Guide* (which comes with the ArcSDE product), the series of instructor-led ArcSDE Administration courses (see http://www.esri.com/training/index.html), or courses from the ESRI Virtual Campus (http://campus.esri.com). Information specific to the scaling of ArcSDE from a software and hardware standpoint is available within the aforementioned *System Design Strategies* document.

Staging

Once you have configured all of the components of your ArcIMS application, you will need to make sure that it will perform reliably once released to the public. The best way to do this is through comprehensive and thorough testing. Good testing can reveal issues that might have been previously overlooked and can save a good deal of downtime or resource allocation in the future.

Ideally, there should be four separate phases of implementing a high-availability ArcIMS application: design, development, staging, and production. Once your project has been designed and developed, you're ready to move on to the staging phase—in other words, testing.

[1] For more information about leveraging DAS, SAN, and NAS design for GIS data, see the *Enterprise GIS Storage Technologies* technical brief from the ESRI Systems Integration Team, accessible at http://www.esri.com/systemsint/kbase/techbriefs.html.

The best scenario for staging a high-availability application is to set up a test environment that will come as close to the production environment as possible. Try to set up the staging system with the exact same components and on the same hardware that will be used in production. If testing with security or monitoring software, make sure it is as close to the versions used in production as possible.

There are many performance evaluation software suites on the market, such as Microsoft Application Center Test, which can simulate target conditions on the site and record how well the application and configuration handles it. Whatever monitoring tools you employ, they can provide hard numbers that can be used to modify availability calculations—part of your testing strategy could simulate a hardware, software, or network failure so that you can evaluate the ability of the network to continue supporting the application, thereby gaining tangible information about system recovery (MTTR) values.

As an alternative to commercial or shareware monitoring and testing tools, you may want to consider building a custom load test application. This could be accomplished with a variety of programming languages and the appropriate ArcIMS Connector (Java, ColdFusion, ActiveX, or Servlet Connector). To ensure accurate results, your load testing application should be designed to run independently of your primary ArcIMS application. One example of a custom ArcIMS load testing application can be found on the ArcScripts Web page (http://arcscripts.esri.com/details.asp?dbid=12388).

Be sure to incorporate the ArcIMS and ArcSDE log files within your monitoring strategy. Staging is a perfect place to fine-tune indexing, pyramid layers, and grid sizes within ArcSDE or to identify bottlenecks in the system when production conditions are simulated. Testing every conceivable scenario will help ensure a successful rollout.

Production and maintenance

Once staging tests are complete, the production phase can commence. This is when all your careful planning pays off—your high-availability ArcIMS application is ready to be released to the world.

One of the primary concerns of a production environment is security. In most cases, security strategies will already be in place in a production Internet server environment. There is generally some array of perimeter network architecture separating the Internet from the internal network of your organization. In most cases, these perimeter networks are defined by the use of packet filtering routers, more commonly known as firewalls.

Understanding the roles of firewalls in your production environment is a key element of successful implementation of an ArcIMS application. In order to properly place firewalls between the different components of your application, you will need to understand the network communication channels for all ArcIMS components, including ArcSDE if applicable. Packet header information should be understood and incorporated into the firewall software so that it only allows communication to and from the appropriate components. A Systems Integration technical brief titled *Identifying Firewall TCP Server Ports In a Enterprise ArcIMS Configuration* explains the necessary firewall configuration for ArcIMS in additional depth (see http://www.esri.com/systemsint/kbase/techbriefs.html). Detailed information on using firewalls within a distributed ArcIMS installation is also available in the *System Design Strategies* white paper.

Regardless of the configuration preferred by your organization, certain strategies should be put in place to make sure that your system is reasonably secure from hackers. A greater investigation of security strategies specific to ArcIMS is available through the *ArcIMS Administration* instructor-led course. There is also a wealth of information about Web security strategies in print and digital form. Consult with your IT department to formulate a comprehensive plan for Internet security for your ArcIMS application.

Once the application has been moved into a production Web environment, the work is far from over. In fact, the level of attention that the application will need might increase significantly. You will need to keep a close eye on things to make sure that the application performs as expected as popularity and traffic increase. It would be ideal if all possible contingencies have been identified and dealt with during staging tests, but there will inevitably be a few issues that arise during the production phase. It is important to expect this and be prepared to deal with issues as efficiently as possible.

Availability should be monitored regularly, and a production plan should be clearly communicated to all who are involved. Who is responsible for maintaining the availability of the site? How will application updates or data updates be handled? How much downtime can be expected during these updates?

Site monitoring software, such as Mercury Interactive's SiteScope®, can make sure that failures or security breaches are recognized immediately. Alternatively, custom monitoring applications can be written to detect failures within the system and notify the proper administration contacts so that time to recovery (MTTR) can be minimized. Proper monitoring is crucial to the long-term success and overall availability of the site.

Conclusion

Implementing a high-availability ArcIMS application is not a trivial matter. Careful planning is essential for the final product to be successful. However, there is no reason to believe that the task is beyond the capabilities of any organization. Many highly available ArcIMS applications currently exist, hosted by organizations that have successfully navigated the process outlined in this paper.

By now you should feel confident that you can identify the critical points and processes that must be considered at the outset, and formulate a strategy that best allows you to handle each phase of implementation. The conceptual knowledge in this paper should be a solid foundation for designing, developing, staging, and producing a high-availability ArcIMS application. Soon you can add your application to the growing list of successful, reliable, and highly available Web sites that have been created with ArcIMS.

Support

Enterprise GIS system design is addressed in the *System Design Strategies* white paper, which can be accessed at http://www.esri.com/systemsint/index.html.

For answers to additional GIS capacity planning and solution questions, contact ESRI Systems Integration at sihelp@esri.com.

For technical support, contact ESRI Technical Support at http://support.esri.com/contactus.

For information on instructor-led or Virtual Campus courses, view the Training and Events Web site at http://www.esri.com/training_events.html.