The Architecture of ArcIMS, a Distributed Internet Map Server

Russell East, Roop Goyal, Art Haddad, Alexander Konovalov, Andrea Rosso, Mike Tait, and Jay Theodore

ESRI
380 New York Street
Redlands, CA 92373-8100
Phone: 909-793-2853
Fax: 909-307-3112
{reast, rgoyal, ahaddad, akonovalov, arosso, mtait, jtheodore}@esri.com
http://www.esri.com/arcims

Abstract. ESRI® ArcIMS® is an Internet Map Server software that facilitates authoring of maps, designing of Web sites using them, and their publication on the Internet. Its distributed architecture offers customization of server, plugability of components, scalability, load balancing, and fail over/recovery and facilitates 24x7 operations. This paper describes the architecture of ArcIMS 3, its components, and some of the live Web sites that use it. The paper also discusses some considerations for the next version of ArcIMS.

1 Introduction

GIS technology has historically been developed and deployed on monolithic (i.e., 1-tier) systems, such as ESRI's ArcInfo™ software [5], that have applications and geographic data installed on them. On such systems, if users want to view a map or make a query, they need to walk to the host machine. Network file servers enabled the sharing of data between machines on LANs within an enterprise. Geographic data is usually massive, expensive, requires powerful computers to host it, and requires its management by GIS experts. The above methods of hosting geographic data on a stand-alone machine or on an LAN are suitable only for GIS-centric high-end users. There are numerous users whose core businesses are not GIS-centric, but who are interested in making use of geographic information to enhance their business. For example, a toy company's marketing department would like to study people's preferences based on geographic regions and their household incomes. Although geographic information is not the mainstream business for these users, it is an important piece in the puzzle of decision-making.

Client/Server (i.e., 2-tier) architecture offers a separation between GIS users and GIS hosts. For example, ESRI's ArcSDE™ software (http://www.esri.com/sde) allows clients to access spatial data both from machines other than the Server machine and from different networks. The separation makes such systems more distributed than 1-

tier systems, but they still may not be well suited to occasional users of geographic information, whose users need to have custom client applications on their machines.

Most geographic information users, experts, and novices; individuals and organizations; and occasional as well as frequent users, have access to the Internet (a.k.a. World Wide Web) and Web browsers. This makes the Web the vehicle of choice to serve people's GIS needs. The Internet also offers the advantage of existing networking protocols and languages, specifically HTTP, TCP, IP, and HTML. The separation and connectivity between hosts and users provided by the Internet enable the hosting of massive geographic data on powerful computers managed by Internet–GIS professionals. A data set can be shared and viewed by many people using their Web browsers, which makes it economically feasible to host expensive, accurate, and up-to-date data. Three-tier architecture, including the Web tier, has significant performance improvements over 2-tier architecture [1]. In order to provide scalability, fail over/recovery, and load balancing, ArcIMS is designed as a distributed multitier (i.e., *n*-tier) system.

Section 2 of this paper describes the architecture and components of ArcIMS 3. Section 3 discusses the advantages of the architecture. Section 4 briefly describes some Web sites that are live using ArcIMS. Section 5 discusses the technological and architectural issues under consideration for the next version of ArcIMS. Section 6 presents concluding remarks.

2 The Architecture of ArcIMS 3

ArcIMS can be divided into three parts on the basis of their functionality: *Spatial Server*, *Middleware*, and *Client-tier* (Fig. 1). Middleware and Spatial Server have multiple tiers within themselves.

Spatial Server has five components: *Image Server, Feature Server, Query Server, Geocode Server,* and *Extract Server*. The Image Server generates images corresponding to the requests from clients. Feature Server streams vector features to the clients. Geocode Server provides geocoding services such as locating an address on a map. Query server handles spatial and attribute queries for Image services. Extract server responds to feature subset extraction requests.

ArcIMS Middleware has *Connectors, Application Server, Monitor,* and *Tasker*. Application Server receives requests from clients through Connectors; it responds to site-information requests and forwards the requests meant for Spatial Server to respective Spatial Servers. Once Spatial Server sends back the response, it conveys it back to the client that sent the request. A Monitor starts Spatial Servers and makes sure that Spatial Servers are always running. The Tasker cleans up the images generated by Image Server after a predefined time interval so that the Web site does not run out of disk space.

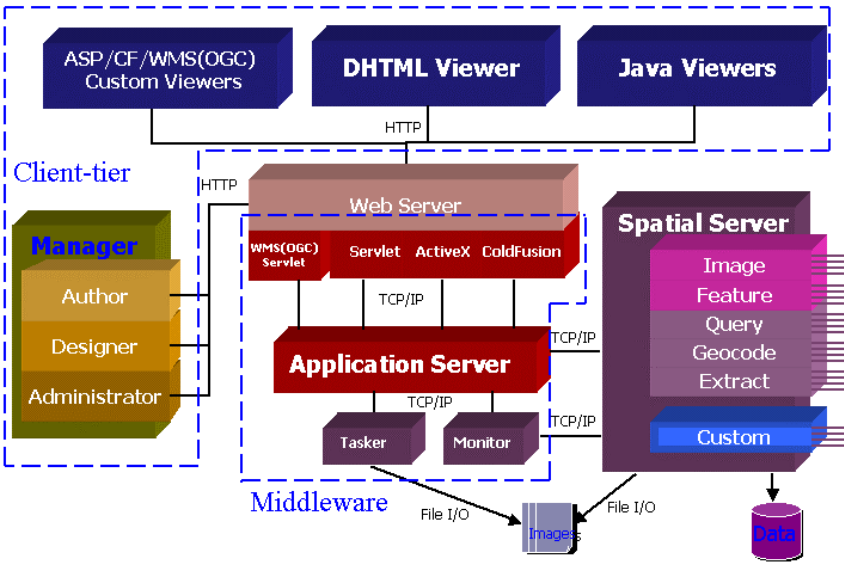


Fig. 1. An ArcIMS site has Client tier, Middleware, Spatial Server, and data sources.

ArcIMS introduces the concepts of MapServices, MapNotes, and EditNotes, which are defined below:

- A *MapService* is a configured map that is published on an ArcIMS Web site. A site can have multiple MapServices running concurrently.
- A *MapNote* is a text and/or graphic annotation(s) on published MapServices.
- An *EditNote* is an edited (i.e., deleted, changed, or inserted) spatial feature on a published MapService.

Client tier has stand-alone Java™ applications—*Author*, *Designer*, and *Administrator*—which are used for authoring MapServices, designing Web sites using MapServices, and administering the ArcIMS site, respectively. Client tier talks to Middleware using ESRI's ArcXML [2], which is a language developed on top of extended markup language (XML). ArcXML requests from Client tier are sent over HTTP or secure-HTTP (S-HTTP) connections, and responses from Middleware to clients also come back in ArcXML. Users can view published MapServices over the Internet using their Web browsers, which are complemented by viewers on the server side.

ArcIMS facilitates collaborative mapping using MapNotes and EditNotes. For example, a user may want to indicate the extent of a forest fire on a prepublished map by drawing a polygon or a circle and annotate the graphic elements with text labels. Users can converse with each other using phone or e-mail while viewing the MapNotes drawn by other users across the Internet. EditNotes allow editing of feature data and their sharing across the Internet. A MapService can have more than one layer of MapNotes and EditNotes; users can choose to see the layers of their interest.

In order to provide platform independence, the middleware uses Sun's Java technology
(http://www.javasoft.com/) for most components. In order to use existing libraries and to provide connectivity to third party data sources, Spatial Server is developed in C++. To provide choices to ArcIMS developers, a family of viewers is developed in Dynamic HTML (DHTML), Java, Microsoft's (http://www.microsoft.com/) Active Server Pages (ASP), and Allaire's (http://www.allaire.com/) ColdFusion® technologies.

2.1 Spatial Server

The ArcIMS Spatial Server, a stand-alone application, is composed of *Container, AXLParser*, *Data Access Manager*, and five standard components: Image Server, Feature Server, Query Server, Geocode Server, and Extract Server (Fig. 2).

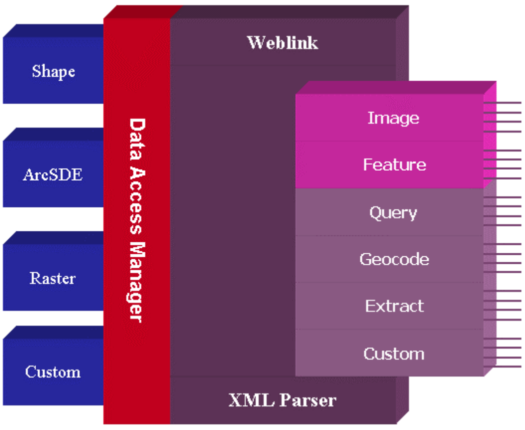


Fig. 2. Components of Spatial Server

A Container is launched by a Monitor and managed by the Application Server. After starting, the Container creates two controlling threads, which keep it connected to the Monitor and Application Server during the whole session. If Container crashes, then Monitor is able to detect the crash and restart it. Container's communications with Application Server and Monitor are realized in an XML-based protocol and performed via the *Weblink* module. The main tasks of the Container are to register, start, and stop Server Components. Every instance of each Server Component is launched within the context of a separate thread inside Container and works independently of all other threads. All the components use Data Access Manager to access ESRI's shape (SHP) files, ArcSDE databases, and Raster databases. The Manager also provides an interface to plug in a custom database

.

Five standard Server Components are available out of the box. Each of them serves a particular purpose, and together, they cover a wide range of GIS users' needs. A request is parsed by AXL Parser and processed by Components that produce proper outputs in the forms of images and binary and ArcXML streams.

Image Server processes data extracted from Data Access Manager and produces map images in JPG, GIF, and PNG formats, which may be downloaded by a thin client such as HTML viewer. Query Server runs in the background of Image Server and handles all spatial and attribute queries. Feature Server is used by thick clients, such as Java viewers, to get raw data in the form of a highly compressed binary stream. It is up to the client to unpack, store, and render the data. Feature Server is able to serve a client with a significantly fewer number of interactions. Extract Server extracts data and packs it in the form of zipped SHP/DBF files. The data may be downloaded in the same way as images generated by Image Server. Geocode Server can be used with both Image and Feature Servers to find the location for a street address and/or street's intersection. This component is built on top of the ArcView® GIS 3.2 geocoding engine and works with both SHP and ArcSDE data sources. The Container's internal structure makes it possible to use Custom Server Components. The *Route*MAP™ IMS extension is one such example.

2.2 Middleware

Middleware, as the name suggests, is the "junction-box" of ArcIMS. Interactions between clients and Spatial Server pass through Middleware (Fig. 3). Middleware works closely with Web servers. The most common Web servers are Netscape® *iPlanet*™ *(*http://www.iplanet.com/*)*, Microsoft® *IIS (*http://www.microsoft.com/iis/*)*, and Apache Foundation (http://www.apache.org/) *Apache* Web server. Sections 2.2.1–4 describe the four parts of Middleware, and Section 2.2.5 describes the concept of *Virtual Servers* used for grouping Servers.

2.2.1 Connectors. A client request is received by a Web server, which conveys the request to Application Server with the help of a Connector. A Connector is a software program that runs on the machine that runs the Web server and works closely with it. It is configured to work with the Application Server of an ArcIMS site. It connects to the Connector Port of the Application Server, which uses a Server type Socket (Fig. 3). The Application Server assigns it a new Socket on which it can send the request and receive the response from the Application Server. ArcIMS has four types of Connectors: *Servlet Connector, WMS* C*onnector, ActiveX Connector*, and *ColdFusion Connector* (Fig. 1). The Servlet Connector works with both Image and Feature MapServices, whereas the remaining three Connectors work with only Image MapSevices. Sections 2.2.1.1–4 describe the four types of Connectors.
2.2.1.1 Servlet Connector. The ArcIMS Servlet Connector is a Java Servlet, which is a serverside Java applet [4]. We have chosen the Servlet API to develop a Connector because Servlets have better request handling capabilities than a Web server's native scripts or Common Gateway Interface (CGI) scripts. To run a Java Servlet, a Web server needs a Servlet Engine, which can be either native or external to the Web server. For example, iPlanet has a native Servlet Engine. IIS can be configured to use

an external Servlet Engine such as New Atlanta's Servlet Exec (http://www.servletexec.com/) or Allaire's JRun (http://www.jrun.com/).

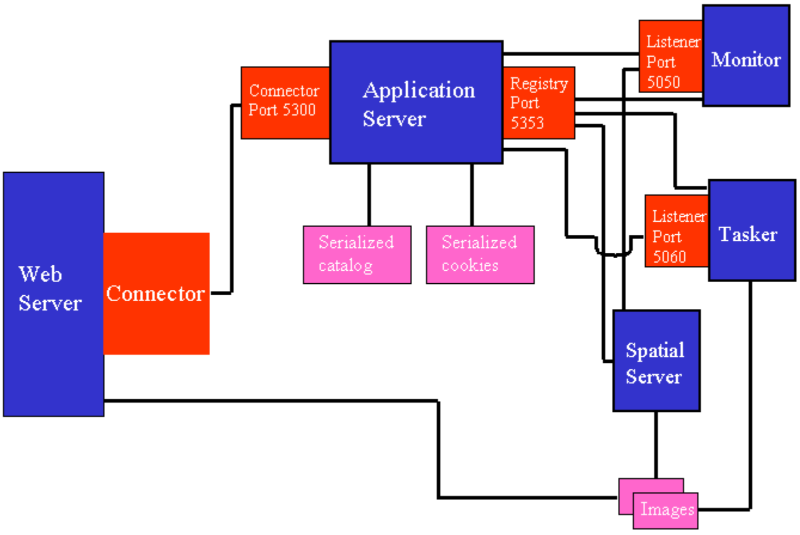


Fig. 3. ArcIMS Middleware.

Servlet Connector can serve requests for Feature as well as Image MapServices; therefore, it can interact with both HTML and Java viewers. The Servlet Connector supports the Basic and Digest authentication [3] for controlling user access. For content security, it supports S-HTTP, which is essentially HTTP over Secure Socket Layer (SSL) [7].

2.2.1.2 WMS Connector. The ArcIMS Web Map Server (WMS) Connector is a Java Servlet, which serves WMS requests that come from WMS compliant browsers/ clients. The Connector converts a WMS request into an ArcXML request and sends it to the Application Server. There are three types of WMS requests: *map*, *capability*, and *feature_info* [6]. A *map* request specifies the image format (Fig. 4) and the Connector responds back with an image of the map to the client. For a *capability* request, the connector returns back information that includes WMS version, supported formats for images, and text responses. The response to a *feature_info* request includes information about the queried geometric feature, which is returned as either XML or HTML text.

2.2.1.3 ActiveX Connector. The ActiveX Connector Object Model has built-in objects that allow developers to create custom ASP or Visual Basic® Applications. Each object in the Connector creates its own piece of ArcXML, which is sent to the Spatial Server through the Application Server. All client requests are handled on the server side, which has the following advantages: (1) the responses are lightweight and browser independent and (2) business logic code is not exposed to Web pages, which enhances security. Some basic functionalities available with the ActiveX Connector are panning, zooming, individual layer rendering, adding or removing layers from local data sources, and changing attribute data.

```
http://mapserver.esri.com/servlet/ArcImsWMT1?WMTVER=1
.0&REQUEST=map&LAYERS
=layer1&STYLES=style1&BBOX=10.1,20.5,12.4,23.6&SRS=EP
SG:4326&WIDTH=400&
HEIGHT=300&EXCEPTIONS=INIMAGE&FORMAT=JPEG&TRANSPARENT
= TRUE&BGCOLOR=0x45ffff&
```

Fig. 4. A sample WMS request to get a map.

2.2.1.4 ColdFusion Connector. The ColdFusion Connector allows developers to work with ArcIMS directly through a set of defined ColdFusion tags specific to GIS and mapping. The ColdFusion connector provides a design-time control inside Cold-Fusion Studio, which facilitates the use of GIS mapping and database functionality on Web sites due to the flexibility of adding tags. It requires significantly less programming expertise than conventional programming languages. Like ActiveX Connector, ColdFusion Connector also handles requests on the server side, which offers lightweight and browser-independent responses and enhanced security.

2.2.2 Application Server. The ArcIMS Application Server is central to the functioning of an ArcIMS site (Fig. 3); it can serve many Web server/connectors, and it provides brokering of Spatial Servers. When Monitors, Spatial Servers, and the Tasker start, they register themselves with the Application Server on the Registry Port. A client request comes through the Client Port. If it is a site-administration request, the Application Server executes it. If it needs to be processed by a Spatial Server, the Application Server forwards it to an available Server. For example, ArcXML client requests GET_IMAGE, GET_FEATURES, and GET_SERVICE_INFO need processing by the Spatial Server.

Application Server keeps information about Monitors, the Tasker, Spatial Servers, MapServices, and Virtual Servers (Section 2.2.5) of an ArcIMS site in its catalog. The catalog is serialized into a file. It also stores MapNotes and EditNotes as Cookies in another serialized file. If the Application Server is restarted, it will restore its state from the Catalog and Cookies serialized files.

2.2.3 Monitor. The ArcIMS Monitor process ensures that all Spatial Server instances on a machine keep running. It is also responsible for starting and stopping all the Spatial Servers on a machine. Upon starting, it registers with the Application Server. If the Application Server is not on, it tries to register at a regular interval until it succeeds. Monitor receives requests from the Application Server and Spatial Server via its Listener Port (Fig. 3). While an ArcIMS site can have more than one Monitor, a machine that runs Spatial Server(s) needs only one Monitor, because it can monitor all Spatial Servers on a machine. It sends pings to each Spatial Server at a regular interval; if a ping fails it launches a Spatial Server process for the crashed Spatial Server (Fig. 5).

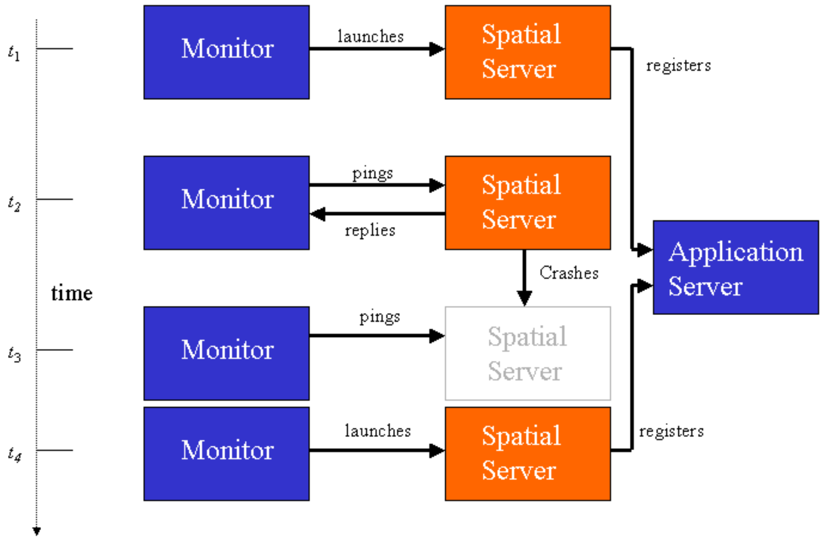


Fig. 5. Resurrection of a Spatial Server following its crash: At time t_1, Monitor launches a Spatial Server process, which registers with the Application Server. At t_2, Monitor pings the Spatial Server and receives a reply. After t_2 and before t_3, Spatial Server crashes. At t_3, when Monitor sends a ping, it does not receive a reply, because the Spatial Server is dead. Monitor immediately (i.e., at t_4) launches a Spatial Server, and the new Spatial Server registers with the Application Server.

2.2.4 Tasker. The ArcIMS Tasker process is responsible for cleaning up images that are generated for MapServices on Image Servers. After starting up, it registers with the Application Server. While creating a MapService on an Image server, an administrative user specifies an image output directory and a cleanup interval for the MapService. Application Server sends this information to the Tasker on the latter's Listener Port (Fig. 3). The Spatial Server writes image files in the output directory. A client views images in response Web pages. As soon as an image reaches an age equal to the cleanup interval, Tasker claims it. Since the image output directory needs to be accessed by Spatial Servers, Web browsers, and the Tasker, it must be shared between them. This cleaning ensures that the Web server does not run out of disk-space.

2.2.5 Virtual Server. A *Virtual Server* is a set of Server instances of a given type. A Server Instance on an ArcIMS site always belongs to one and only one Virtual Server. There are two types of public Servers: Feature and Image Servers. The member Servers of a Virtual Server can run inside one Spatial Server or many Spatial Servers.

They can reside on one machine or many machines. The following example shows some groupings for Virtual Server sets to illustrate the flexibility of Server grouping.

Example 1. Fig. 6 has four machines—*A*, *B*, *C*, and *D*. Machines *A*, *C*, and *D* run Spatial Servers $S1_A$, $S1_C$, and $S1_D$, respectively. Machine *B* runs two Spatial Servers $S1_B$ and $S2_B$. Spatial Servers have Server instances running inside them. For example, $S1_B$ has a Feature Server instance $F1_B$ and an Image Server instance $I1_B$. Virtual Servers VS_1 and VS_2 are groups of Feature Servers that run on different machines (Eqs. 1–2). Virtual Server VS_3 has only one Server instance of Image type (Eq. 3). Virtual Server VS_4 has Image Server instances from Spatial Servers *B*, *C*, and *D* (Eq. 4).

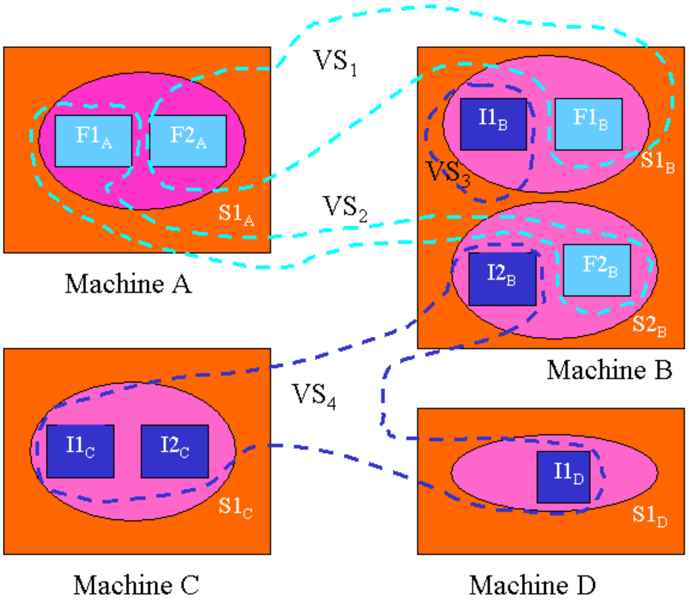


Fig. 6. Virtual Server groups.

$$VS_1= \{F2_{A,} F1_B\} \tag{1}$$

$$VS_2=\{F1_A, F2_B\} \tag{2}$$

$$VS_3=\{I1_B\} \tag{3}$$

$$VS_4=\{I2_B, I1_C, I2_C, I1_D\} \tag{4}$$

The concept of Virtual Servers helps in making sets on the basis of the security and reliability of MapServices. When a MapService is added to a Virtual Server, Application Server adds it to all Server instances that are members of the Virtual Server. When Application Server receives a request for a MapService, it looks for an available Server in the Virtual Server group on which this MapService was added and sends the request to the available Server. Thus Virtual Server is an important concept in ArcIMS, which is used for brokering of Spatial Servers.

2.3 Client Tier

The client tier, also known as the presentation tier, provides a user interface for the end user of the system to interact with the Distributed GIS. This interaction ranges from displaying a map in a Web page to managing MapServices on the server. In ArcIMS, this tier uses ArcXML as the language of interaction with all components within the system over HTTP transport. The significance of ArcXML in ArcIMS is that any application that can read and/or write XML can be a client to ArcIMS. Client tier uses the following applications (Sections 2.3.1–5) for accomplishing its tasks.

2.3.1 Author. The ArcIMS Author is a tool that enables an end user to author an electronic map for the Web. The tool allows a user to visually create a configuration file that can be used by the ArcIMS Administrator to create a MapService. Users can add spatial data, symbolize the data, set scale dependencies, and perform similar operations on a map window using a Java 2 application or applet running inside of the ArcIMS Manager Web pages.

2.3.2 Administrator. The ArcIMS Administrator is a Java 2 Application and applet that enables the ArcIMS Administrator to manage MapServices, Spatial and Virtual Servers, and Folders (i.e., EditNotes and MapNotes). Creating and managing an ArcIMS site is simplified using this tool.

2.3.3 Designer. The ArcIMS Designer is a tool that enables a user to generate a "ready-to-run" Web site using a wizard. The Designer is a Java 2 Application and applet that is run after a MapService is created. This enables a user to get up and running quickly with a Web site that can be modified to suit the project needs. The Designer outputs the three ArcIMS Viewers—HTML, Java Custom, and Java Standard.

2.3.4 Manager. The ArcIMS Manager is a Web site designed for enabling a user to quickly begin using and managing an ArcIMS site. The Manager consists of the ArcIMS Administrator, Author, and Designer. The Manager uses the following three-step approach to ArcIMS site management:

1. Author a map for the Web.
2. Create a MapService.
3. Design a Web site.
   All are available using the HTML interface.

2.3.5 HTML and Java Viewers. The three ArcIMS viewers—HTML, Java Custom, and Java Standard—use a Web site template to provide the functionality and graphic look of a Web site. Since these viewers are implemented as a template or a starting point, they can be modified to behave and appear according to the requirements of a project. The Java viewers are Java 2-based applets and require the

Java 2 plug-in to operate within the browser. The HTML Viewers use Dynamic HTML and JavaScript™ to interact with MapServices.

HTML viewer is the thinnest client and serves maps by downloading a raster image from the server for each request. Java viewer downloads vector features, as well as raster images, and renders them on the client screen. In the case of vector features, a Java viewer only gets a particular feature set if the viewer does not already have it. The HTML viewer is a lightweight client, whereas the Java viewer is a fast client for repeated requests on the same data set.

3 Advantages of the Architecture

The distributed architecture of ArcIMS offers the separation between clients and data sources across the Internet, which makes it feasible to host expensive, high-accuracy, and up-to-date data. Sections 3.1–5 briefly describe some advantages of the architecture.

3.1 Scalability

One can start hosting an ArcIMS site on a single machine that runs all components of the site and then add more machines to the site for distributing computing load as the traffic on the site increases. One can gradually add machines for additional Web servers to handle increasing client load (Fig. 7). Spatial Server processes on a machine can be added when waiting time increases. Machines to run Spatial Servers can be added to further reduce waiting time for requests. It is possible to do these changes without taking an ArcIMS site out of service.

3.2 Availability

In order to take a machine (let us say *A*), running Spatial Server, offline without affecting the availability of Servers, one can do the following: (1) add another machine (let us say *B*) to run Spatial Server, (2) associate Servers on *B* to existing Virtual Servers, (3) disassociate the Servers on *A*, or (4) take *A* offline without affecting the availability of MapServices.

In order to take a machine (let us say *C*) running Application Server offline, one can run Application Server on another machine (let us say *D*) with all the new Spatial Servers pointing to *D*. Following this the Connector can be set to point to *D*, and the Connector can be restarted to free machine *C* from the site.

To avoid disruptions in the availability of an ArcIMS site due to man-made or natural disasters, one can host mirror sites that serve the same set of MapServices using the same data.

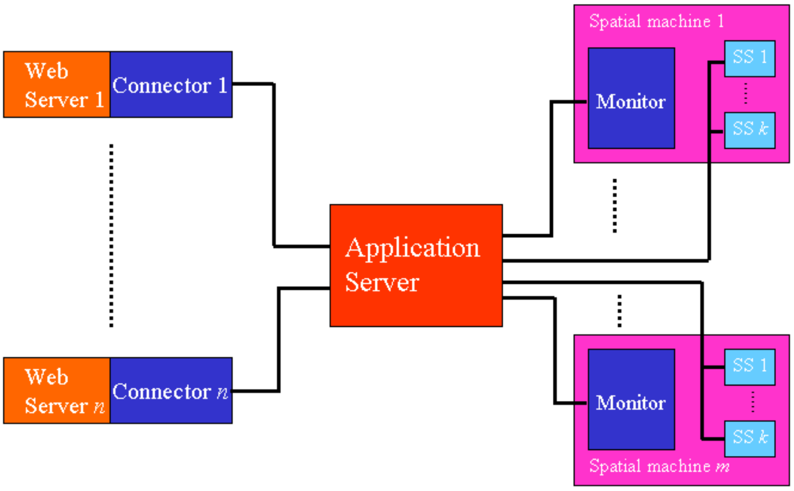


Fig. 7. Scalability of ArcIMS architecture

3.3 Customizability

ArcIMS architecture offers a high degree of customizability. One can customize servers to serve route maps, weather data, traffic data, and so on. Viewers and Connectors can be customized, which is evident from the family of Connectors and Viewers (Fig. 1).

3.4 Fail Over/Recovery

Application Server, Monitor, and Tasker run as background processes: on Windows NT® and Windows® 2000 platforms, they run as *Services*, and on UNIX® platforms they can be run as *daemons*. The advantage of running these processes in the background is that in the event of a power interruption, once the machines restart, the processes start on their own and resume their operations without human intervention.

3.5 Platform Independence

ArcIMS is supported on Windows NT, Windows 2000, Sun™ Solaris™, and IBM® AIX® operating systems, which is made possible by Java technology and the distributed architecture.

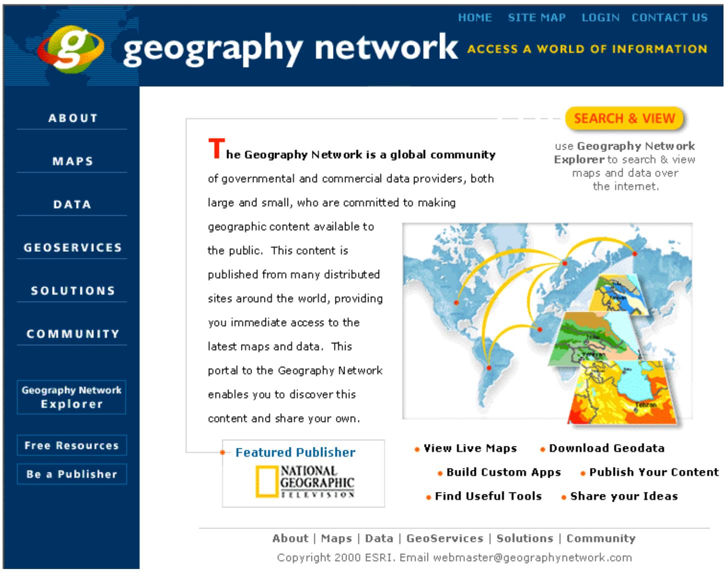
4 Example Web Sites

Several large Web sites have been implemented on the ArcIMS platform. These sites serve thousands or even millions of maps and queries a day. Two great examples of sites using ArcIMSare Geography Network^{SM} (Section 4.1) and National Geographic MapMachine (Section 4.2). These sites benefit from the distributed architecture of ArcIMS (Section 4.3).

4.1 Geography Network

Geography Network (Fig. 8) is a global network of geographic information users and providers. It provides the infrastructure needed to facilitate the sharing of geographic information between data providers, service providers, and users around the world. The Internet is used to deliver geographic content to the user's browser and desktop. Through Geography Network, one can access many types of geographic content including live maps, downloadable data, and more advanced services. Geography Network content is distributed at many locations around the world, providing users access to the latest information available directly from their sources.

The openness of ArcXML facilitated the creation of Geography Network. Through the Geography Network, data providers can register their ArcIMS site and MapServices, and users from around the world can then search, find, and use this data directly off their site. Users will also be able to purchase access to commercial MapServices and use it in their browser or clients, thus enabling users to have access to the latest commercial data without having to maintain it themselves. Currently the site generates about 250,000 maps per day.



The Geography Network is a global community of governmental and commercial data providers, both large and small, who are committed to making geographic content available to the public. This content is published from many distributed sites around the world, providing you immediate access to the latest maps and data. This portal to the Geography Network enables you to discover this content and share your own.

use Geography Network Explorer to search & view maps and data over the internet.

About | Maps | Data | GeoServices | Solutions | Community
Copyright 2000 ESRI. Email webmaster@geographynetwork.com

Fig. 8. Geography Network (http://www.geographynetwork.com/) uses ArcIMS to serve and share geographic information.

4.2 National Geographic

The National Geographic MapMachine (Fig. 9) was developed jointly by ESRI and National Geographic to serve a variety of maps to help people better visualize and understand the world around them. The site allows users to browse several different types of thematic data such as vegetation and population density anywhere in the world. It also provides maps of satellite imagery, political maps, and scanned plates from the National Geographic Atlas. Overall the site is serving about 100 different kinds of maps for any given area in the world. Currently, this site generates 300,000 maps per day.

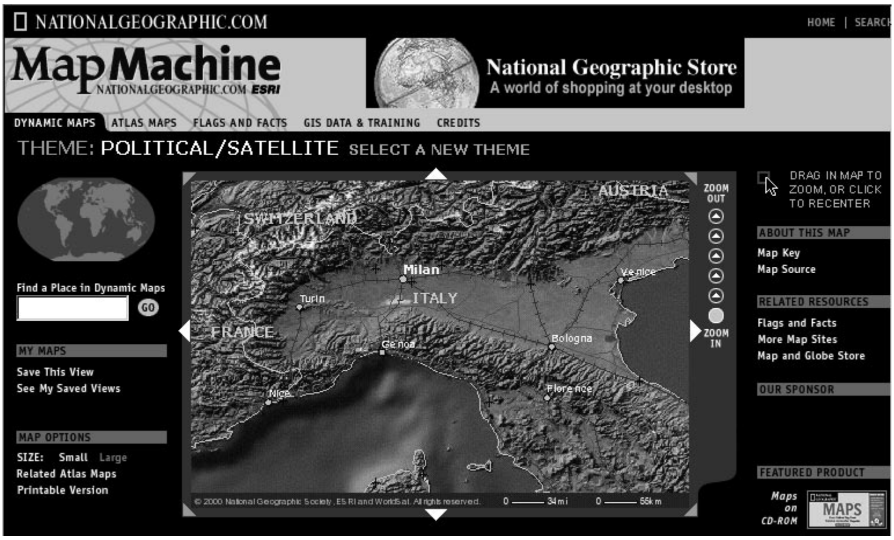


Fig. 9. National Geographic's (http://www.nationalgeographic.com/) MapMachine uses ArcIMS for serving geographic information.

4.3 The Architecture of the Site

Both of the above sites are running on a Sun Solaris platform on several different machines (Fig. 10). To ensure high availability, two different ArcIMS systems are used. The National Geographic site was built with the ColdFusion Connector and uses ColdFusion exclusively to make database accesses and ArcIMS calls. The Geography Network site was built partly with ColdFusion to do database queries and partly with JSP/Java Servlets to do the ArcIMS mapping calls. The data server runs ArcSDE and Oracle® and contains over 120 GB of spatial data, ranging from detailed street databases to thematic data to satellite imagery. The site also hosts some other Web sites in addition to the above two Web sites, generating more than 1,000,000 maps per day on the current hardware.

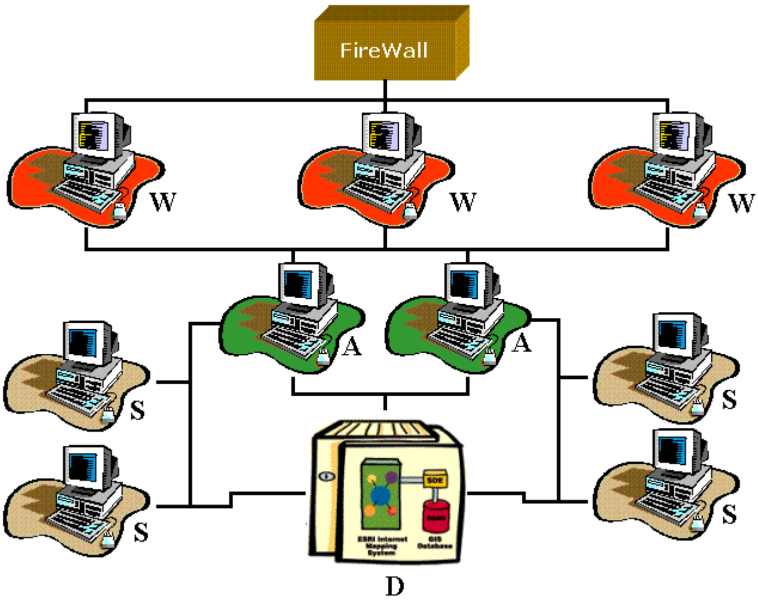


Fig. 10. The architecture of the ArcIMS Web site that supports Geography Network and National Geographic portals. *Web*, *Application*, *Spatial*, and *Data* servers are labeled as *W*, *A*, *S*, and *D*, respectively.

5 Considerations for the Next Version of ArcIMS

ArcIMS 3.1 has the same architecture as ArcIMS 3; the former is an enhanced and internationalized version of the latter. For the next version of ArcIMS higher than 3.1, we are considering the following technological and architectural points:

- Defining a modular procedure for writing business logic using *.Net* Framework (http://www.microsoft.com/net/) and Java 2 Enterprise Edition (J2EE) (http://java.sun.com/j2ee/) architecture. In the case of J2EE, we can leverage *Model-View-Control* (http://java.sun.com/j2ee/blueprints/designPatterns/MVC.html) design pattern. The View part will use Java Server Pages (JSPs) and JSP Tag-libraries; the Control part will use Servlets and JavaBean; and the Model part will use Enterprise JavaBean (EJB).
- Migrating the middleware-server communication from synchronous to asynchronous channel.
- Defining a modular procedure to integrate data access logic into business logic.
- Viewing business and data-access logic as a chain of objects with standard interfaces.
- Defining mechanisms to maintain session-based stateful or stateless communications. This mechanism will give ArcIMS site developers an option to make this

choice at deployment time or leave it to different clients (users/applications), similar to the use of cookies in Web browsers.

- Providing a standard directory of services, such as Universal Description Discovery and Integration (UDDI) (http://www-3.ibm.com/services/uddi/), for MapServices.
- Providing mechanisms for tool-generated business logic customization.
- Defining deployment classifications for the following:
  - Basic services (SHP files and ArcSDE database accesses)
  - Business services such as vehicle breakdown locator and help dispatcher. OnStar (http://www.onstar.com/) is an example of such a business service.
  - Portal services, such as Realtor.com and GeographyNetwork.com

6 Concluding Remarks

ArcIMS 3 has successfully addressed the needs of GIS users by offering a distributed architecture. It has scalability, maintainability, and fail over/recovery. It balances the load between Spatial Servers to provide fast response to the user. It allows collaborative mapping between users across the Internet. It can be used for viewing maps, making spatial and attribute queries, and sharing data in a secured environment. It is platform independent and runs on Windows and UNIX operating systems. It has been deployed on the sites that generate millions of maps per day. To further expand the scope of the advantages of ArcIMS 3, the architecture of the next version of ArcIMS will offer greater distributivity and customizability.

Acknowledgments. Doreen Brinkman, Vince Cogan, Cliff Davis, Bob Flores, Guchuan Liu, and Alexei Tikhonov participated in developing Client tier. Jamie Powell participated in ActiveX and ColdFusion connector development. Jose Thomas participated in the development of WMS connector. Kerry Coffin participated in developing Spatial Server.

References

1. Combs, J. (1996). "N-Tier vs.2-Tier Client-Server Architectures." http://www.ilt.com/AEDIS/Ntier/index.html.
2. ESRI (2000). ArcXML Programmer's Online Reference Guide. http://arconline.esri.com/arconline/documentation/ims_/WebHelp/ArcXMLGuide.htm.
3. Franks, J., P. Hallam–Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart (1999). *HTTP Authentication: Basic and Digest Access Authentication*, RFC 2617 (http://rfc.fh-koeln.de/rfc/html/rfc2617html).
4. Hunter, J., and W. Crawford (1998). *Java Servlet Programming*. O' Reilly, Sebastopol, California.
5. S. Morehouse (1989). "The Architecture of ARC/INFO." In *AUTO-CARTO 9, Ninth International Symposium on Computer-Assisted Cartography*, Baltimore, Maryland, USA, pages 266–277.
6. OpenGIS (2000). OpenGIS Web Map Server Interface Implementation Specification. Revision 1.0.0, April 19, 2000, http://www.opengis.org/techno/specs/00-028.pdf.
7. S. Thomas (2000). *SSL and TLS Essentials: Securing the Web*, Wiley, New York.