ArcGIS® Server Concepts and Terminology

An ESRI® Technical Paper • September 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-9326

ArcGIS Server Concepts and Terminology

An ESRI Technical Paper

Contents **Page**

ArcGIS Server Concepts and Terminology

Introduction

The following is a description of the ArcGIS® Server system. It is targeted for ESRI systems integration (SI) consultants and others who need to talk to customers about planning for ArcGIS Server. This material was taken from the *ArcGIS Server Developer's Guide* and other sources and summarized to provide a view of this information that is pertinent to system functionality and performance from a systems architecture perspective.

This is a developer's release and a developer's platform. Therefore, it is important to understand at least some of the concepts and terminology that pertain to this area. Although some of this material may seem rather technical, it will be useful to organizations considering ArcGIS Server. This document limits this information to areas that bear directly on the ESRI® system architecture design process and products.

To understand the concepts and terminology, it is helpful to have a rudimentary understanding of object-oriented programming and concepts. Words such as "object" and "instance" have meanings specific to the ArcGIS Server documentation.

This document contains the following five sections:

- ArcGIS Server Overview (from the developer's guide).
- ArcGIS Server Software Components (from SI)—This may be somewhat redundant, but it provides a slightly different view of these components.
- ArcGIS Server Glossary (from the developer's guide with comments).
- ArcGIS Server Application Performance and Tuning Topics (from the developer's guide).
- System Sizing Notes.

ArcGIS Server Overview

ArcGIS Server is a distributed system consisting of several components that can be distributed across multiple machines. Each component in the ArcGIS Server system plays a specific role in the process of managing, activating, deactivating, and load balancing the resources that are allocated to a given server object or set of server objects. The components of ArcGIS Server can be summarized as follows:

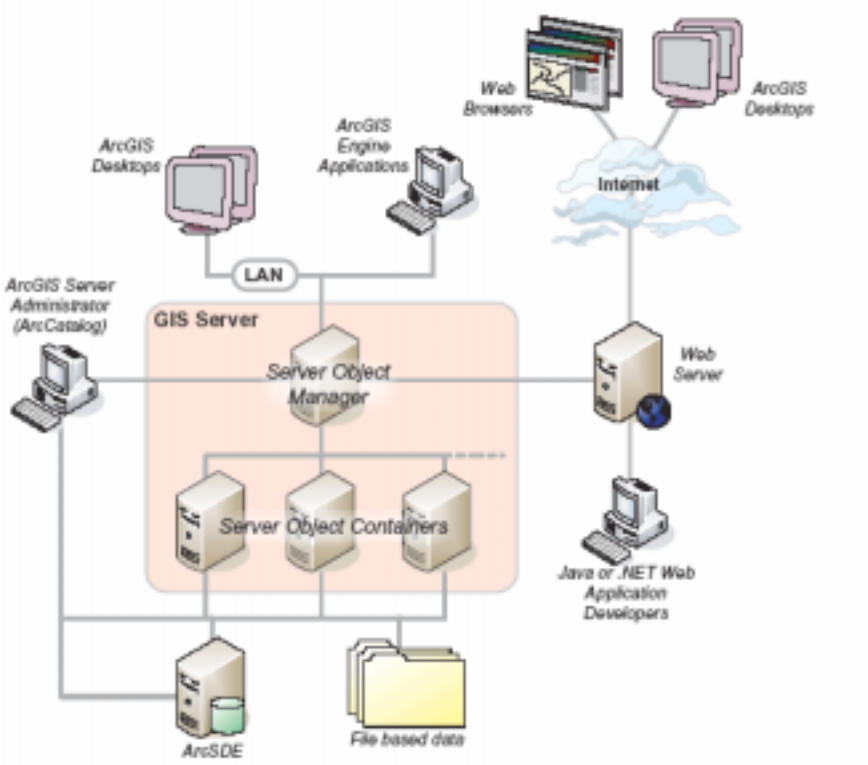
J-9326

- GIS server—Hosts and runs server objects. The geographic information system (GIS) server consists of a server object manager (SOM) and one or more server object containers (SOCs).
- Web server—Hosts Web applications and Web services that use the objects running in the GIS server.
- Web browser—Used to connect to Web applications running in the Web server.
- Desktop applications—Connect over HyperText Transfer Protocol (HTTP) to ArcGIS Web services running in the Web server or connect directly to GIS servers over a local area network (LAN) or wide area network (WAN).

The GIS Server

The GIS server, responsible for hosting and managing server objects, is the set of objects, applications, and services that makes it possible to run ArcObjects™ components on a server. Before describing the various aspects of the GIS server, the server object and the role of ArcObjects in ArcGIS Server will be defined.

The ArcGIS Server is a distributed system that consists of a server object manager, server containers, and clients to the server, such as desktop and Web applications.



A server object is a software object that manages and serves a GIS resource such as a map or a locator. For example, a server object named RedlandsMap may serve a map document of data for the city of Redlands, while the server object RedlandsGeocode may serve an address locator for geocoding addresses. ArcGIS Server objects are themselves ArcObjects components.

Server objects are managed and run within the GIS server. A server object may be preconfigured and preloaded in the server and can be shared between applications. Server applications make use of server objects and may also use other ArcObjects components that are installed on the GIS server.

The Server Object Manager

The GIS server is composed of a SOM, which is a Windows service running on a single machine, and SOCs, which run on one or more machines (container machines). The SOM manages the set of server objects that are distributed across one or more container machines. When an application makes a direct connection to a GIS server over a LAN or WAN, it is making a connection to the SOM, so the parameter that is provided for the connection to be made is the name or Internet Protocol address of the SOM machine (URL).

The Server Object Containers

The container machine or machines actually host the server objects that are managed by the SOM. Each container machine is capable of hosting multiple container processes. (A container process is a process in which one or more server objects is running.) Container processes are started and shut down by the SOM. The objects hosted within the container processes are ArcObjects components that are installed on the container machine as part of the installation of ArcGIS Server. All server objects have the potential to run on all container machines and are balanced equally across all container machines. Therefore, it is important that all container machines have access to the resources and data necessary to run each server object. It is also important to note that the GIS server assumes that all container machines are configured equally, such that they are all capable of hosting the same number of server objects. Server object resources and data are discussed in more detail in the next section.

The Server Directories

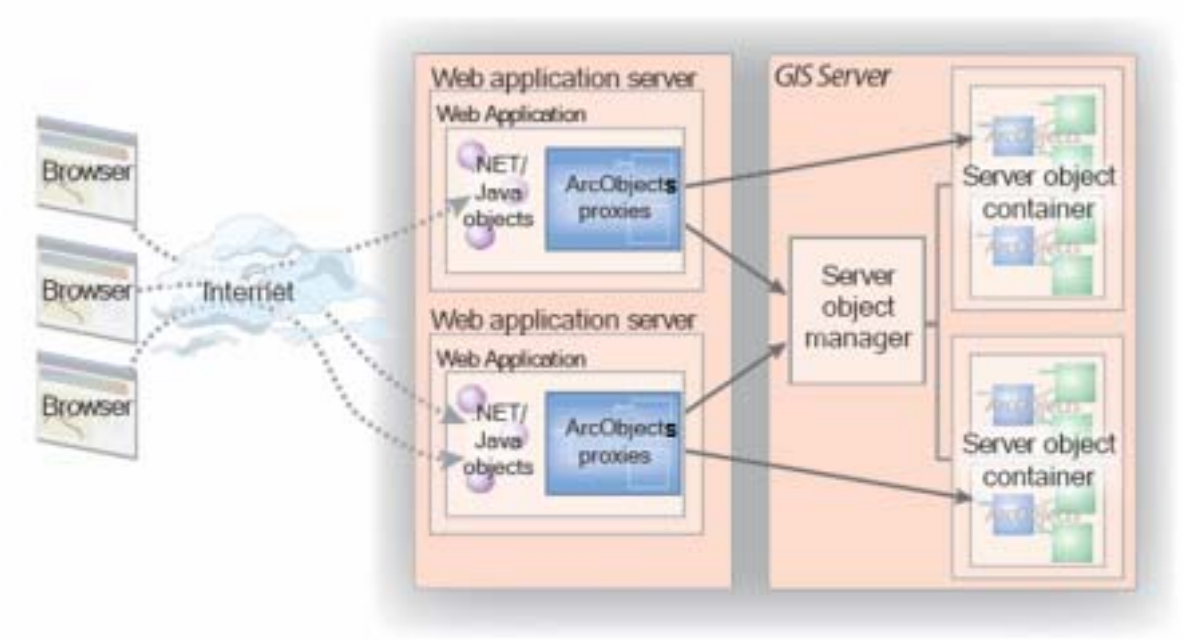
A server directory is a location on a file system. The GIS server is configured to clean up any files it writes to a server directory. By definition, a server directory can be written to by all container machines. The GIS server hosts and manages server objects and other ArcObjects components for use in applications. In many cases, the use of those objects requires writing output to files. For example, when a map server object draws a map, it writes images to disk on the Web server machine. Other applications may write their own data; for example, an application that checks out data from a geodatabase may write the checkout personal geodatabase to disk on the server.

Typically, these files are transient and need only be available to the application for a short time, for example, the time for the application to draw the map or the time required to download the checkout database. As applications do their work and write out data, these files can accumulate quickly. The GIS server spawns a special SOC process that will automatically clean up its output if that output is written to a server directory.

A server directory can be configured such that files created by the GIS server in it are cleaned based on either file age or time since they were last accessed. The maximum file age is a property of a server directory. All files created by the GIS server that are older than the maximum age or have not been accessed during the time defined by the maximum age are automatically cleaned up by the GIS server.

In addition to supporting output of image files to the file system, ArcGIS Server also supports streaming of images from the SOC (via the Web server) to the user. This is useful since the streaming occurs over port 80, which avoids having to open additional TCP ports for drive mapping from the SOC to the Web server.

The Web Server



The Web server hosts server applications and Web services written using the ArcGIS Server application program interface (API). These server applications use the ArcGIS Server API to connect to a SOM, make use of server objects, and create other ArcObjects for use in their applications. These Web services and Web applications can be written using the ArcGIS Server Application Development Framework (ADF™), which is available for both .NET and Java developers. Examples of Web applications include mapping applications, disconnected editing applications, and any other application that makes use of ArcObjects and is appropriate for Web browsers.

Examples of Web services include those for exposing map and geocode server objects that desktop GIS users can connect to and consume over the Internet. It is possible for users to create their own native .NET or Java Web services whose parameters are not ArcObjects types but do perform a specific GIS function. For example, it is possible to write a Web service called FindNearestHospital that accepts *x,y* coordinates as input and

returns an application-defined Hospital object that has properties such as the address, name, and number of beds.

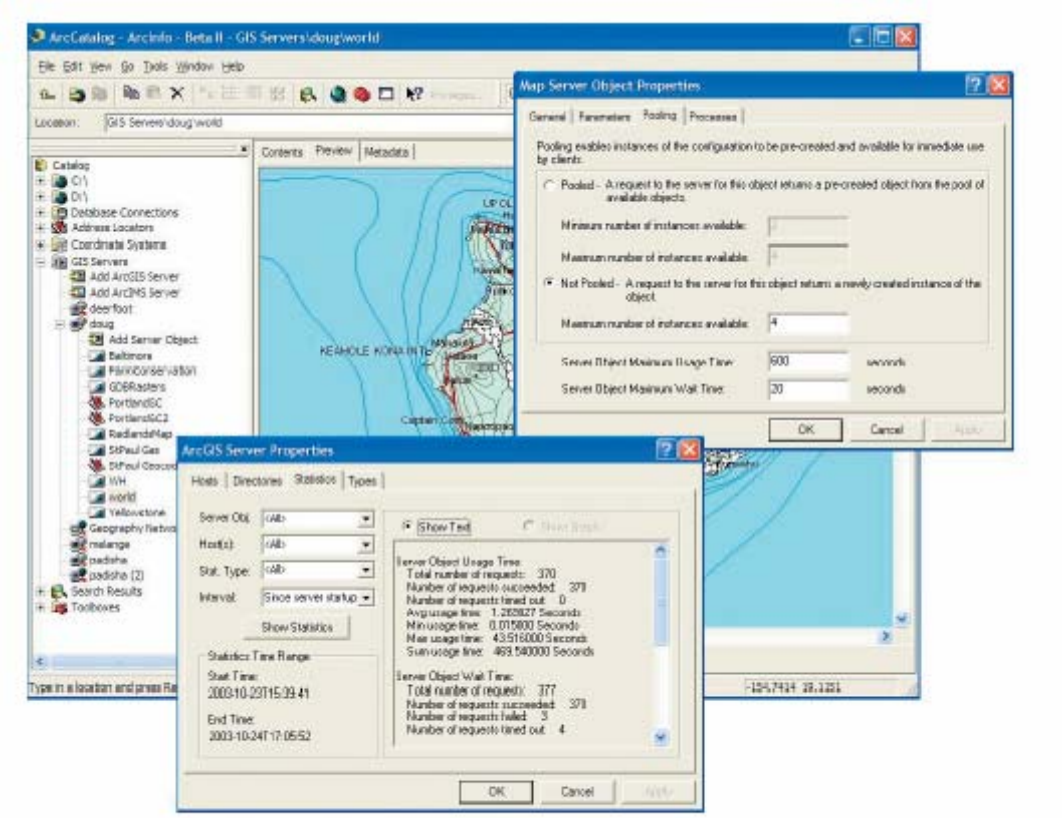
Web applications connect to GIS servers within their organization over the LAN. In this sense, the Web application or Web service is a client of the GIS server. Users connect to Web applications and Web services over the Internet or Intranet, but all the Web applications' logic runs in the Web server and sends HyperText Markup Language (HTML) to the browser client. The Web application itself makes use of objects and functionality running within the GIS server. This allows the development of Web applications to make use of ArcObjects in the server as would a desktop application connecting to the GIS server in client/server mode over the LAN or WAN.

As users interact with their browsers, they make requests to the Web application, which in turn makes requests on the SOM. The SOM returns a proxy to a server object or server objects that are running within the GIS server. The Web application uses the proxy to work with the object as if it existed in the Web applications process, but all execution happens on the GIS server.

ArcGIS Desktop Applications

Users can connect to ArcGIS Server using ArcGIS Desktop applications to make use of map and geocode server objects running in the server. Users can use ArcCatalog™ to connect to a GIS server directly on the LAN or WAN. They can also specify the URL of a Web service catalog to indirectly connect to a GIS server over the Internet to make use of map and geocode server objects exposed by that Web service catalog. In addition, the set of server objects and their properties are managed by the GIS server administrator using ArcCatalog. Administrators can connect to the GIS server over the LAN/WAN and use ArcCatalog to add and remove map and geocode server objects. They can also configure how server objects should be run including the set of container machines that are available for the server and the directories on the server that they can use to write any output.

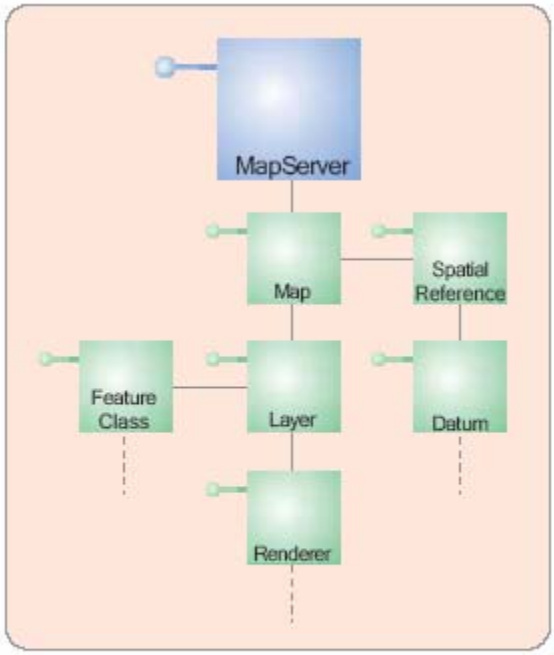
J-9326



Users can connect to ArcGIS Server using ArcGIS Desktop applications to consume and administer server objects.

As described earlier, a server object is a software object that manages and serves a GIS resource such as a map or a locator. Server objects are managed by the GIS server and run within processes on container machines. A server object is simply a coarse-grained ArcObjects component, that is, a high-level object that simplifies the programming model for doing certain operations and hides the fine-grained ArcObjects components that do the work. These coarse-grained objects allow clients to perform large units of work, such as drawing a map or geocoding a table of addresses, using a single method call. These coarse-grained objects use the finer-grained ArcObjects components on the server to draw the map and geocode the addresses.

J-9326



A server object is a coarse-grained ArcObjects component that has other ArcObjects components associated with it.

Server objects also have Simple Object Access Protocol (SOAP) interfaces for handling SOAP requests to execute methods and returning results as SOAP responses. This support for SOAP request handling makes it possible to expose server objects as Web services that can be consumed by clients across the Internet.

ArcGIS 9 Server includes two coarse-grained server objects: the GeocodeServer and the MapServer server objects. Note that a server object also has other associated objects that a developer can get to and make use of, for example, a developer working with a MapServer object can get to the map and layer objects associated with that map. These are the same map and layer objects that an ArcGIS Desktop or Engine developer would work with, except they live in the server.

A server object, unlike other ArcObjects components, can be preconfigured by a GIS server administrator. Once these server objects are preconfigured, they can be used by developers and end users who can connect to the server through ArcGIS Desktop applications. When a server object is configured by the server administrator to run in a GIS server, the following configuration aspects of the server object need to be specified:

- Name of the server object
- Type of server object
- The initialization data and parameters for the server object

- Whether or not the object is pooled
- The minimum and maximum number of instances that can be running
- How long a client can wait for and use a server object
- The isolation level of the server object
- Whether the object is recycled

Each of these aspects of server objects will be described in more detail.

ArcGIS Server Software Components

- Third party standard Web server software (Web servers)
- Custom applications (customer or third party)
- Application Development Framework
  - Runtime system (ArcGIS Server Runtime)
  - ArcGIS Server Application Developer Kit
- Server object manager
  - Manages service configurations, which are basically rules for creating SOCs and object instances on container machines (servers). A single SOM can manage multiple SOCs on multiple container machines. A single ArcGIS Server has one SOM.
  - Creates SOCs with the minimum number of object instances as set by the service configuration parameters.
  - Accepts new connection requests from clients and routes them to an appropriate SOC.
  - Creates new object instances in the server object container as needed, as service demands increase, up to the maximum number as specified in the service configuration.
  - Controls load balancing for ArcGIS Server by creating new instances in container machines with the least number of existing instances utilizing parameters defined in the context configuration.
  - Controls recycling of objects through managing object definitions.
  - Contains contexts, a set of configuration parameters that determine how load balancing will occur. Contexts define on which container machine new SOCs and instances will be created.
  - Performs load balancing between container machines through object instance creation.

- Server object container
  - Each SOC is a single Windows process on which one or more server object instances can execute.
  - There are two ways that SOCs can be configured to manage this work—high isolation and low isolation.
    - High isolation—Objects do not share a container process; each object has a dedicated container process (SOC). This must be used if the object is not thread safe.
    - Low isolation—The SOC process can support multiple objects. This is used when objects are thread safe.
  - Serves ArcObjects, a set of GIS specific executables that provides various spatial services including mapping and geocoding.
  - Serves ESRI provided server objects (MapServer and GeocodeServer at version 9.0).
  - Serves custom server objects.
- Data source (ArcSDE®)
  - Client environments
    - Web browsers (Web clients)
    - ArcGIS Desktop applications (either Web- or Network-based clients)
    - Custom desktop applications
  - Third party software
    - Web server
    - Database software (database management system)
- Server platforms
  - Web application server
    - Hosts the Web server
    - Hosts ArcGIS Server Runtime
    - Hosts business and application logic for Web applications
  - GIS server
    - SOM—Can be in separate tier or on same tier as container machine.
    - Container machine—Hosts ArcObjects and server objects.

        - ♦ Can also host Web or network services (these may not be load balanced).
    - Data server—Hosts ArcSDE and relational database.
- ArcGIS Server service options
    - ArcGIS Server services (network GIS desktop clients)
    - Web services (Web GIS desktop clients)
    - Web applications (Web browser clients)

ArcGIS Server Glossary

ADF—Application Development Framework—The set of custom Web controls and templates that can be used to build Web applications that communicate with a GIS server. ArcGIS Server includes an ADF for both .NET and Java.

ADF Runtime—The components required to run an application built with ADF. See also ADF—Application Development Framework.

ArcGIS Server Web Service—A Web service processed and executed from within an ArcGIS Server. Each Web service is a distinct HTTP endpoint (URL). Administrators can expose MapServer and GeocodeServer objects as generic ArcGIS Server Web services for access across the Internet. Web services can also be exposed to ArcCatalog users.

ArcObjects—A library of software components that makes up the foundation of ArcGIS. ArcGIS Desktop, ArcGIS Engine, and ArcGIS Server are all built on top of the ArcObjects libraries.

Container Process—A process in which one or more server objects are running. Container processes run on SOC machines and are started and shut down by the SOM. This is another name for a SOC.

GeocodeServer—An ArcGIS Server software component that provides programmatic access to an address locator and performs single and batch address matching. It is designed for use in building Web services and Web applications using ArcGIS Server. This is one of two coarse-grained ESRI software-provided server object types that can be hosted on ArcGIS Server. (MapServer is the other.)

GIS Server—The components of ArcGIS Server that host and run server objects. A GIS server consists of a server object manager and one or more server object containers.

MapServer—An ArcGIS Server software component that provides programmatic access to the contents of a map document on disk and creates images of the map contents based on user requests. It is designed for use in building map-based Web services and Web applications using ArcGIS Server. This is one of two coarse-grained ESRI software-provided server object types that can be hosted on ArcGIS Server. (GeocodeServer is the other.)

Object Pooling—The process of precreating a collection of instances of classes such that the instances can be shared between multiple application sessions at the request level. Pooling objects allows the separation of potentially costly initialization and acquisition of resources from the actual work the object does. Pooled objects are used in a stateless manner. Nonpooled objects are created for each application session and are destroyed when they are released. This has overhead but may be necessary due to the requirements of the application. Applications that change objects (such as adding or changing a map layer) must be nonpooled. Object pooling can only be used for read-only objects. Editing applications will require nonpooled objects. The overhead of nonpooled objects will be in proportion to the amount of resources (file, database connections, memory, etc.) required to initialize the objects.

Recycling—The process by which objects in an object pool are replaced by new instances of objects. Recycling allows objects that have become unusable to be destroyed and replaced with fresh server objects and to reclaim resources taken up by stale server objects. Recycling will need to be more frequent if pooled objects are used and if applications are not written correctly. ArcGIS Server does not enforce proper object instance sharing, and it is fairly easy to create applications that do not "share" properly. These applications will corrupt the objects they touch and cause other users/applications to fail or obtain unpredictable and incorrect results. Recycling is a way of creating fresh instances of objects. In a nonpooled environment, recycling has no real meaning—objects are already recycled each time they are used.

Server Context—A space on the GIS server where a server object and its associated objects are running. A server context runs within a server object container process. A developer gets a reference to a server object through the server object's server context and can create other objects within a server object's context.

Server Object—A coarse-grained ArcObjects component that manages and serves a GIS resource such as a map or a locator. A server object is a high-level object that simplifies the programming model for performing certain operations and hides the fine-grained ArcObjects components that do the work. Server objects also have SOAP interfaces, which makes it possible to expose server objects as Web services that can be consumed by clients across the Internet. ArcObjects components are fine-grained objects. They can also be accessed from applications but will require more complex applications either at the GIS server level or at the Web server level. Server objects can be written by customers, provided by third parties, or provided by ESRI. In the 9.0 release, ArcGIS Server comes with two server objects: MapServer and GeocodeServer. With custom server objects, the performance profile will be somewhat difficult to predict due to the flexibility of the platform, which allows an enormous variety in complexity of applications.

Server Object Isolation—Describes whether server objects share processes with other server objects. Server objects with high isolation run dedicated processes, whereas server objects with low isolation share processes with other server objects of the same type. "Process" is a Windows term, meaning a single program/task as would be seen in the Task Manager. SOC is basically a single process on which one or more server objects execute. In high isolation, a SOC services only one server object. This must be used for

programs that are not thread safe. Running server objects in high isolation will use more server resources.

Server Object Type—Defines what a server object's initialization parameters are and which methods and properties it exposes to developers. At ArcGIS 9, there are two server object types: MapServer and GeocodeServer.

SOAP—Simple Object Access Protocol—An XML-based protocol developed by Microsoft/Lotus/IBM for exchanging information between peers in a decentralized, distributed environment. SOAP allows programs on different computers to communicate independently of an operating system or platform by using the World Wide Web's HTTP and XML as the basis of information exchange. SOAP is now a W3C specification.

SOC—Server Object Container—A process in which one or more server objects is running. SOC processes are started and shut down by the SOM. The SOC processes run on the GIS server's container machines. Each container machine is capable of hosting multiple SOC processes. See also SOM—Server Object Manager.

SOM—Server Object Manager—A Windows service that manages the set of server objects that are distributed across one or more server object container machines. When an application makes a connection to ArcGIS Server over a LAN, it is making a connection to the SOM. See also SOC—Server Object Container.

Web Server—A computer that manages Web documents, Web applications, and Web services and makes them available worldwide.

Web Service—A software component accessible over the World Wide Web for use in other applications. Web services are built using industry standards, such as XML and SOAP, and thus are not dependent on any particular operating system or programming language, allowing access through a wide range of applications.

Web Service Catalog—A collection of ArcGIS Server Web services. A Web service catalog is itself a Web service with a distinct endpoint (URL) and can be queried to obtain the catalog's list of Web services and their URLs. See also ArcGIS Server Web Service.

ArcGIS Server Application Performance and Tuning Topics

Application State and Scalability

The question of stateful versus stateless use of the GIS server is central to the scalability of the user's application. An application is more scalable than another application if it can support a larger number of users with the same amount of computer resources. The keys to scalability are

- Making stateless use of the GIS server

- Using pooled server objects
- Minimizing the time an application holds on to a server object by releasing server objects as soon as possible and not relying on .NET or Java garbage collection

Using the above criteria, it is clear that stateless or shallowly stateful applications can make use of object pooling and, therefore, are more scalable than deeply stateful applications. The question of stateful versus stateless use of the GIS server will be critical in designing an application.

ArcGIS Server Performance and Tuning

As discussed in the previous sections, developing ArcGIS Server applications is all about remotely programming ArcObjects. As such, the server API is wide and includes a large number of ArcObjects components organized into a series of object libraries. Developers can build any kind of application using the server API, so it is possible to build high-performance, scalable applications as well as extremely slow applications that do not scale. This section will discuss strategies for avoiding the latter, but it will ultimately be in the hands of the developer and the GIS server administrator to work together to design applications that can perform and scale given available hardware resources.

ArcGIS Server and Fine-Grained ArcObjects Components

ArcGIS Server uses the same ArcObjects components that ArcGIS Engine and ArcGIS Desktop use; therefore, if a server application includes GIS functionality that performs poorly in an ArcGIS Engine or ArcGIS Desktop deployment, that same functionality will likely perform poorly in server deployments. Conversely, if that GIS functionality performs well in an ArcGIS Engine or Desktop deployment, it will also perform well in server deployments if the application is properly tuned with respect to how it makes use of ArcObjects components in the server.

Both coarse-grained calls to remote ArcObjects components, such as the methods on the *MapServer* and *GeocodeServer*, and fine-grained calls to remote ArcObjects components, such as looping through all the vertices of a polygon, are exposed through the ArcGIS Server API and can be used in the user's application. However, it is important to note that when making a call against an object running in the server from a Web application, that call is made across processes. The Web server is running in one process, while the object is running in another. Calls to objects across processes are significantly slower than calls to objects in the same process. It is also likely that the Web application is running on a Web server that is actually a different machine from the machine on which the object is running, so the calls are not only cross process but also cross machine. This performance difference on the scale of a single call or tens of calls is not significant in terms of the overall performance of the application. However, if the application is making thousands of fine-grained ArcObjects calls to the server across process or machine, there can be a significant performance impact on the application. Minimize the number of round-trips to the GIS server from the application by minimizing the number of fine-grained calls to remote objects.

If the nature of the application demands a lot of fine-grained ArcObjects work, one strategy for supporting such functionality while keeping remote calls to a minimum is to extend the GIS server with application specific utility COM objects that can be developed in Visual Basic, C++, or .NET.

Performance Tuning Summary

- Minimize the number of fine-grained calls to remote ArcObjects components.
- If large numbers of fine-grained ArcObjects calls are necessary, think about extending the server by creating COM objects that move the fine-grained ArcObjects usage into the server.
- Do not allow application users to perform queries that result in large numbers of rows, and limit the number of results from queries that you process.
- Work with your GIS server administrator to ensure that the built-in limits for queries and output for a particular MapServer or GeocodeServer are configured appropriately for the application.

Object Pooling

A server object may be configured to be pooled or nonpooled. Nonpooled server objects are created new for each application use and destroyed when released by the application to the server. The creation of the object includes creating the object and loading any initialization data such as the map document associated with a MapServer server object. Each user of a server application who makes use of a nonpooled server object requires an instance of that object dedicated to his or her application.

The number of users on the system at any one time has a 1:1 correlation with the number of running server object instances, so the number of concurrent users the GIS server can support is equal to the number of server objects that it can support effectively at any one time. When configuring a server object to be nonpooled, the maximum number of instances can be limited by specifying it as a property of the configuration. Once the maximum number of instances has been reached (i.e., the maximum number of concurrent users of that server object), additional users will be queued until the number of users drops below the maximum.

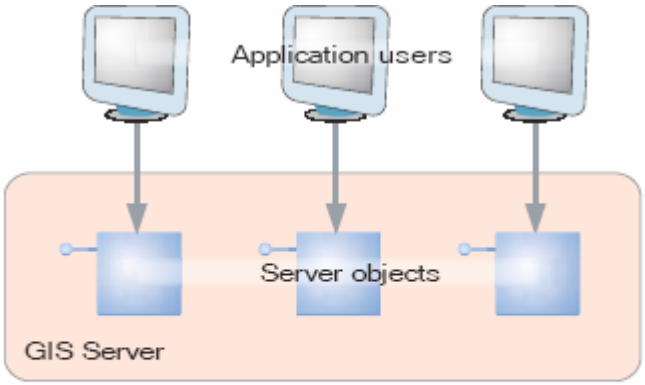
ArcGIS Server allows the user to pool instances of server objects so they can be shared between multiple application sessions at the per-request level. This allows the ability to support more users with fewer resources allocated to a particular server object. When a server object is configured to be pooled, the minimum and maximum number of instances must be specified as properties of the server object configuration. When the server object is started, the GIS server will precreate and initialize the minimum number of server objects. When an application asks the server object manager for an instance of that server object, it will get a reference to one of the preloaded server objects in the pool.

The advantages of pooling server objects include

- Separating the potentially expensive initialization of a server object from the actual work that the object performs for each client
- Sharing the cost of expensive initialization and acquisition of resources, such as database connections, across all clients
- Precreating objects at server object manager startup before any client requests come in

- Administratively configuring pooling to take best advantage of available hardware resources

Applications that use pooled server objects keep their reference on that object for the duration of their request (e.g., draw map, identify feature, geocode address), then return the object to the server. When the object is returned to the server, it is available for use by another user's request. Users of such an application may be working with a number of different instances of a server object in the pool as they interact with the application, all of which is transparent to the users. If there are more simultaneous requests on a pooled server object than the minimum, new instances of the server object will be created until the maximum number of instances is reached, at which point the user is queued until objects in the pool become free.



The number of non-pooled server objects running in the GIS server is 1:1 with the number of application users making use of that server object.

Pooling server objects allows the GIS server to support more users. Because applications can share a pool of server objects, the number of concurrent users on the system is greater than the number of server objects that the GIS server can effectively support at one time. Nonpooled and pooled server objects support different types of server applications. Applications are expected to make stateless use of pooled server objects, meaning they do not make changes to the server object when they are using it, and they are released back to the pool in a timely manner when the request has been processed.

For example, a stateless mapping application that wants to draw a certain extent of a MapServer's map will get a reference to an instance of a MapServer server object from the pool, execute a method on the MapServer to draw the map, and then release it back to the pool. The next time the application needs to draw the map, this is repeated. Each draw of the map may use a different instance of the pooled server object; therefore, each pooled object must be the same (have the same set of layers, the same renderer for each layer, and so on). If the state of one of the pooled objects is changed (e.g., a new layer is added or a layer's renderer is changed), then as a user pans and zooms around the map, he or she will see inconsistent results.

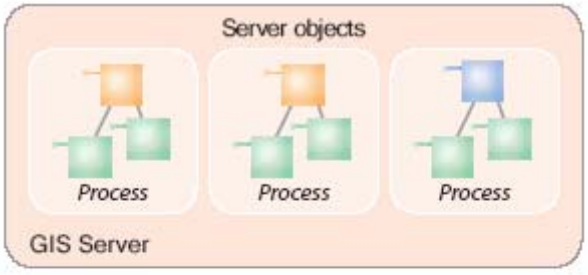
Nonpooled server objects are created for each application session that uses it. Since server objects can take time to initialize, the application that makes use of a nonpooled object typically holds a reference to the server object for the duration of the application's session. Since the server object is destroyed when it is returned to the server, the application is free to change any aspect of the server object's state.

Server Object Isolation

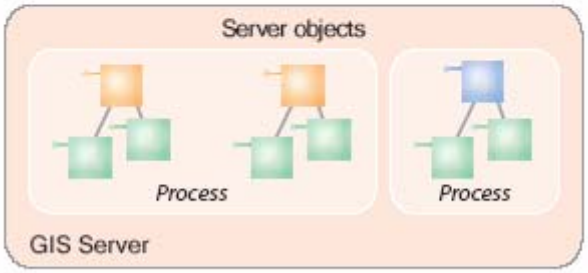
Server objects run within processes on the container machines. Server objects can be configured such that they run in a dedicated process on the server, or they can be configured to run in processes they share with other server objects. How they share processes is referred to as their isolation level. Server objects with high isolation do not share a process with other server objects. Each instance of a server object with high isolation has its own dedicated process on the server. Server objects with low isolation can share processes with other server objects of the same type.

Up to four server objects can share the same process. When more than four server objects of a particular type (e.g., four RedlandsMap server objects) are created, an additional process is started for the next four server objects, and so on. As server objects are created and destroyed, they will vacate and fill spaces in these running processes. Instances of server objects whose isolation level is high require more resources on the server to run, as they require dedicated processes. Since instances of server objects with low isolation can share processes, they make more efficient use of server resources. However, isolation does have its benefits: since server objects with high isolation do not share processes, if an error occurs on the object causing its process to shut down or crash, it will not affect other server objects. However, if a server object is sharing its process with other server objects and the process is shut down or crashes, all the server objects in that process will be destroyed.

J-9326



Server objects with high isolation run in dedicated processes on the GIS server.



Server objects with low isolation can share processes with other server objects of the same type.

Server Object Recycling

Recycling allows server objects that have become unusable to be destroyed and replaced with fresh server objects; recycling also reclaims resources taken up by stale server objects. This process allows users to keep the pool of server objects fresh and cycle out stale or unusable server objects. Pooled server objects are typically shared between multiple applications and users of those applications. Through reuse, a number of occurrences can make a server object unavailable for use by applications. For example, an application may incorrectly modify a server object's state or incorrectly hold a reference to a server object, making it unavailable to other applications or sessions. In some cases, server objects may become corrupted and unusable.

Nonpooled server objects whose isolation level is low can also be recycled. This recycling will shut down and restart the processes in which nonpooled objects are started up and run.

In each case, recycling occurs as a background process on the server. The time between recycling events is called the recycling interval. A server object's recycling interval can be configured by the administrator. During recycling, instances of server objects in use by clients are not recycled until released, so recycling occurs without interrupting the user of a server object.

System Sizing Notes

A single GIS server can host a variety of objects and applications. It can host a Web service catalog; other ArcGIS Server Web services; server applications written by third parties or by the customer; server objects that are provided by the customer, ESRI, or third parties; and ArcObjects. It can be architected such that most of the load is on the Web application's tier, on the GIS server tier, or a combination of both. Providing sizing and capacity information about this system may be challenging as a result, as fairly simple models may not be pertinent to many situations.

Best practice is to create a prototype application and obtain performance metrics prior to performing a production system sizing exercise. Also, the case for ArcGIS Server needs to be made as an overall cost-to-build and cost-to-own analysis, along with the value of the focused functionality that ArcGIS Server can provide. Clearly, ArcGIS Server out of the box will not perform as well as ArcIMS® or ArcGIS Desktop on Window Terminal Service (WTS)/Citrix for the same functionality. So there must be other reasons for implementing it.

ArcGIS Server platforms are generally implemented on dual processor servers, adding servers (scaling out) as needed to meet the total performance and capacity needs of the organization. ArcGIS Server supports load balancing and high availability across multiple platforms, which means it is a scalable architecture. For additional information on ArcGIS Server platform configurations and sizing, refer to Section 4.8 in the *Systems Architecture Design* white paper produced by the Systems Integration Group. This is available at http://www.esri.com/library/whitepapers/pdfs/sysdesig.pdf.

Conclusion

ArcGIS Server is a powerful and flexible platform for providing a variety of spatial data and services to GIS users and applications. It provides the ability for organizations to implement server-based spatial functionality for focused applications utilizing the rich functionality of ArcObjects. Building robust, scalable applications is not a simple task, and proper application design is required to support systems that will perform well. As mentioned earlier, production system configuration and sizing are best performed after the applications are built and performance metrics are collected on those specific applications.