An ESRI® White Paper • June 2007

Geospatial Service-Oriented Architecture (SOA)

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

Geospatial Service-Oriented Architecture (SOA)

An ESRI White Paper

Contents **Page**

Appendixes

Geospatial Service-Oriented Architecture (SOA)

Executive Summary

Vendors around the world proclaim their alignment with the concepts of service-oriented architecture (SOA), but how can customers validate these claims? There are a variety of opinions about the key components of a service-oriented architecture, which leads to confusion for many customers and many questions:

- Is an enterprise service bus required?
- Does providing a Web service to different types of clients make an SOA?
- Is SOA an Open-Source initiative or Java-only solution?
- How do I manage Web service and geospatial metadata?
- Does rich client software fit into an enterprise SOA?
- Does SOA require a portal with portlets or webparts?
- How do ESRI® geospatial components interoperate with the above components?

The focus of this white paper is to first describe the basic components that support a geospatial SOA in a vendor-neutral framework, then describe how ESRI components fit into this framework.

1.0 Introduction

Service-oriented architecture is beginning to fundamentally change the way in which enterprises deploy information technology (IT) to support business operations.

Historically, IT departments have held an application-centric view of the world. Most of their budgets have been allocated to the purchase, deployment, and maintenance of individual applications. In more recent times, a growing percentage of the budget has been allocated to integration projects in an attempt to deliver broader and more seamless support for business processes. However, these projects have often fallen short of their promise due to the inherent rigidity of hardwired integration: any change to a single application must be propagated across the entire integration, creating an expensive and change-averse infrastructure.

SOA enables IT departments to make the transition from an application-centric view of the world to a process centric one. IT departments now have the freedom to combine business services from multiple applications to deliver true end-to-end support for business processes. IT can upgrade or change applications without impacting other applications in the SOA by utilizing integration mechanisms such as Web services.

ESRI has responded to this fundamental shift in the technology landscape by making ArcGIS® 9.2 SOA enabled, with full Web service integration. This allows customers to readily expose ArcGIS 9.2 standards-based functionality to other applications and interfaces, thus dramatically improving its value and return on investment (ROI).

SOA is about empowering customers to utilize best-of-breed components that do not lock them into a single vendor solution. ESRI ensures support of the right components for the right job by providing choices for

- Database (Microsoft SQL, Oracle, Informix, DB2)
- Operating systems (Windows, Linux, AIX, Solaris)
- Development platforms (.NET and Java)
- IT and GIS standards (World Wide Web Consortium [W3C], Organization for the Advancement of Structured Information Standards [OASIS], ISO, Open Geospatial Consortium, Inc. [OGC])

1.1 Document Purpose

The purpose of this document is to provide a

- Common definition of service-oriented architecture and its components
- Mapping of how ESRI components may be utilized in a service-oriented architecture

1.2 Structure

This document is divided into three sections:

- **Introduction:** Describes the purpose and structure of the document as well as key assumptions
- **Service-Oriented Architecture Defined:** A common definition of service-oriented architecture and its components
- **ESRI Components in an SOA:** A mapping of how ESRI components may be utilized in a service-oriented architecture

J-9662

2.0 SOA Defined

Service-oriented architecture is a method of building business applications that utilize common services to support business functions.

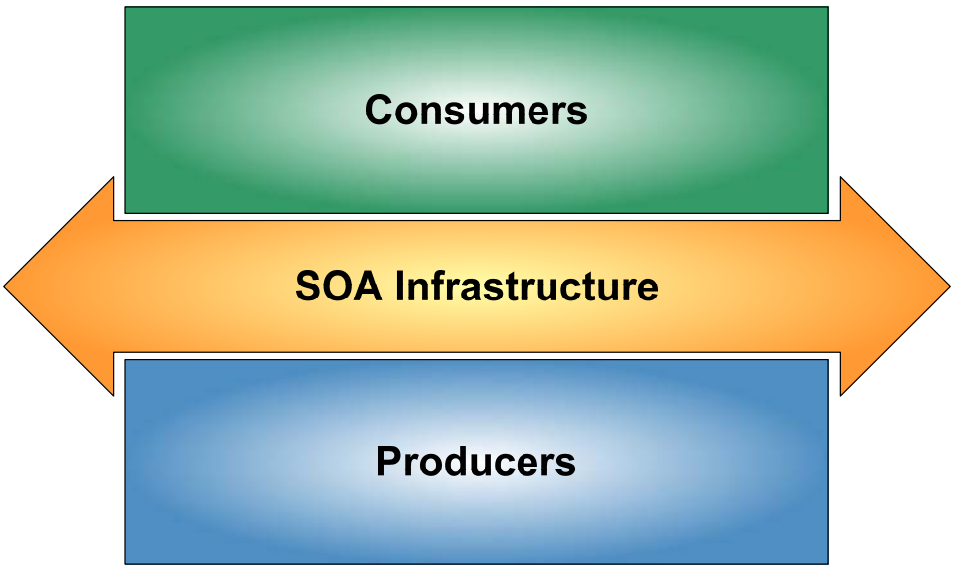
SOA has a variety of definitions regarding what it consists of, sometimes driven by vendor-specific solutions instead of a general architecture. In response to some of the inconsistencies, the Organization for the Advancement of Structured Information Standards recently provided a reference model for SOA so that there is general consensus on its key aspects. The key principles of the model are found in appendix B and serve as the core of ESRI's SOA view.

Although this paper focuses primarily on the technical details of the components required to implement an SOA, it is important to remember that SOA is about more agile business processes and not purely a technology-driven solution.

2.1 Creating an SOA Solution

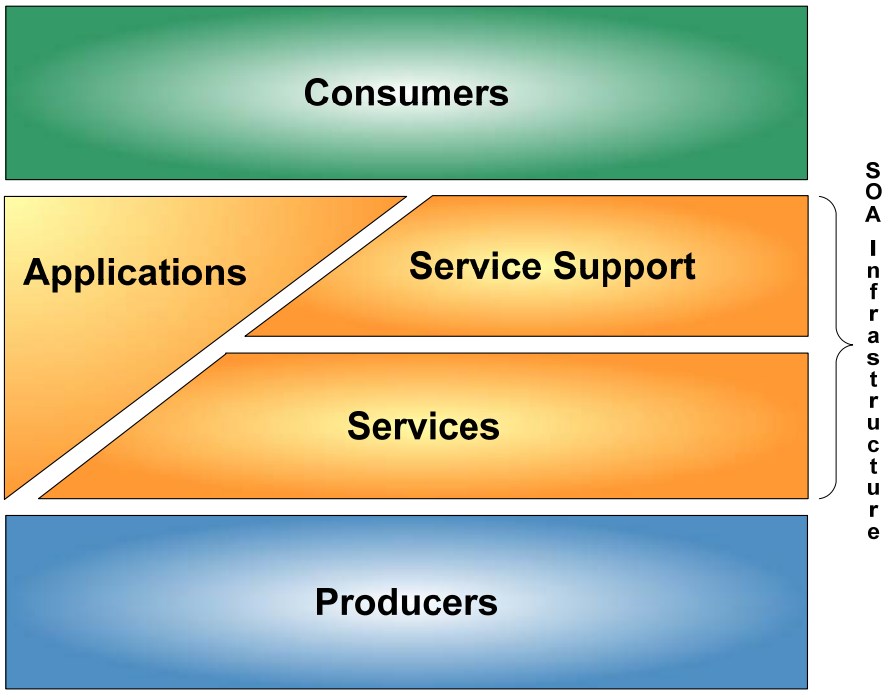
SOA is an architectural concept, and hence to realize an SOA solution, one must map the architecture to a logical construct followed by an implementation using a specific set of technologies/products/platforms. As with any other solution, an SOA is characterized by a set of both mandatory and optional components. An SOA solution consists of the following three main logical components shown in figure 1:

Figure 1
Basic Logical SOA Components



The focus of this paper is to primarily provide deeper insight into components of the SOA infrastructure layer. Therefore, the SOA infrastructure layer can be divided into three subcomponents as shown in figure 2:

Figure 2
SOA Infrastructure Subcomponents



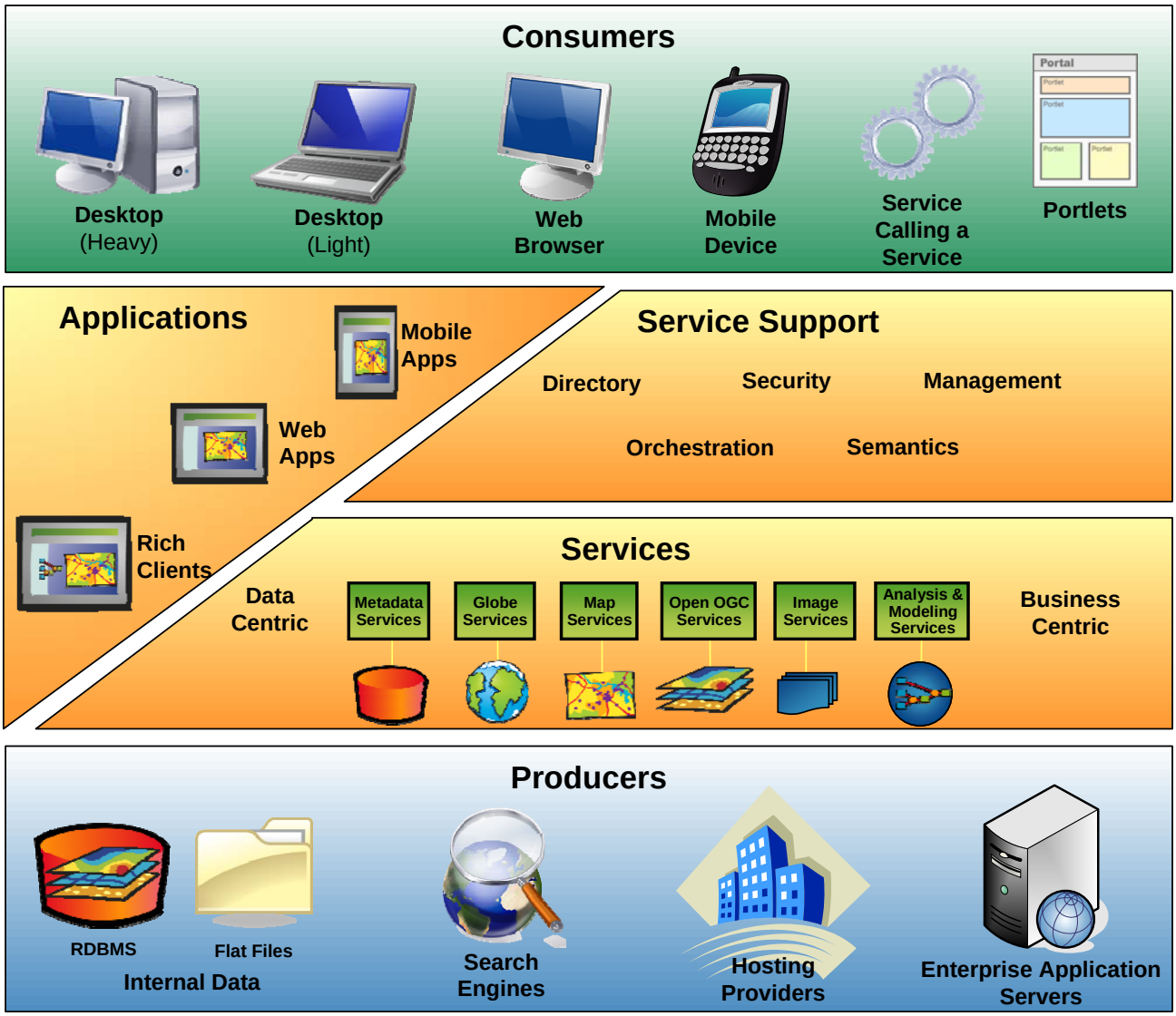
It is quite common for the components of an SOA to be laid out by vendors based on their current software solutions. This is the area where marketing and architecture design overlap and result in many different representations of an SOA.

The purpose of this section is not to focus on a single vendor's solution but to provide a framework for how different functions fit together to support a true enterprise SOA. These five main SOA components can be defined as follows:

- **Consumers—**An entity that makes use of the service offered by the producer
- **Applications—**Provide a graphic interface and varying degree of tightly coupled business logic for consumers to perform tasks (a crossover area of enterprise solutions versus the SOA loosely coupled ideal)
- **Services—**An entity that performs a specific task when invoked
- **Service Support—**An entity that provides the background support functions for SOA
- **Producers—**An entity that offers a specific service or functionality

Each of these components contains additional subcomponents as shown in figure 3, which are described in detail below.

Figure 3
Geospatial SOA Component Details



2.1.1 Consumers

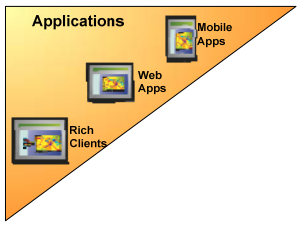
Consumers participate in service-oriented architectures by providing the end user interface. Geospatial visualization and analysis are frequently performed through both desktop (heavy and light) and mobile interfaces. Other common interfaces include browser, portal, and even other Web service components, providing a platform, device, operating system, and programming language-independent mechanism for obtaining information. This tier communicates with the SOA infrastructure through a method transport protocol. For the .NET platform, this protocol is usually either Distributed Component Object Model (DCOM) or SOAP. For Java 2 Enterprise Edition (J2EE)

systems, it is typically SOAP or Internet Inter-ORB Protocol (IIOP). A brief summary of the various consumer interfaces is listed below:

- **Desktop**
  - Heavy—Complete application functionality, but limited to specific platforms
  - Light—Small, focused applets/applications
- **Web Browser**
  - Platform- and OS-neutral user interface
- **Mobile Device**
  - A simplified user interface for phones, pagers, PDAs, and Tablet PCs
- **Service Calling a Service**
  - A single service calling multiple other services for consolidating information
- **Portlets**
  - A portal Web site component providing access to a specific information source or application

Consumers can be designed to operate completely independently of common SOA infrastructure components (service support and services) or utilize them to varying degrees.

2.1.2 Applications



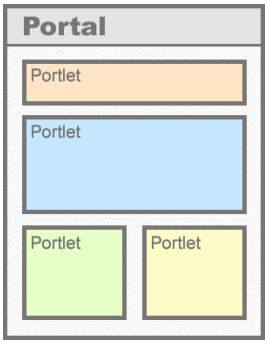
The applications component is represented as a wedge shape, since rich client solutions on the left side are less dependent on the services and service support components to perform their work, whereas Web applications (Web apps, mobile apps) are reliant on additional services and service support functionality.

- **Rich Client Solutions**
  - Typically related to the desktop heavy and light clients, allowing for robust, highly interactive analysis and visualization beyond current technological capabilities of Web or mobile application solutions
  - Can perform SOA infrastructure layer-type functions (in addition to the presentation layer functions) so that users may have full application functionality without requiring additional SOA infrastructure middleware
  - Can operate in disconnected environments
- **Web Application Solutions**
  - Can be developed independently of both the common service support and services components or be completely dependent on them. Complete dependence

of a vendor-provided Web app on external common components forces a user to deploy SOA infrastructure components that may not align to their standards such as a J2EE or .NET application server.

- Single Interface—Application interface designed as a stand-alone solution, not requiring portal infrastructure. These solutions may be tightly coupled solutions that are stove-piped from other applications or provide standards-based system interfaces for interoperability needs.

- Portals and Portlets—Portlets are applications designed as a component that can be integrated into a portal interface. Users are provided with an interface where they can add or remove components from their screen via point-and-click operations. The application programming interface (API) standards for portal subcomponents are different for Java and .NET solutions:



  - Java systems utilize portlets based on Java Specification Request (Java SR)-168, which does not currently support asynchronous calls, so AJAX implementations can be problematic. Further information is available at http://developers.sun.com/portalserver/reference/techart/ajax-portlets.html.

  - .NET systems utilize webparts (a portlet variation) based on the Microsoft SharePoint portal.

  - See appendix D for a high-level comparison of Java versus .NET functionality.

- **Mobile Application Solutions**

  - Custom user interfaces are typically utilized for mobile devices to provide a specific user experience via mobile development platforms.

2.1.3 Services

Services are the architecture components that process business logic for the organization. A more formal definition of *service* is an abstract notion that must be implemented by a concrete *agent* where

- The agent is the piece of software or hardware that sends and receives messages.

- The service is the resource characterized by the set of functionality that is provided.

- Services are abstract in the sense that you might provide the same service using one particular agent one day, then use a different agent the next.

There are two primary perspectives on services:

- **Business Perspective**
  - IT assets that correspond to real-world business activities and recognizable business functions
- **Technical Perspective**
  - Coarse-grained, reusable IT assets that have well-defined interfaces (service descriptions), which can be broken down further into two categories (as shown in figure 3):
    - Datacentric—Tightly coupled with specific datasets and may offer access to customized portions of that data
    - Businesscentric—Services designed based on business process needs rather than data access needs

The primary driver for the creation of services in an SOA is the business perspective, including workflow and governance, which are independent of the technology being implemented. However, the focus of this paper is on the technical perspective and to facilitate an understanding of the technical components within a geoenabled SOA.

Services provide tangible results for consumers, but the big question remains, What is the framework or support structure underneath these services in an SOA? Service Support, discussed in the next section, addresses this need.

2.1.4 Service Support

Service support is the heart of the various components that support an SOA. Service support is commonly represented as an arrow bridging communication between consumers and producers and has a variety of names depending on the author:

- SOA infrastructure
- Integration platform
- Enterprise service bus (ESB)
- Enterprise services

Instead of attempting to dissect various support offerings separately from each other (such as, What is in an ESB?), this section provides an overview of the key standards and functions necessary to support an SOA infrastructure.

The purpose of service support is not to provide new user-facing functionality as much as to facilitate a robust SOA infrastructure to later deploy technical services (datacentric or businesscentric) into. Before discussing each of the service support functions in detail, it is worthwhile to have a high-level understanding of the standards behind some of them including Web service standards.

2.1.4.1 Standards

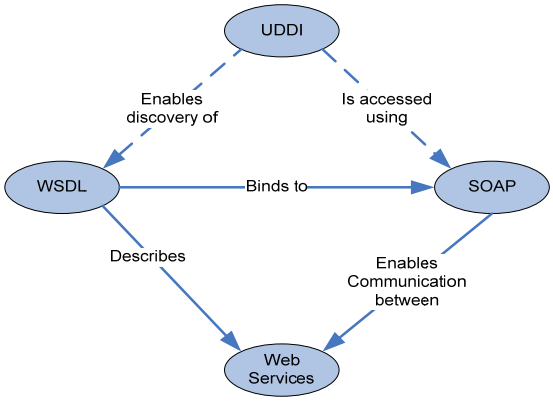
While Web services and SOA are usually thought to be synonymous from the technical perspective, they are not. Web services are an important tool and one implementation method for SOA, but there are other patterns that may be more appropriate for any given use case.

For the purposes of this paper, service protocol standards are divided as follows:

- Web service APIs
  - SOAP-based Web services
    - WS-* standards
    - Prevalent on internal and business-to-business systems
  - Plain Old XML (POX)
    - Current OGC standards
    - Most prevalent Internet Web service protocols
  - Representational State Transfer (REST)
    - Common, simple API for large Internet commerce sites
- Direct Data Links
  - SQL—ODBC/Java Database Connectivity (JDBC)
  - Proprietary

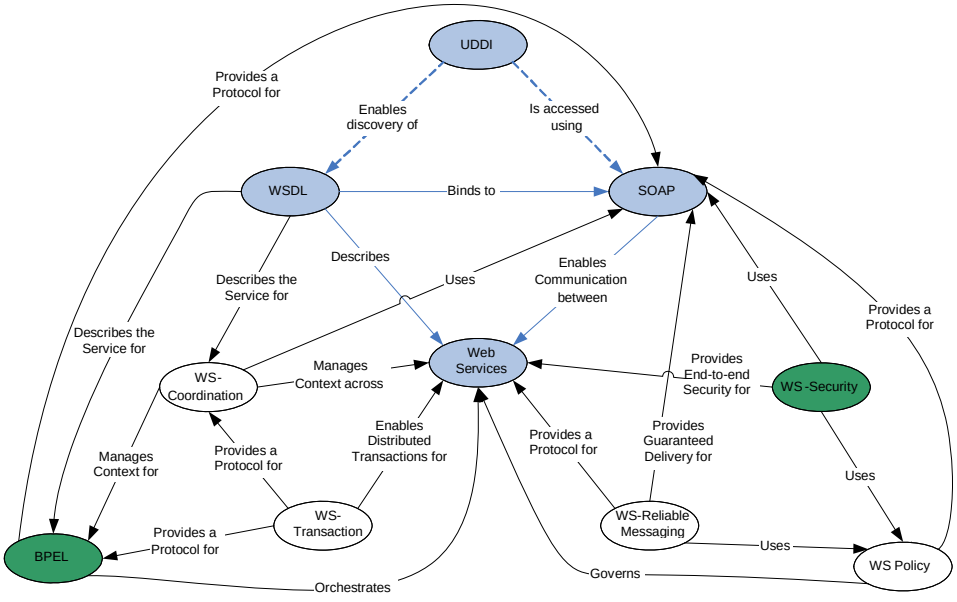
Web services are considered key to making SOAs practical because they provide a relatively easy-to-implement mechanism for enabling application interoperability. Web services are based on an industry-standard protocol stack that includes individual protocols for discovery, description, and remote service calls (see figure 4) as well as Hypertext Transfer Protocol (HTTP) and TCP/IP. The protocols in widest use today include Web Services Description Language (WSDL) (for discovery) and SOAP (for calling business services). SOAP-based Web services are the most marketed form of Web services in the IT industry.

Figure 4
SOAP-Based Web Services



Additional SOAP-based Web service standards, shown in figure 5 below, are called second-generation Web services (notice the core Web services in the middle). Many of these are still being finalized or have competing standards; however, WS-Security and Business Process Execution Language (BPEL) are two notable exceptions, as they now have general acceptance.

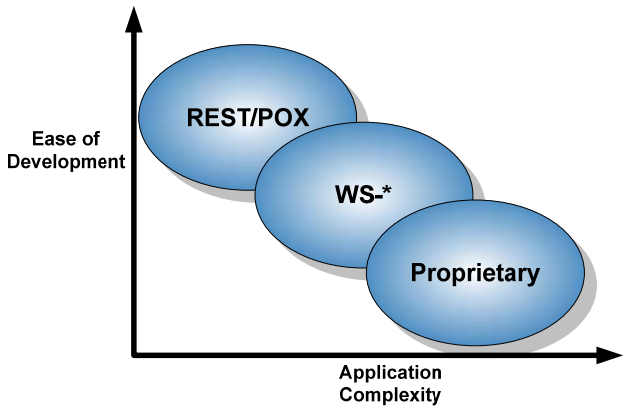
Figure 5
Relationship of Second-Generation Web Services



Web services are no less appropriate than the alternatives if the fundamental criteria for successful distributed object architectures are met. If these criteria are met, Web services technologies may be appropriate if the benefits they offer in terms of platform/vendor neutrality offset the performance and implementation immaturity issues they may introduce.

There are, and will be in the future, plenty of Web services that use raw HTTP as a data transfer protocol and some mutually agreed-upon XML format as the message content (also called POX). XML over HTTP Web services requires little infrastructure to support operations. This is primarily why communication standards, such as REST, are a common Web service solution. Figure 6 below provides a generalized comparison of deployment ease and application complexity for various APIs.

Figure 6
Generalized Interface Comparison



Web services technologies may not *necessarily* be the best choice for implementing SOAs—if the necessary infrastructure and expertise are in place to use COM or CORBA as the implementation technology and there is no requirement for platform neutrality. Using SOAP/WSDL may not add enough benefits to justify their costs in performance and additional complexity.

Examples of solutions that might utilize mechanisms other than SOAP Web services include

- Rich client functionality required for authoring large datasets via database services
- Eliminating performance overhead of SOAP/XML for network-intensive operations
- Providing fine-grained remote procedure call (RPC) functionality
- Message-oriented middleware (MOM)
- Distributed object protocols (such as CORBA-IIOP and DCOM)
- Nontraditional application connection paths such as e-mail
- Providing a simplified API via REST style for Web consumers
- OGC standard Web services

In general, Web services are most appropriate for applications for which the following are true:

- It must operate over the Internet where reliability and speed cannot be guaranteed.

- There is no ability to manage deployment so that all requesters and providers are upgraded at once.

- Components of the distributed system run on different platforms and vendor products.

- An existing application needs to be exposed for use over a network and can be wrapped as a Web service.

Direct data links are still a common mechanism for local components requiring high throughput. These links require no Web service support infrastructure; however, they are relatively "chatty" protocols typically best implemented on local area networks (LANs). Examples of direct data links include SQL database calls over protocols such as ODBC or JDBC.

2.1.4.2 Service Support Functions

Service Support functions have been divided into five key areas for this paper:

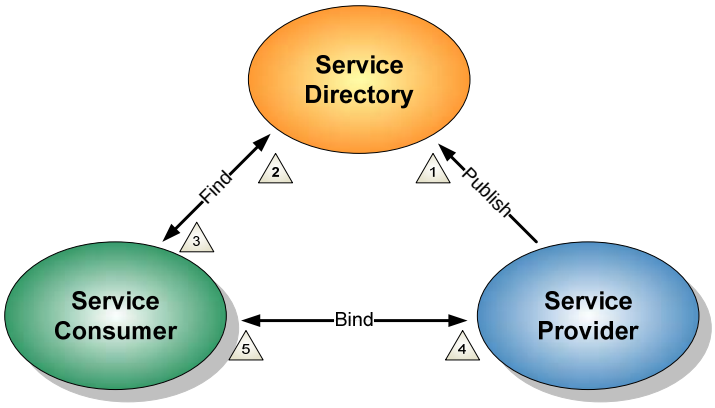
- Directory
- Security
- Management
- Orchestration
- Semantics

2.1.4.2.1 Directory

An important part of the service-oriented architecture approach is the extensive use of metadata in service directories (figure 7). This is important for several reasons:

- Fosters interoperability by requiring increased precision in the documentation of Web services

Figure 7
Service Directory



- Permits tools to give a higher degree of automation to the development of Web services

- Lowers the cost of deployments and speeds fielding solutions by encouraging reuse

- Provides information about service functional capabilities and operational characteristics such as quality of service

For the purposes of this paper, directories may be divided into two main groups, primarily based on implementation scope:

- **Registry—**Systemwide metadata
  - Universal Description, Discovery, and Integration (UDDI)—Most widely implemented registry solution in the IT industry
  - ebXML/ebRIM—Most flexible and extendable registry and repository solution
- **Catalog—**Domain specific such as geospatial metadata only
  - Web Catalog Service (CS-W)—An OGC standard for geospatial metadata

UDDI is an online "yellow book," used by both service providers and service consumers, that is domain independent and complies with the WS-I standards of IT-level interoperability. Service providers register their Web services into UDDI and service consumers search the service descriptions from this online registry, which will finally lead to the services they desire.

The ebXML registry provides a more structured environment, as it stores rigidly defined business processes and related metadata along with collaboration protocol profiles. Both UDDI and ebXML allow businesses to discover one another to dynamically initiate business-to-business transactions. The similarities between ebXML and UDDI are apparent when looking at the design of the Java API for XML Registries, which abstracts UDDI and ebXML and provides a single API for both types of XML registries.

The ebXML Messaging Service is an extension of SOAP—meaning that ebXML provides a highly structured SOA. The ebXML Messaging Service extends SOAP to provide support for security, improved error handling, and reliable messaging. With these extensions, the ebXML Messaging Service can rely on a well-known and well-established protocol to perform sensitive and timely business transactions.

Increasingly, registry *and* repository, as compared in table 1, are seen as integral parts of an SOA "platform."

Table 1
Comparing Registry and Repository Functions

	Design Time	Runtime	IT System Analogy
Registry	Discovery Description	Contracts Policies Versioning	Domain name server (DNS)
Repository	Code versions Documentation	Queriable message store Logging Auditing	WhoIs database

Catalogs typically contain more granular-level metadata for a service and, therefore, might be implemented as a domain-specific solution. However, recently, there have been efforts to create crossover directory functionality between geospatial standard catalogs and IT standard registries. OGC's ebXML/ebRIM profile for the CS-W standard is such an example. This solution allows different types of queries (both proprietary and standardized) to be executed against different APIs pointing to a common registry metadata set.

Without metadata registries and catalogs, valuable deployment resources may be wasted because key individuals, divisions, and partners will find it difficult to locate Web services. The key questions when selecting registry and repository products should be whether the needs are focused on design time or runtime. By understanding which functions are needed for the life cycle of your SOA services, a better selection can be achieved.

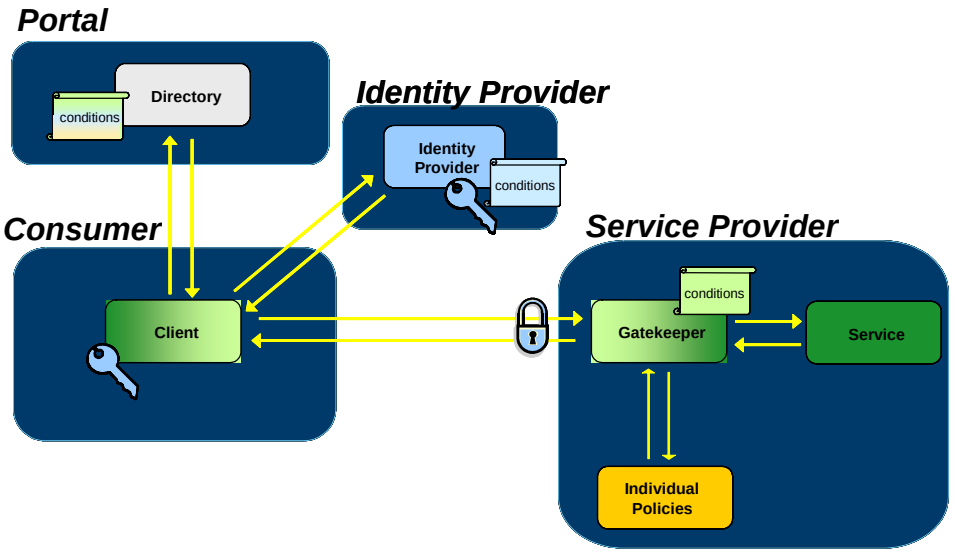
2.1.4.2.2 Security

Threats to Web services involve threats to the host system, the application, and the entire network infrastructure. To secure Web services, a range of XML-based security mechanisms is needed to solve problems related to authentication, role-based access control, distributed security policy enforcement, and message layer security that accommodates the presence of intermediaries.

Secure Sockets Layer (SSL) (a point-to-point security solution) can address many immediate security concerns but is not the technology of choice for SOA. When services begin to take on greater amounts of processing responsibility, the need for message-level security begins to arise. The WS-Security framework establishes an accepted security model supported by a family of specifications that permeate service-oriented application and enterprise architectures on many levels. The importance of acquiring a sound knowledge of the WS-Security framework now cannot be emphasized strongly enough.

SOA abstracts application functionality to a business services layer, regardless of the location or awareness of the underlying applications. Therefore, identification and access policies for different systems may not be the same or easily coordinated. Users accessing a composite application may need to be distinctly authenticated to each participating system. A common solution for this issue is to utilize an enterprise identity management and security policy framework with at least some level of centralized administration. Among SOA-relevant technologies, Security Assertion Markup Language (SAML) is intended to enable federated authentication. Extensible Access Control Markup Language (XACML) may then be utilized to support an authorization policy framework as seen in figure 8.

Figure 8
XACML Policy-Based Security for SOA



Traditional, connection-oriented, point-to-point security mechanisms may not meet the end-to-end security requirements of Web services. However, security is a balance of assessed risk and cost of countermeasures. Depending on the implementer's risk tolerance, point-to-point transport-level security may provide enough security countermeasures.

2.1.4.2.3 Management

A common question for architects starting to research SOA is, Do I need an ESB to manage my SOA? It is better to think of ESB as a service architecture of infrastructure capabilities, then consider what products best provide an implementation of each of the infrastructure services required. This may result in the selection of a product (or products) bearing the ESB label, or it may not. Most J2EE Web application server vendors, such as IBM and BEA, incorporate ESB-type functionality into their suite of offerings. Microsoft does not plan to release an ESB suite, as much of the functionality is embedded in the operating system and with new messaging functionality in its forthcoming Windows Communication Foundation (WCF).

Web service management capabilities typically include

- Monitoring
- Transaction management
- Routing
- Transformation
- Controlling and reporting of service qualities
- Service usage

Service qualities include health qualities such as availability (presence and number of service instances), performance (e.g., access latency and failure rates), and accessibility (of end points). Facets of service usage information that may be managed include frequency, duration, scope, functional extent, and access authorization.

A Web service becomes manageable when it exposes a set of operations that support management capabilities. These management capabilities realize their monitoring, controlling, and reporting functions with the assistance of a management information model that models various types of service usage and service quality information associated with management of the Web service. Typical information types include request and response counts, begin and end timers, life cycle states, and entity identifiers (e.g., of senders, receivers, contexts, or messages).

Although the provision of management capabilities enables a Web service to become manageable, the extent and degree of permissible management are defined in management policies associated with the Web service. Management policies, therefore, are used to define the obligations for, and permissions to, managing the Web service.

Just as the Web service being managed needs to have common service semantics that are understood by both the requester and provider entities, Web service management requires common management semantics, with respect to management policies and management capabilities, to be understood by the requester and provider entities.

2.1.4.2.4 Orchestration

An orchestration expresses a body of business process logic that is typically owned by a single organization, whereas choreography attempts to organize information exchange between organizations. Orchestration is a capability that controls the execution of software-based units (e.g., applications, components, services) so as to achieve a prespecified effect.

A key aspect to how orchestration is positioned within SOA is the fact that orchestrations themselves exist as services, called composite services. Therefore, building on orchestration logic standardizes process representation across the organization while addressing the goal of enterprise federation and promoting service orientation.

The primary industry specification that standardizes orchestration is Web Services Business Process Execution Language (WS-BPEL). The second version of BPEL was approved in April 2007. Many organizations are focusing their initial SOA infrastructure efforts in other areas first since some of the supporting Web services surrounding WS-BPEL, such as WS-Coordination and WS-BusinessActivity, are still immature. The initial standard does not incorporate humans into the workflow but is being addressed through two avenues:

- BPEL4People—A proposed extension for WS-BPEL
- Treating people as services in the workflow

2.1.4.2.5 Semantics

Service providers and consumers must have a shared understanding of the meaning of the messages they exchange. Semantic integration is critical to dynamic, automated interactions between companies.

The semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. It is a collaborative effort led by the World Wide Web Consortium with participation from a large number of researchers and industrial partners.

The semantic Web addresses two main issues:

- Common formats for interchange of data

  The initial Web focused on the interchange of documents.

- Language for recording how the data relates to real-world objects

These capabilities allow a person (or a machine) to start off in one database and move through an unending number of databases that are connected not by wires but by common formats/languages.

In February 2004, the W3C released the Resource Description Framework (RDF) and the Web Ontology Language (OWL) as W3C recommendations. RDF is used to represent information and to exchange knowledge on the Web. OWL is used to publish and share sets of terms called ontologies, supporting advanced Web search, software agents, and knowledge management. Note that of all the support functions discussed so far, these components are some of the least developed and require significant additional research before their long-term goals are reached as seen in the Gartner estimated rollout dates below:

- 2006 Semantic Hypertext (semantic tags)
- 2016 Semantic Data (RDF/OWL data)
- 2026 Semantic Environment (multiple RDF/OWL implementations with tuple space)

2.1.5 Producers

Web service producers define a service request format and publish it to service directories for discovery and reuse.

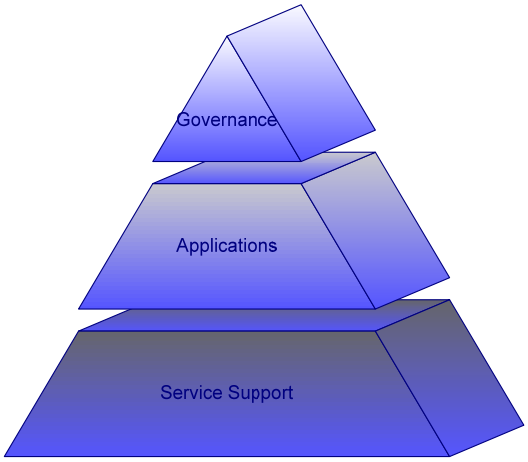
- Internal Data—Data is available for processing via high-speed LAN.
- Enterprise Application Servers—Examples include extending functionality of enterprise resource planning and customer relationship management products.
- Hosting Providers—Data is available via WAN, allowing usage of data not necessarily owned by consumers.
- Search Engines—Allow for discovery of detailed metadata about services.

2.2 SOA Delivery Strategy

What is the primary approach of your SOA implementation? Is the goal to cut across silos and implement some services so that data is exposed to do great things? If so, you may be on a path to SOA failure. Why? Because successful SOA implementations typically follow an opposite or middle-out approach (figure 9):

- First, identify relevant business process scenarios.
- Next, identify where the scenarios touch silos.
- Last, identify opportunities for services where scenarios and silos intersect.

Figure 9
SOA Functionality Drivers



"Services should not be driven bottom up from technology, but rather from the top down—with the use cases."—Grady Booch

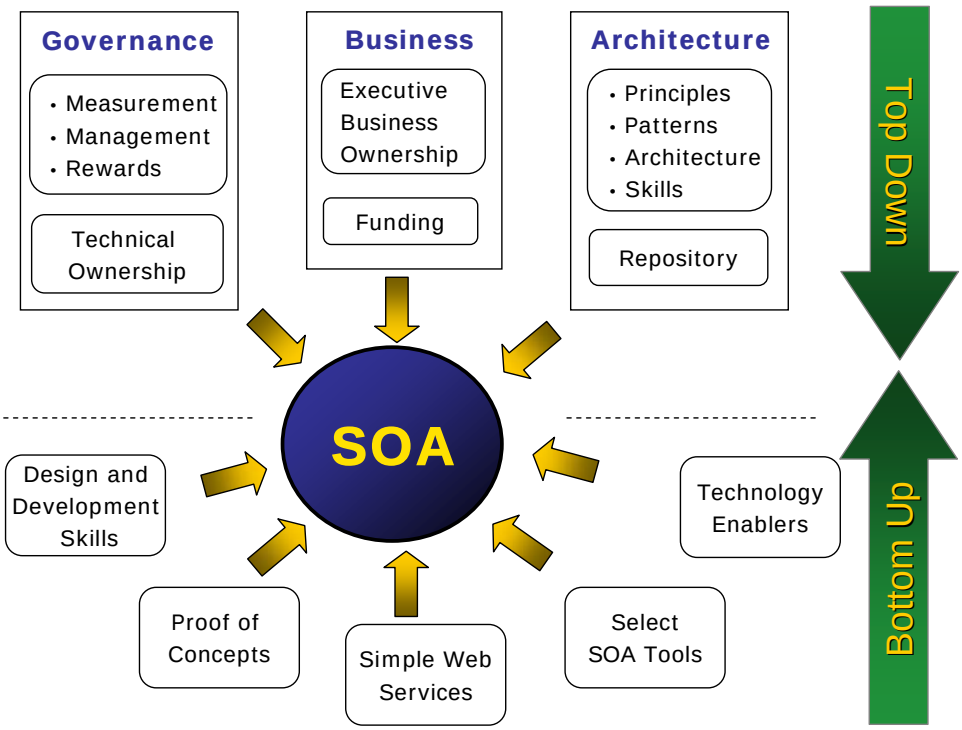
A pure top-down strategy as suggested by Booch can result in the highest-quality level of SOA, but it also imposes a significant volume of up-front analysis work. Many large organizations' current efforts require immediate results, and therefore, a top-down approach is typically avoided.

A purely bottom-up strategy is based on the delivery of application services on an as-needed basis. This common approach is easy to follow but does not result in the advancement of service orientation. After a year or two, these same organizations complain that SOA doesn't work.

For many organizations, it is useful to view these two approaches as extremes and to find a suitable middle ground. SOA experts have proposed an agile strategy that attempts to balance the extreme strategies as seen in figure 10. Using the agile approach, ongoing analysis is supported while still allowing the immediate delivery of services. As analysis progresses, existing services are revisited and revised as required. This method is more complex because it needs to meet two opposing sets of requirements.

J-9662

Figure 10
Putting Together an SOA Delivery Strategy



2.3 New Technologies Facilitating SOA

A key benefit of SOA is that it is independent of a particular technology standard. This allows for easily incorporating appropriate new technologies as they become available. Below is a brief listing of noteworthy new technologies that can help facilitate implementation of an SOA:

- Integrating XML Gateways
  - Allows functionality of XML gateways to be integrated into load balancer hardware
  - Reduces number of times XML needs to be parsed as it passes through systems while also improving security
- Web 2.0
  - Web 2.0 is much more than a single technology; however, AJAX is a key component of its successful emergence over the last year. More than 30 percent of Web 2.0 mashups incorporate mapping imagery and an AJAX presentation layer.
    - AJAX is a collection of techniques that Web developers use to deliver an enhanced, more responsive user experience in the confines of a modern browser. A narrow-scope use of AJAX can have a limited impact in terms of making a difficult-to-use Web application somewhat less difficult. However,

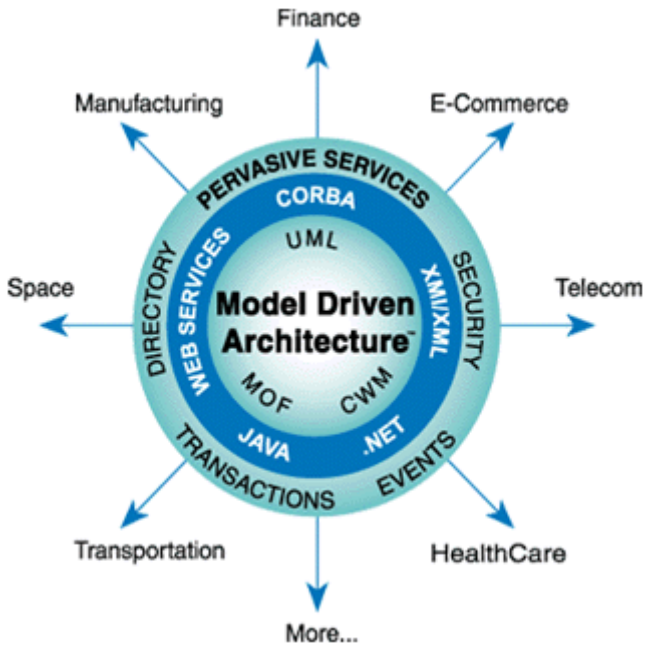
industry analysts say that even this limited impact is worth it, and users will appreciate incremental improvements in the usability of applications. High levels of impact and business value can only be achieved when the development process encompasses innovations in usability and reliance on complementary server-side processing.

- Architecture

  - Event-driven architecture (EDA) is a common style for distributed applications that are typically designed into modular, encapsulated, shareable components with event services. The services can be created through an application, adapter, or agent acting non-invasively. Industry experts estimate EDA is at least five years from mainstream maturity.

  - Model-driven architecture (MDA) is a technology from the Object Management Group (OMG). The synergistic relationship between SOA and MDA quickly becomes apparent viewing the MDA framework in figure 11. The technology distinguishes business-level functionality from the technical complexity of its implementation, enabling the applications to be modeled by standards like Unified Modeling Language (UML). This allows the models to operate free from potential platform limitations and instantiate them into specific runtime implementations using a target platform of choice. Both MDAs and EDAs will find their niche with developers, largely thanks to SOA and their bottom-line boosting perks.

Figure 11
Model-Driven Architecture



J-9662

2.4 General SOA Summary

A traditional business application is nothing more than a software vendor's opinion of how a business process should be performed. Until recently, this dynamic has prevented IT departments from delivering maximum value to the enterprise because often vendor opinions and real-world processes did not match. As a result, IT systems have largely failed to support business processes and have instead focused on automating portions of processes.

SOA is changing this by enabling IT departments to take a process centric approach to architecture. This is also changing the way in which enterprises evaluate IT vendors. ESRI recognizes the strong value proposition SOA provides for enterprise solutions, and the road map for ESRI's alignment with SOA is discussed in the next section.

3.0 ESRI Components in an SOA

Geospatially enabling an SOA requires knowledge of not only the organizational business processes but also the robust capabilities and benefits that geographic information system (GIS) technology offers. First, business processes should be distilled into common functions that can be delivered throughout the enterprise in conformance with the overall mission and goals of that organization, with services being created for each function. Second, business processes should be evaluated with an understanding of how GIS technology can extend and enrich those processes by providing value in overall efficiency, accuracy, accessibility, and cost savings. While these concepts are by no means revolutionary, they are evolutionary because they continue to bring technology, policies, and practices in harmony with organizational business processes.

ESRI offers solutions today for implementing a geospatial SOA that includes both desktop (ArcGIS Desktop) and server (ArcGIS Server) technology (see figure 12). These solutions enable organizations to author, publish, and serve their geographic knowledge to the broader organization and external entities. The role of ArcGIS Desktop is in defining and authoring the content that becomes the basis for common, reusable spatial services. GIS professionals can leverage their existing geographic expertise to author and design geospatial content such as maps, globes, geoprocessing models, locators, and data management functions.

Figure 12
ArcGIS Components Supporting an SOA

ArcGIS Server is a comprehensive Web-based GIS that provides out-of-the-box end user applications and services for mapping, analysis, data collection/editing, distribution, and management of spatial information. It provides a standards-based platform upon which ArcGIS Desktop users can easily publish and serve their geographic knowledge to the broader organization through an integrated IT architecture.

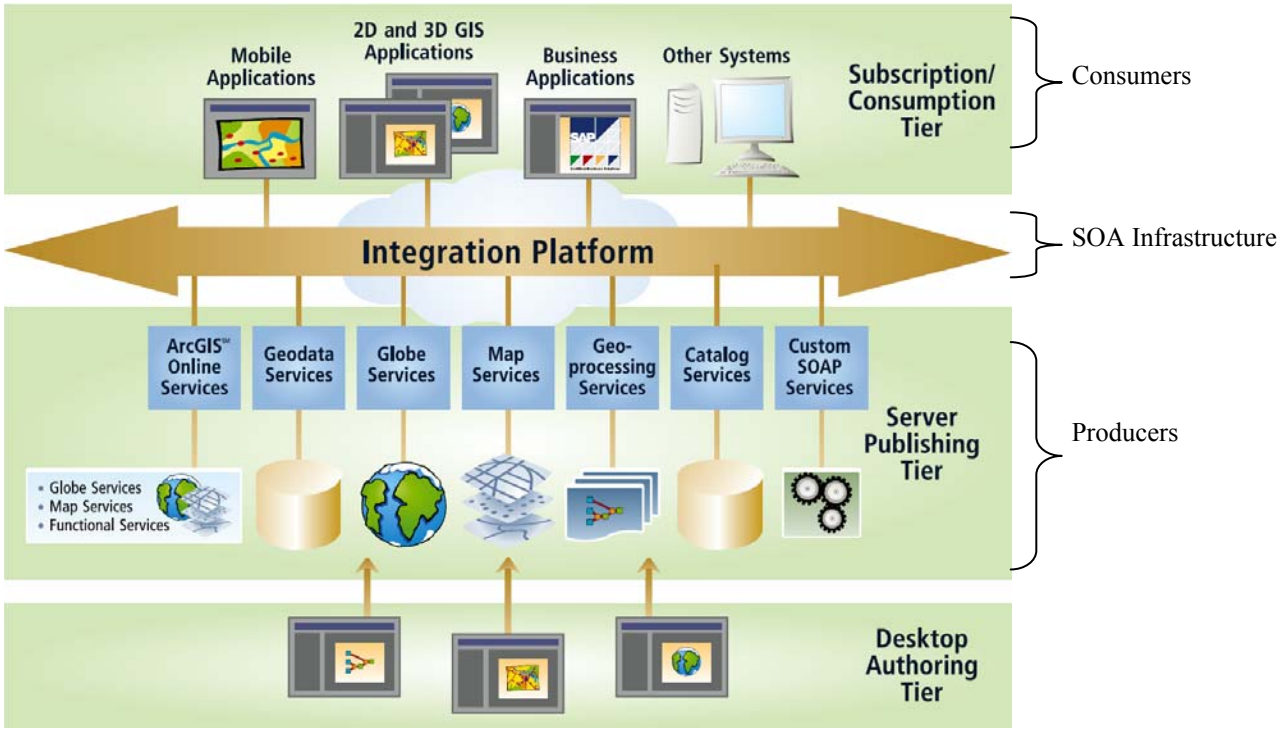
3.1 Why Use GIS in a Service-Oriented Architecture?

Integrating GIS with other key business systems can extend the value of those systems by increasing accuracy, efficiency, and productivity. For example, an enterprise resource planning (ERP) tool can be used to identify government personnel with specialized skills to assist in a chemical spill accident. GIS services can be leveraged by the ERP tool to more accurately understand the location of those personnel in relation to the event and use calculated travel times to allocate appropriate resources, while avoiding potentially dangerous areas downwind from the accident location. In this case, GIS services extend the value of the ERP system and provide a more timely and efficient response.

Another practical application is the use of SOA by a large county to integrate a Web application framework with customer relationship management (CRM), GIS, and heritage information systems to improve responsiveness and reduce costs. A case study of this scenario is located in appendix E.

Some common services that support a geospatial SOA include map (2D) and globe (3D) services (transportation, demographics, physical environment, and asset maps/globes), locator services (geocoders and gazetteers), geoprocessing services (site selection models, dispersion/plume models, network analytics, raster analytics, image processing, etc.), and data management services (replication; data check-in/checkout; spatial extraction, transformation, and loading; and catalog services). Shared GIS services such as these can be used to add value to established business systems (e.g., work order, asset, and customer relationship management systems) and support enterprise-wide initiatives for collaborative computing (see figure 13).

Figure 13
GIS SOA Conceptual View



GIS is a proven and valued technology that plays an important role in today's SOA strategies and initiatives. Using SOA, organizations can integrate GIS into their existing workflows and solve today's challenges of providing open access to common geospatial data, services, and applications from within and beyond.

3.2 ESRI Alignment to Key SOA Principles

ESRI is using the industry-standard OASIS reference model for service-oriented architectures as part of its ArcGIS Server product. The reference model is abstract, so ESRI is using it as a design paradigm for parts of its products. A summary of ESRI's conformance with the seven SOA principles appears below:

- Services—ESRI server objects are provided as standard Web service implementations and can act as the service provider.
- Service Descriptions—WSDL description files are automatically generated for ArcGIS Server Web services.
- Visibility—WSDL descriptions can be stored in industry-standard service directories such as UDDI. Service information may also be discovered via the ArcIMS® metadata service via industry-standard interfaces of OGC CS-W 2.0 and/or the OASIS standard of ebXML.
- Interaction—ESRI fully utilizes industry W3C standards, such as SOAP, for facilitating Web services and OGC standards for GIS domain-specific standards.
- Real-World Effect—Consumers are provided industry-standard OGC service results through Web Map Service (WMS), Web Feature Service (WFS), Web Catalog Service, and Web Coverage Service (WCS) in addition to SOAP-based Web services with WSDL descriptions.
- Execution Context—SOAP-based Web services are available out of the box with ArcGIS Server.
- Contract and Policy—WSDL provides a starting framework for contract and policy information. ESRI is also leading viewer advancements within OGC's Geospatial Digital Rights Management (GeoDRM) initiative.

3.3 ESRI Alignment to Standards and Interoperability

ESRI has evolved tools in its software to support and integrate with virtually all the commonly accepted industry and domain standards. For our users, this means compatibility and interoperability support with major enterprise systems such as enterprise resource planning, customer resource management, enterprise application integration (EAI), and work management systems.

ArcGIS 9.2 supports all the leading IT development and application environments as well as OGC and ISO GIS. Some of the supported standards include

- Operating systems including Windows, UNIX (AIX, Solaris), and Linux
- DBMSs, such as IBM DB2 Universal Database and Informix Dynamic Server, Oracle, and Microsoft SQL Server, including support for all spatial types

- Spatial data formats including translators; direct read and data access via SQL, OLE COM, XML, and GML; Web services; published APIs; CAD data; and other GIS formats
- Network protocols such as TCP/IP, HTTP, and HTTPS
- Developer environments including Visual Basic (VB); C#; C++; Visual Studio .NET; and J2EE; Java 2 Micro Edition (J2ME); Java 2 Standard Edition (J2SE); Active Server Pages (ASP)/JavaServer™ Pages™
- Handheld devices, including Windows CE and Pocket PC, within the 802.11 standard IT solutions from organizations such as Autodesk, Bentley, Intergraph, Leica, MapInfo, and Trimble
- Enterprise applications such as SAS, IBM DB2, Oracle, SAP, IBI, and FileNET
- Enterprise service bus middleware such as Sonic, iWay, BEA AquaLogic, and IBM WebSphere ESB
- Web services such as XML, SOAP, UDDI, and WSDL and OGC specifications such as WFS 1.1, WMS 1.3, WCS 1.0, CS-W 2.01, and GML
- J2EE 1.4-certified Web application servers such as Oracle, BEA WebLogic, JBoss, and IBM WebSphere Application Server

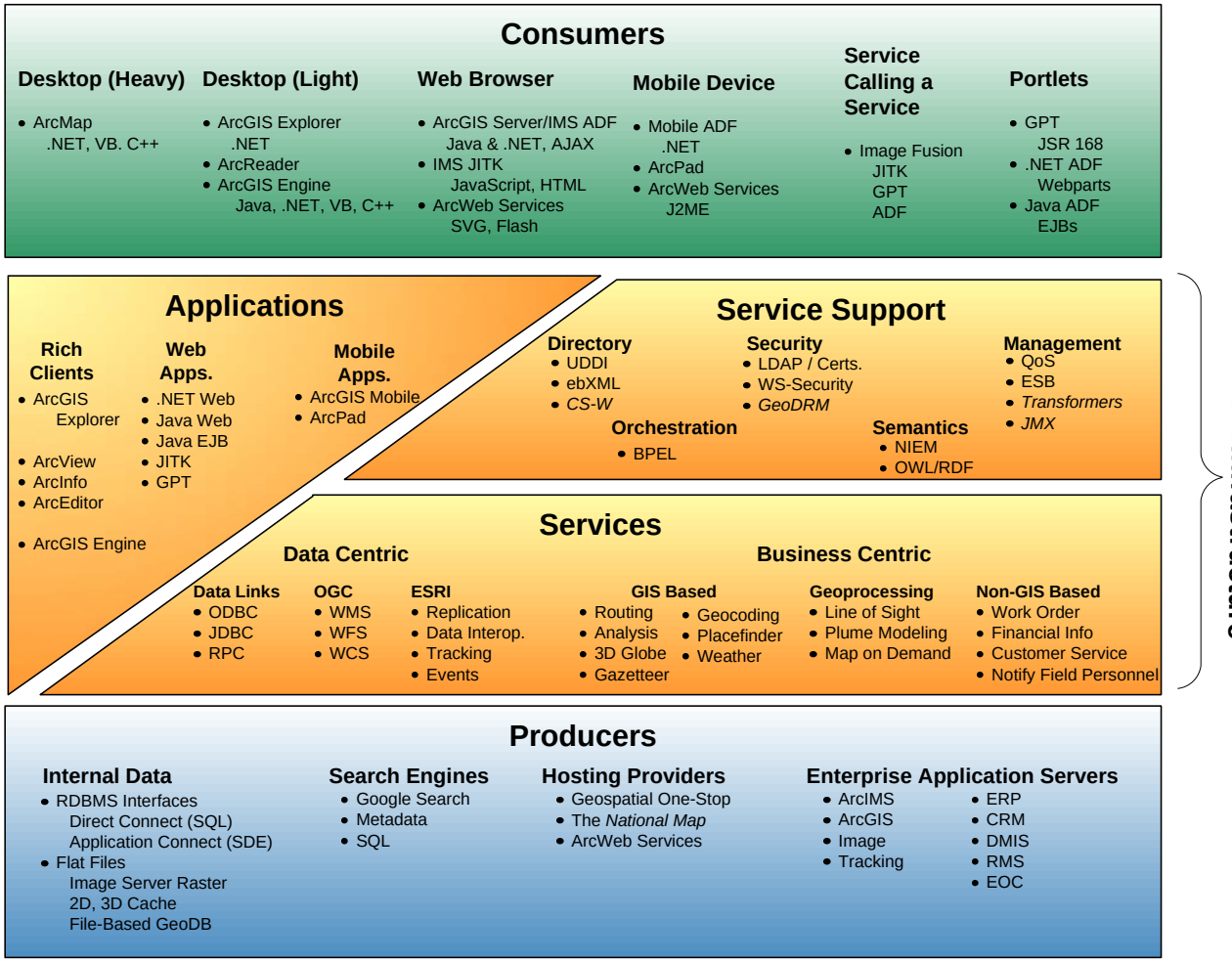
3.4 ESRI Support of SOA Components

ESRI has separated SOA components into the following functions:

- Consumers
- SOA infrastructure consisting of
  - Applications
  - Service support
  - Services
- Producers

Figure 14 provides an overview of ESRI components that address each of the main SOA functions. Each of these functions is discussed in detail below.

Figure 14
ESRI Components Incorporated into SOA



3.4.1 Consumers

This section briefly discusses the technologies provided to consumers through ESRI products and an overview of the types of users consuming services.

3.4.1.1 Technologies

- Desktop
  - Heavy—ArcMap™ for robust GIS editing and analysis
  - Light—ArcGIS Explorer for rich 3D imagery and to utilize ArcGIS Server functions via tasks
- Web Browser—ArcGIS Server and ArcIMS, out-of-the-box AJAX, Web 2.0 map viewers
- Mobile Device—ArcPad®, ArcGIS Mobile, ArcWeb℠ Services, J2ME

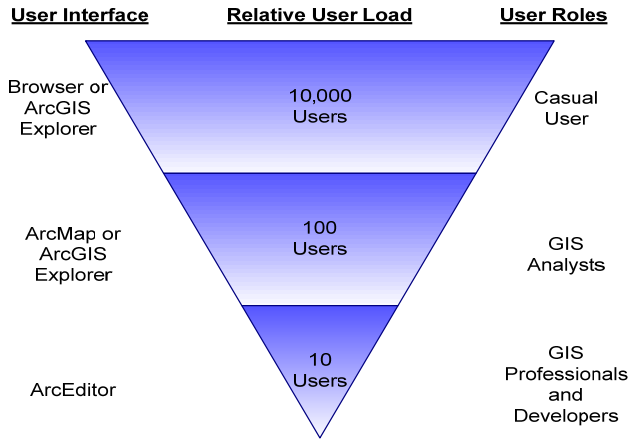
- Service Calling a Service—ArcGIS Server to fuse maps from multiple data sources
- Portlets
  - Geospatial Java Specification Request-168 portlets interchanged across J2EE portals
  - HTML pages that contain Java ADF components can be deployed to portlets
  - ArcGIS Server Java Application Development Framework (ADF™) components AJAX enabled and cannot be deployed separately within portlets

3.4.1.2 User Roles

The GIS user base can be grouped into three primary roles as follows (figure 15):

- Casual Users
  - Consume GIS-based Web services via focused applications where geospatial services are dynamically discovered or infused into the fabric of an application.
- IT Architects/Developers
  - Discover consumable services published by GIS professionals when building or customizing applications in support of business workflows such as work order management systems.
- GIS Professionals and Analysts
  - Publish and promote their services in the form of shared maps, globes, processes, and functions and also consume services that are published by others in the GIS community.

Figure 15
Relative GIS User Base



3.4.2 Applications

ESRI provides out-of-the-box Web 2.0 mapping applications. ESRI also supports extending application functionality extensively in both the .NET and Java platforms, as seen in table 2. To further improve the developer experience, ESRI utilizes managed code for areas where main developer interaction is expected. There are three main types of applications that may be created with ESRI's software developer kits (SDKs):

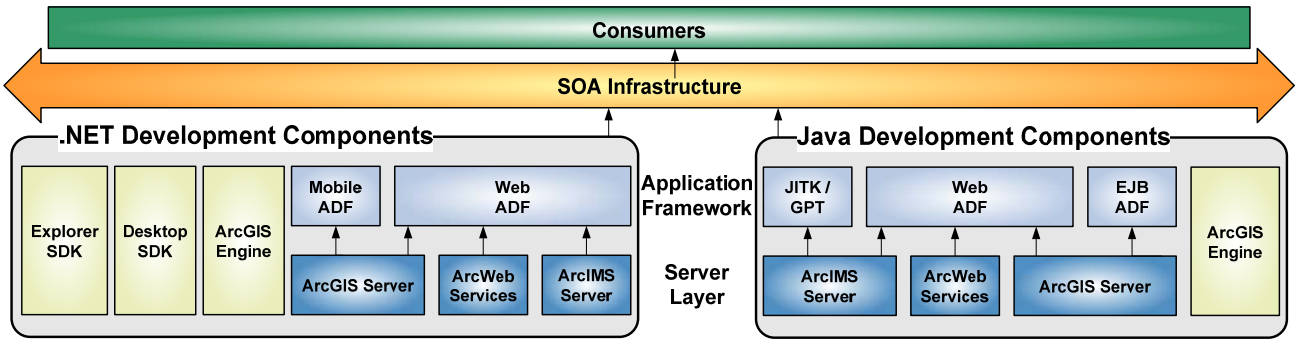
- Rich client solutions
- Web application solutions
- Mobile solutions

Table 2
ArcGIS Software Developer Kits

Application Type	Development Framework	.NET	Managed Code	Java
Rich Client	ArcGIS Engine	X		X
	ArcGIS Desktop	X		
	ArcGIS Explorer	X	X	
Web Application	ArcGIS Server Web ADF	X	X	X
	ArcWeb Services	X		X
	ArcIMS Web ADF	X	X	X
	Java Integration Toolkit			X
	GIS Portal Toolkit (GPT)			X
Mobile	ArcGIS Mobile	X	X	
	ArcWeb Mobile			X

Figure 16 summarizes the different ESRI development frameworks grouped by development platform (.NET and Java) and their supporting server components. Rich clients are represented as yellow, and server components are blue. (Each application development framework is described further below.)

Figure 16
ESRI Development Frameworks



3.4.2.1 Rich Client Solutions

It is critical for application components to interoperate with non-HTML centric applications.

- ArcGIS Engine
  - COM, .NET, Java, and C++ application programming interface for developers.
  - Developers can build focused GIS solutions with simple interfaces to access any set of GIS functions or can embed GIS logic in existing user applications to deploy GIS to broad groups of users.
  - It supports standard Web service protocols for communication with diverse systems, such as ArcGIS Server, in addition to direct access connections to database systems for optimal performance.
- Desktop
  - The ArcGIS Desktop Software Developer Kit supports the COM and .NET programming frameworks.
  - Many users apply the ArcGIS Desktop Software Developer Kit to add extended functions, new GIS tools, custom user interfaces, and full extensions for improving professional GIS productivity of ArcGIS Desktop.
  - Three license versions are available: ArcInfo®, ArcEditor™, and ArcView®.
- ArcGIS Explorer
  - A lightweight desktop client for ArcGIS Server and an integrated part of the overall product suite
  - Provides free access to default services for public noncommercial use
  - Allows access to comprehensive global imagery services (15-meter minimum resolution globally, 1-meter resolution or better for the United States)
  - Supports the integration of local data (shapefiles, file geodatabases, images)
  - Can be used with 2D maps or 3D globes
  - Supports ArcWeb, ArcIMS, ArcGIS Server, WMS, and Keyhole Markup Language (KML) services
  - Enables custom tasks to be implemented that leverage any ArcGIS Server capability such as publishing globes, executing geoprocessing tasks, or running models
  - Allows maps, tasks, results, and layers to be saved (as an XML file) and shared, providing a user with a personalized environment

3.4.2.2 Web Application Solutions

ESRI provides a variety of software development frameworks to allow implementation across almost any platform. The Web Application Development Framework includes support for ArcGIS Server and ArcIMS and is part of both products. OGC WMS and ArcWeb Services are also supported. The Web ADF provides one framework for building Web applications and services no matter what services are needed.

Frameworks are also available for facilitating implementations utilizing Enterprise JavaBeans with J2EE servers, metadata portals, and advanced technology solutions with the Java Integration Toolkit (JITK). This section describes the following ESRI server-based SDKs: .NET Web ADF, Java Web ADF, EJB ADF, JITK, and GIS Portal Toolkit.

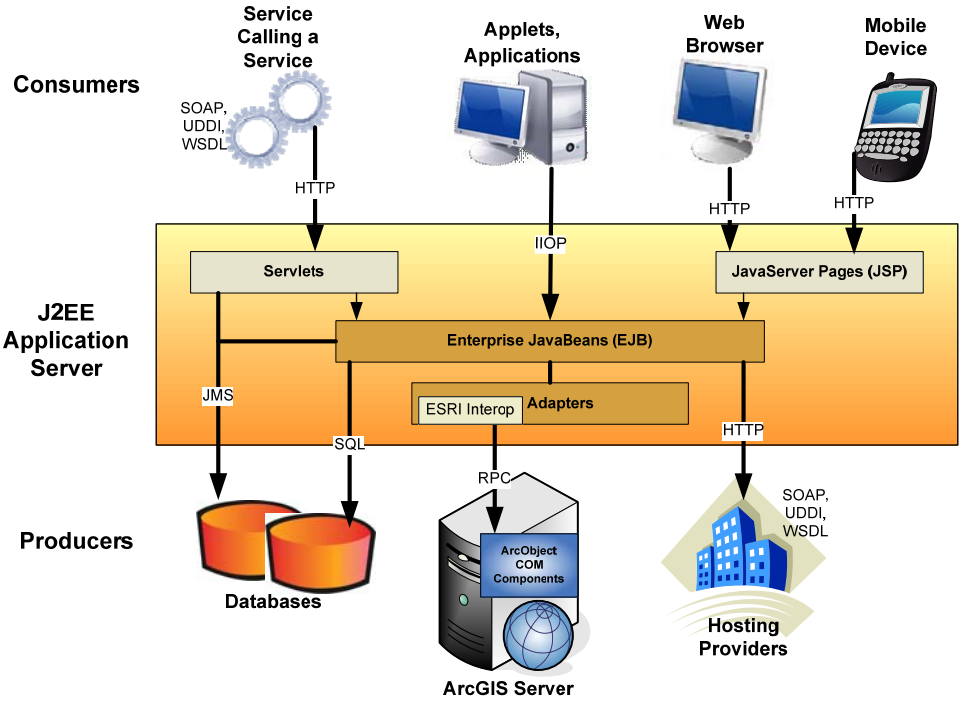
- .NET Web ADF
  - Contains ASP.NET libraries, Web controls, Web application templates, developer help, and code samples.
  - Works with multiple data sources
  - Engineered for full Microsoft integration with
    - Microsoft Visual Studio 2005 integrated development environment (IDE)
    - Microsoft Windows Server
    - Microsoft Internet Information Server (IIS)
    - Microsoft .NET 2.0 Framework
  - Microsoft .NET 3.0 framework is undergoing ESRI certification. Microsoft provides full backward compatibility between .NET 2.0 and 3.0; therefore, applications may be migrated to the new framework without application changes.
  - ArcIMS and ArcGIS Server solutions can be created completely in .NET ADF without writing ArcXML™.
  - Runtime code is available for production deployment.
- Java Web ADF
  - Same framework functionality as .NET ADF, except for the Java platform with Web controls exposed as JavaServer Pages tags
  - Certified operating systems
    - Red Hat Enterprise Linux
    - SUSE Linux Enterprise Server
    - Microsoft Windows Server
    - Solaris SPARC
- IDE plugins
    - Eclipse
    - Sun Java Studio Creator

J-9662

- Enterprise JavaBeans ADF

  - Enterprise JavaBeans are generally considered a required component of a robust J2EE architecture. J2EE application servers, such as WebSphere and WebLogic, provide the framework for Enterprise JavaBeans to reside in (see figure 17). Most of the advanced features of J2EE servers—such as caching, pooling, clustering, load balancing, and failover—are primarily utilized through Enterprise JavaBeans.

Figure 17
J2EE Operating as the SOA Infrastructure



  - The ArcGIS Server Manager tool packages Enterprise JavaBeans functionality into convenient Enterprise JavaBeans modules (enterprise archive format or .ear files) that are ready to be deployed across major application servers with point-and-click ease.

  - Enterprise JavaBeans provide a scalable architecture for development of distributed applications by which you can access ArcGIS Services.

  - Both SOAP and ArcObjects™ components-based APIs are available to ArcGIS Server (ArcObjects API optionally, via J2EE Connector Architecture; shown as ESRI Interop component in figure 18).

- Certified application servers include
  - Sun Java System (Sun One) Application Server
  - Oracle Application Server
  - JBoss
  - BEA WebLogic
  - IBM WebSphere Application Server
- ArcGIS Enterprise JavaBeans provides the framework to build and deploy GIS-enabled J2EE applications and services to meet a variety of needs using various clients by exposing ArcGIS Server object functionality. ArcGIS Enterprise JavaBeans includes a suite of GIS Enterprise JavaBeans services including
  - Map—Provides export, layer access, map computations, and so forth
  - Geocode—Provides export GeoCode ServerObject to remote clients, single address, batch geocode, reverse geocode, display of geocode results in map, and so forth
  - Geoprocessing—Allows asynchronous execution of a tool/model remotely
  - Network Analysis—Solves transportation network problems, manages stops/barriers, determines regions, determines shortest path in a stateless environment

- Java Integration Toolkit
  - JITK for ArcIMS is a precursor to the Application Development Framework available for both ArcGIS Server and ArcIMS. JITK functionality is being rolled into ADF and will continue to serve as a mechanism for advanced functionality to be deployed with ESRI server products before it is available to general users within ADF.

- GIS Portal Toolkit 3.1
  - Provides advanced GIS metadata search capabilities and associated user interface
  - Provides an interface for entering metadata for any geospatial service including both ArcIMS and ArcGIS Server
  - Wide variety of standards supported
    - Map Viewer—OGC WMS, WFS, and WCS services
    - Metadata Repository Protocols—ArcIMS, Z39.50, Open Archives Initiative, Web-Accessible Folders, and CS-W Catalogs

ArcGIS Server Development Options

ArcGIS Server provides a robust and flexible set of options for deploying and creating Web-based applications. Five different levels of Web application deployment and development are available, ranging from a simple, wizard-based point-and-click setup for nondevelopers to utilizing the ArcGIS Server libraries directly.

- ArcGIS Server Manager Web Application Wizard
  - Point-and-click Web interface allows easy creation of mapping applications without programming.
- Web Mapping Application—Integrated Development Environment Template
  - Once you've created the application in Manager, you can edit it in an IDE, or you can start from the Web Mapping Application template in the IDE without going through the Manager wizard.
- Web Controls
  - Both the Java and .NET ADFs include controls such as maps and toolbars that can be dragged and dropped within the IDE to design and build custom GIS Web applications.
- Utilizing ArcGIS Server Libraries Directly
  - Push fine-grained calls into GIS server for improved performance.
  - Utilize Visual Basic, C++, or .NET.
  - Server Object Extension (SOE)—Allows the server to initialize and manage the extended functionality along with the core server object. Thus, developers don't need to initialize server objects. Also, if the functionality required by a server object extension connects to a database, the cost is paid only once when the server object is initialized.
- Extend the Web ADF
  - Add tools to the toolbar—Write custom code in a class library.
  - Develop new tasks—Built on a common framework in the Web controls library.
  - Support new data sources.

3.4.2.3 Mobile Solutions

One of the most powerful aspects of SOA is its ability to extend the reach of application functionality to the most remote edges of the enterprise. The ArcGIS Mobile Web services foundation enables customers to deploy on any Windows Mobile platform device from Tablet PCs to cellular phones. This frees customers from the limitations of hardwired mobile applications by eliminating the inherent deployment, maintenance, and integration problems.

- ArcPad
  - Complete mobile application
- ArcGIS Mobile
  - Works with ArcGIS Server .NET development framework to create focused, custom mobile applications (also referred to as Mobile ADF)
  - Supported on Windows XP and mobile operating systems
- ArcWeb J2ME Mobile Toolkit
  - Can build ArcWeb Services applications for Java-enabled cell phones and PDAs
  - Contains a set of classes that enables developers to build location-based services applications for mapping, geocoding, routing, address mapping, and points of interest searches using ArcWeb Services
  - Includes functionality to process and save requests

3.4.3 Services

ESRI is strongly committed to creating interoperable systems and services. ESRI supports 96 registered OGC specification implementations as of June 2007 (see current registered components at http://www.opengeospatial.org/resource/products/#ESRI).

To improve system interoperability, with the 9.2 release of ArcGIS, ESRI has fully incorporated ArcSDE® functionality into its Direct Connect, allowing for storage of geospatial information in a variety of RDBMS platforms. A summary of some key ESRI-supported services is shown in table 3.

J-9662

Table 3
Overview of ESRI-Supported Services

Datacentric Services						
Service Name	**OGC**	**SOAP**	**XML over HTTP**	**Direct Link**	**ArcIMS**	**ArcGIS Server**
Web Map Service	x	**	x		x	x
Styled Layer Descriptor (SLD)	x		x		x	*
Web Feature Service	x	**	x		x	*
Transactional Web Feature Service (WFS-T)	x	**	x			*
Web Coverage Service	x	**	x			*
Web Catalog Service	x	x	x		x	
Google (KML)	**		x			x
ODBC				x	x	x
JDBC				x	x	x
RPC				x	x	x
Replication			x			x
Data Interoperability		x	x	x	x	x
Tracking			x	x	x	
Events			x	x	x	**

Businesscentric Services						
Service Name	**OGC**	**SOAP**	**XML over HTTP**	**Direct Link**	**ArcIMS**	**ArcGIS Server**
Routing		x	x		x	x
Network Analysis		x	x			x
3D Globe		x	x			x
Gazetteer		x	x		x	
Geocoding		x	x		x	x
Place Finder		x	x			
Weather		x	x			
Line of Sight		x	x			x
Plume Modeling		x	x			x
Workflow Management		x	x		x	x

* In 9.3 Development
** Future Development

Table 4
ArcIMS Web Services Summary

Capability	What It Does	Required Resource
Image Service WMS	Delivers map content to a client application as JPEG, PNG, and GIF images. A new map image is generated each time a client requests new information. Clients that use image services include the Web mapping application, HTML viewer, Java custom and standard viewers, ArcMap, and ArcPad, among others.	ArcXML file created with Author tool
ArcMap Image Service	A special type of image service that lets you publish maps you create in ArcMap or ArcReader™. Its behavior is similar to that of image services.	Map document (.mxd) or published map document (.pmf)
Feature Service WFS	Delivers map content to the user as streamed features. Feature streaming is a temporary compressed format that remains only as long as the client is open. Requests are sent to the spatial server only when additional data is needed. Clients that can read this stream include the Java standard and custom viewers, ArcExplorer™—Java Edition, and ArcMap.	ArcXML file created with Author tool
Metadata Service CS-W—ebXML	A metadata service lets you and your users share metadata—information about a dataset. Metadata for geographic data typically describes the content, quality, type, collection date, and spatial location of the actual data. Looking at metadata can help your users determine whether a particular dataset is appropriate for their needs.	Metadata configuration file

J-9662

Table 5
ArcGIS Server Web Services Summary

Capability	What It Does	Required Resource
Mapping (2D)	Provides access to the contents of a map document. This capability is always enabled when you publish a map service.	Map document (.mxd) or published map document (.pmf)
WMS	Uses a map document to create a service compliant with the Open Geospatial Consortium's Web Map Service specification.	Map document
Mobile Data Access	Allows extraction of data from a map document to a mobile device.	Map document
KML	Uses a map document to create Keyhole Markup Language features.	Map document
Geoprocessing	Provides access to geoprocessing models from either a toolbox or a tool layer. A tool layer represents a model that has been dragged from ArcToolbox™ and dropped into a map document's table of contents. Enabling this capability while publishing a map document creates an associated geoprocessing service. This capability is always enabled when you publish a geoprocessing service.	Toolbox (.tbx) or a map document with a tool layer
Network Analysis	Solves transportation network analysis problems using the ArcGIS Network Analyst extension.	Map document with a network analysis layer
Globe (3D)	Provides access to the contents of a globe document. This capability is always enabled when you publish a globe service.	Globe document (.3dd)
Geodata	Provides access to the contents of a geodatabase for data query, extraction, and replication. This capability is always enabled when you publish a geodata service.	ArcSDE connection file (.sde), personal geodatabase (GDB), file GDB, or a map document with a layer from a GDB
Geocoding	Provides access to an address locator. This capability is always enabled when you publish a geocode service.	Address locator file (.loc), ArcView 3 locator (.mxs), ArcSDE locator, personal GDB locator, file GDB locator
Workflow Management	Provides an integration framework for ArcGIS multiuser geodatabase environments. It simplifies and automates many aspects of job management and tracking and streamlines workflow.	ArcGIS Server
Data Interoperability	Enables Web applications to directly access hundreds of data formats. The extension also provides access to data translation tools and brings spatial extraction, transformation, and loading (ETL) capabilities to custom server applications via a geoprocessing framework.	ArcGIS Server

Tables 4 and 5 summarize services provided by ArcIMS and ArcGIS Server, respectively. ArcGIS Server Web services support both SOAP and binary messaging formats. SOAP is a common Web service messaging protocol. Binary is used by certain ArcGIS client applications, such as ArcMap and ArcGlobe™, to view services.

Some small reference implementations have attempted utilizing OGC services as the framework for supporting an SOA. WMS operates as the service consumer, WFS operates as the service producer (with GML output), and CS-W operates as the directory service.

3.4.4 Service Support

The primary service support components are

- Directory
- Security
- Management
- Orchestration
- Semantics

3.4.4.1 Directory

ESRI is playing an active role in helping shape effective geospatial Web service directories. The ArcIMS 9.2 metadata service can facilitate Web service directory solutions through the use of a new OGC catalog service for the Web (CS-W 2.0.1) with ebXML profile. This is an extremely effective way for organizations starting SOA implementations to combine GIS domain-specific metadata with IT Web service metadata.

ESRI is also working closely with organizations and vendors on solutions for incorporating effective UDDI directories for storing simple geospatial Web service information. The current UDDI standard does not support range searches, and therefore, simple geospatial queries are not feasible unless utilizing optional proprietary vendor-specific extensions. One potential approach is to populate a common repository, which then utilizes support services (such as an ESB) to synchronize UDDI registries and OGC catalog services. This allows application users and developers to submit geospatial searches to the catalog service while simultaneously offering standard UDDI search capabilities.

Another area of work for UDDI usage is developing, registering, and publicizing tModels pertinent for all service or resource types that are to be referenced by the geospatial community. These tModels can be used to represent standard interfaces, classification information, and namespaces.

ESRI provides directory flexibility by supporting both ebXML and UDDI standards-based solutions. This lets organizations choose the right solution independently of vendor requirements. Once you decide on which standard(s) to utilize for a directory solution, there are a variety of directory implementation options to choose from:

- Registry (UDDI)
- Index (e.g., Google Web crawl)
- Peer-to-Peer (e.g., Distributed Geospatial Intelligence Network [DGINet])
- Federated (e.g., Geospatial One-Stop [GOS] 2)

The key difference between the registry approach and the index approach is not merely the difference between a registry itself and an index in isolation. Indeed, UDDI could be used as a means to implement an individual index: just spider the Web and put the results into a UDDI registry. Rather, the key difference is one of control: who controls what and how service descriptions are discovered. In the registry model, the owner of the registry controls this. In the index model, since anyone can create an index, market forces determine which indexes become popular. Hence, it is effectively the market that controls what and how service descriptions get discovered.

Peer-to-peer (P2P) computing provides an alternative that does not rely on centralized registries; rather, it allows Web services to discover each other dynamically. Under this view, a Web service is a node in a network of peers, which may or may not be Web services. At discovery time, a requester agent queries its neighbors in search of a suitable Web service. If any one of them matches the request, it replies. Otherwise, each queries its own neighboring peers and the query propagates through the network until a particular hop count or other termination criterion is reached.

Because of their respective advantages and disadvantages, P2P systems, indexes, and centralized registries strike different trade-offs that make them appropriate in different situations. P2P systems are more appropriate in dynamic environments in which proximity naturally limits the need to propagate requests such as ubiquitous computing. Centralized registries may be more appropriate in static or controlled environments where information does not change frequently. Indexes may be more appropriate in situations that must scale well and accommodate competition and diversity in indexing strategies.

As service directories propagate, situations will arise where multiple registries and indexes exist on the Web. In such an environment, Web service requesters that must use a discovery service may need to obtain information from more than one registry or index, resulting in a federated approach. Federation refers to the ability to consolidate the results of queries that span more than a single registry or index and make them appear like a single service.

Note that each registry or index may provide a Web service for discovery, so it may be appropriate to use a choreography or orchestration description language to describe the exchanges among these services needed for federation.

3.4.4.2 Security

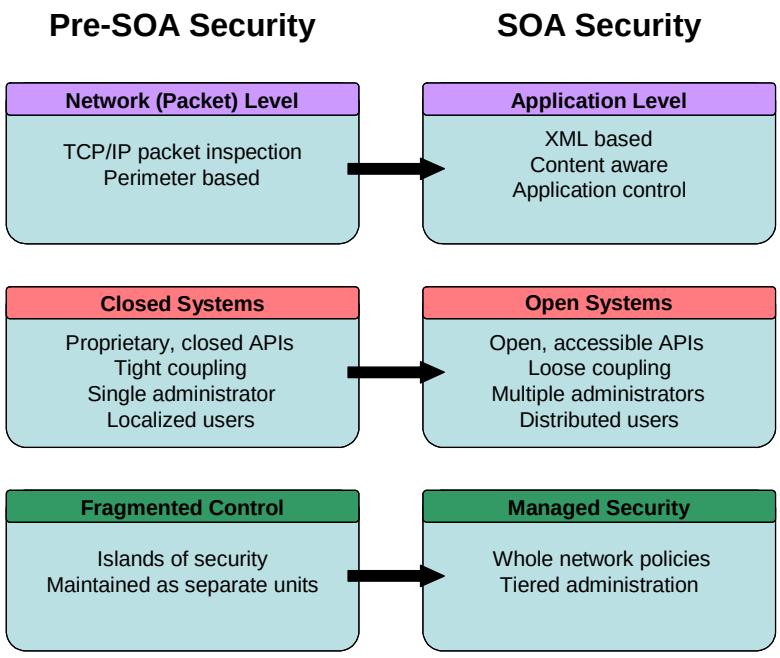
ESRI has always recognized the importance of security, but with increased data sharing occurring in an SOA, security's importance in the geospatial field has increased. Over the last year, ESRI has taken some major steps to ensure secure solutions are provided to customers including

- Releasing a security white paper discussing the options available to secure implementations
- Adding new ArcIMS data access restrictions such as feature limits and restricting geometry in responses
- Hosting multiple Federal Information Security Management Act (FISMA) security certified and accredited sites based on ArcGIS Server technology

- Offering a two-day security workshop that facilitates identifying security options, best practices, and emerging solutions and providing a high-level security assessment
- Demonstrating a GeoDRM viewer as part of the OGC Open Web Services (OWS)-4 demonstration at the end of 2006
- Providing sample security implementation guidance for different security-risk-level environments
- Providing geospatial Enterprise JavaBeans that can utilize standard J2EE security mechanisms

ESRI believes a security in-depth approach is a key principle to ensure secure solutions. Details of this approach are provided in the resources listed above. Securing Web services in an SOA presents new challenges and opportunities for providing secure solutions. Figure 18 provides a brief overview of how security components and approaches are changing with Web services.

Figure 18
Web Service Security



As mentioned in the general SOA security discussion, WS-Security is a key component to providing an end-to-end secure solution in an SOA. WS-Security can be integrated with ArcGIS published Web services consumed by SOAP (not ArcGIS) clients. A custom ArcGIS extension for ArcGIS clients would be required for performing the secure hash operations currently not performed by core ArcGIS.

Communications between client and service can be enhanced using WS-Security standards to ensure that requests have not been altered and cannot be intercepted by third

parties. Confidentiality, integrity, authentication, and authorization can all be provided through WS-Security standards.

Once custom ArcGIS Web service WS-Security controls have been designed into an application, out-of-the-box ArcCatalog™, ArcMap, and ArcGIS Engine software-based applications can no longer consume the secure Web service. However, the clients can still consume the binary message format from ArcGIS Server (ArcGIS Server allows either SOAP, binary, or both messaging formats to be exposed to clients for service access). ArcCatalog, ArcMap, and ArcGIS Engine software-developed applications need a custom ArcObjects interface to administer and consume a WS-Security-controlled Web service. Look for future releases of ArcGIS Server to support WS-Security standards from ArcGIS consumers.

For Microsoft customers, since the ArcGIS 9.2 .NET platform is built on .NET Framework 2.0, customers may choose to implement WS-Security solutions via either Web Services Enhancements (WSE) 3.0 or the new Windows Communication Foundation.

3.4.4.3 Management

ArcGIS Server administrators can use either the Web-based Server Manager or rich client ArcCatalog to publish their GIS resources as services. Interoperability between diverse systems can be supported via dynamic transformations with the ArcGIS Server Data Interoperability extension.

Server Manager

Server Manager is a Web application that supports publishing services, administering the GIS server, creating Web applications, and publishing ArcGIS Explorer maps on the server. Server Manager is the application you use to publish information on your GIS server. From Server Manager, you can add and remove services, edit service properties, and organize services in folders. Server Manager comes with an easy-to-use wizard for creating a Web mapping application. It also includes mechanisms for publishing ArcGIS Explorer maps and KML network links on your server. Finally, Server Manager allows you to configure the machines and directories in your server system and troubleshoot the server using its logs.

Note that .NET ADF implementations may be monitored via standard Windows system service performance monitors, while Java ADF exposes Server Manager functions through Java Management Extensions (JMX) for command-line-level monitoring and management of services directly against the server object manager (SOM) of ArcGIS Server. JMX facilitates the centralized management of managed objects in an enterprise. Java Management Extensions for management and monitoring are an optional extension to the standard Java Development Kit (JDK) and can be used in place of simple network management protocol (SNMP).

ArcCatalog

ArcCatalog is a rich client user interface to GIS servers. ArcCatalog lets you view and, if you're an administrator, manage the set of services running on the server. ArcCatalog provides two distinct views of your GIS server, one for administrators and one for those with consumer privileges.

When you make a user connection to a GIS server, ArcCatalog simply displays the list of services available to you. You can use these services (for example, display a map hosted on the server in ArcMap), but you can't manage them in any way (for example, delete them). When you connect to a GIS server as an administrator, you'll see some extra tools that allow you to manage the services as well.

ArcCatalog lets you administer the set of services running on your server and the set of machines that comprise the server. You can monitor how client applications consume individual services and whether there are enough resources to satisfy demand. At times, you may need to increase the amount of computer resources allocated to a particular service; other times, you may need to add new computers to handle the load.

Transformers

The ArcGIS Server Data Interoperability extension enables custom ArcGIS Server applications to directly access hundreds of data formats. The extension also provides access to data translation tools and brings spatial ETL capabilities to custom server applications via the geoprocessing framework.

3.4.4.4 Orchestration

With the ArcGIS 9.2 release, ESRI provides two solutions to facilitate the orchestration of services and end user workflow. Both of these solutions are built on top of the ArcGIS Server platform and are exposed by standard SOAP end points, which may then be called by an orchestration product that utilizes BPEL to coordinate Web service calls:

- ArcGIS Server Job Tracking extension—An optional user workflow extension to ArcGIS Server
- Geoprocessing service—Included with the Advanced edition of ArcGIS Server

ArcGIS Server Job Tracking Extension

ArcGIS Server Job Tracking extension is an enterprise workflow management extension that expands the job management and tracking functionality of the core product to the Web. The ArcGIS Server Job Tracking extension complements Job Tracking for ArcGIS for the desktop by allowing GIS analysts to cost-effectively organize, standardize, and streamline project workflows on their desktop and publish them to ArcGIS Server Job Tracking extension using integrated tools. Workflow management functions can then be delivered as services throughout the enterprise.

- Provides browser-based access to job status, creation, and assignment
- Enables creation of management and reporting tools for non-GIS users
- Enterprise application integration
  - Accessible to local and distributed consumers
  - Thin (Web) or thick (desktop) client applications
- Provides the foundation for workflow and tasks, enabling a service-oriented architecture

- Centralizes all job-related metadata for on-demand review of information, status, and distribution of the work being performed via the Web

Geoprocessing Service

The new geoprocessing functionality of ArcGIS Server allows users to create standard processes within ModelBuilder™ that may then be uploaded to ArcGIS Server for execution. This allows users to execute complex, fine-grained orchestrated processes server side, while the end user only needs to fill out a simple "task" interface. Geoprocessing services can be used to create tasks in applications based on both ArcGIS Explorer and Web ADF.

ModelBuilder coordinates *fine-grained* calls to components and, therefore, does not use BPEL, which was designed for calls to *coarse-grained* Web services. However, models created with ModelBuilder and set up as a geoprocessing service can be consumed directly through its WSDL. Multiple geoprocessing Web services could then be orchestrated to accomplish more sophisticated and workflow-based analysis.

3.4.4.5 Semantics

ESRI is assisting efforts to move toward a semantic Web; however, there are still considerable research efforts remaining before the full semantic Web is available in the GIS domain.

- ESRI provides generic solutions for serving geospatial and tabular content using Web services.
  - In the case where data models match, information sharing is possible with minimal customization.
  - In the case where data models are not the same, ESRI provides a number of tools to assist with data management activities:
    - Out-of-the-box geoprocessing and data export/import tools
    - ArcGIS Data Interoperability extension for more complex spatial ETL processing
    - Programming language support for developers
  - ESRI is currently considering packaging a set of standardized national information exchange model (NIEM) message services for developers and actively looking for partners and projects in this area.
- ESRI is in the process of working to support a number of organizations on scenarios/specifications for NIEM, specifically
  - Work with the state chief information officer (CIO) council (nascio.org) on scenarios that include geospatial data management processes.
  - Work with the Department of Homeland Security and Federal Geographic Data Committee (FGDC) Homeland Security Working Group on a geospatial data

model that is published through FGDC (http://www.fgdc.gov/fgdc-news/geo-data-model).

- Currently, ESRI publishes a GIS for the Nation data model (www.esri.com/datamodels) that supports many NIEM domains, but we are in the process of developing specific content that corresponds to NIEM domains/communities of interest. The approach and content are evolving, and we are looking for feedback on how to proceed in this area.

3.4.5 Producers

ESRI provides interoperable solutions for many platforms and products.

- Internal Data—Datasets may be stored on a variety of RDBMS products, in addition to no-cost SQL Server Express for workgroup-sized implementations, and the new file-based storage option that allows independence from RDBMS systems.
- Enterprise Application Servers—Adapters are available for facilitating integration with a variety of enterprise application server products including iWay adapters, WebSphere, and SAP.
- Hosting Providers—ArcWeb Services is providing Web 2.0 services including Adobe Flex and Scalable Vector Graphics (SVG) vector outputs.
- Search Engines—ESRI provides a metadata search service with ArcIMS that has industry-standard interfaces for ebXML and OGC's CS-W 2.0.

3.5 ArcWeb Services SOA

ArcWeb Services enable businesses to use geospatial technology via Web services even if they don't have the typical GIS back-end infrastructure (staff, hardware, software) and GIS is not a primary focus of their business or department.

ArcWeb Services APIs include:

- JavaScript—Customize ArcWeb Explorer based on Adobe Flex.
- REST—Customize ArcWeb Explorer even further.
- SOAP—Maximum customization options.
- OpenGIS Location Services (OpenLS) and IBM WebSphere Everyplace Access Location Aware Services (WEA LAS)—Mobile applications.

ArcWeb Explorer is a Web-based map viewing application based on a vector graphics file format called SWF that is natively supported by Adobe's Flash Player as either a browser plug-in or stand-alone implementation. The vector mapping technology enables maps to render quickly in the browser instead of having the server render them. The ArcWeb Explorer public API makes it easy to create custom user interfaces (UI) and mashups. In addition, ArcWeb Explorer includes some automatic processing capabilities to make mashups easy for nonprogrammers. For example, a user uploads an Excel spreadsheet with addresses; the back-end application automatically geocodes the data and sends it back to the browser to be mapped as vectors: mashups without coding.

ArcWeb Services also can output SVG, which is a standard from WC3; however, competing implementations such as SWF currently dominate the market. Vector data from ArcWeb Services can be transformed easily into KML format to integrate into Google mashups and vice versa. SWF and SVG support in ArcWeb Explorer gives businesses a way to retain the value of vector graphics but still keep their data private. An overview of all ArcWeb Services APIs is available at http://www2.arcwebservices.com/v2006/images/awxapi.pdf.

3.6 New ESRI Technologies Facilitating SOA

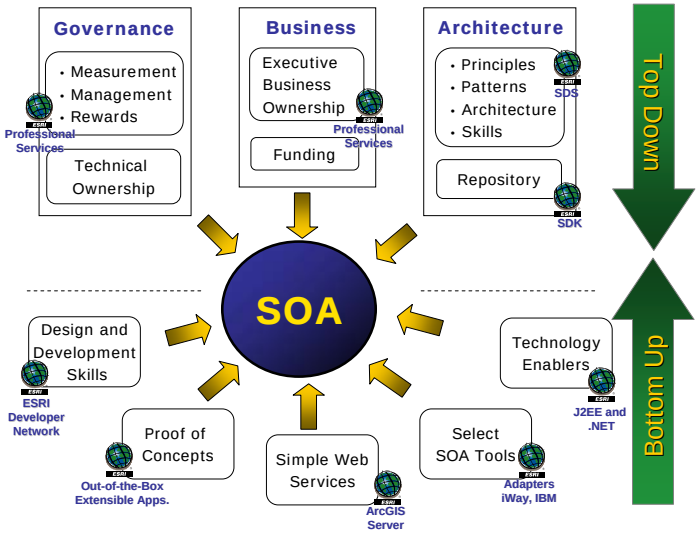
- New Levels of Service Granularity
  - Server Object Extensions
    - Push fine-grained calls into the GIS server for improved performance.
    - Support SOA ideal of Web services focusing on coarse-grained calls.
  - Geoprocessing Tasks
    - Facilitate execution/orchestration of complex tasks via a simple UI.
    - Tasks encapsulate a particular piece of GIS functionality, such as querying or editing, with a user interface in a format that you can easily add to your Web application. Tasks make it easy for the end user of the application to perform certain functions, and in many cases, they facilitate the developer's job because they can be added to a Web application without writing any code. You can configure the Web mapping application mentioned above to contain tasks that help the end user run geoprocessing models, edit data, query attributes, and find places and addresses. These tasks are available whether you're building your application in Server Manager or an IDE. The Web ADF contains classes for developing custom tasks based on user and functional need in addition to the ones mentioned above.
- SOAP/WSDL for ArcGIS Server Services
  - ArcGIS Server exposes its services via industry-standard SOAP with WSDL descriptions automatically as services are established.
- ArcGIS Server Enterprise JavaBeans
  - Advantages
    - Accessible by many types of clients
    - Integrates well with J2EE systems
    - Clear separation of business from client tier
    - Easily distributed
  - Disadvantages
    - Adds another layer and interfaces to code
    - Not suitable for all designs

- Advanced Image Processing on the Fly
  - ArcGIS Image Server provides
    - Fast access and visualization of large quantities of file-based imagery
    - Near-instant display of output imagery for a number of users working simultaneously, without needing to preprocess the data and load it into a DBMS
    - On-the-fly image enhancement, orthorectification, pan sharpening, and complex image mosaicking
- Web 2.0 Functionality
  - AJAX—The end user experience is improved through seamless pan and zoom via an AJAX-based map view out of the box.
  - Fusion—Datasets/Maps may be fused from multiple data sources, either server side or client side.
  - Cached map services—For greatly improved drawing speeds, you can use ArcGIS Server to create a disk cache of prerendered map tiles. These cached map services are useful for basemaps, imagery, and other data that is unlikely to change often or needs to display quickly.

3.7 ESRI SOA Delivery Strategy

To implement a GIS SOA solution, a customer will need to not only research what technical options are best for deployment but also what its business processes are and how to best govern them. Figure 19 provides an overview of the various components that feed the design of a GIS SOA and how ESRI can help with each.

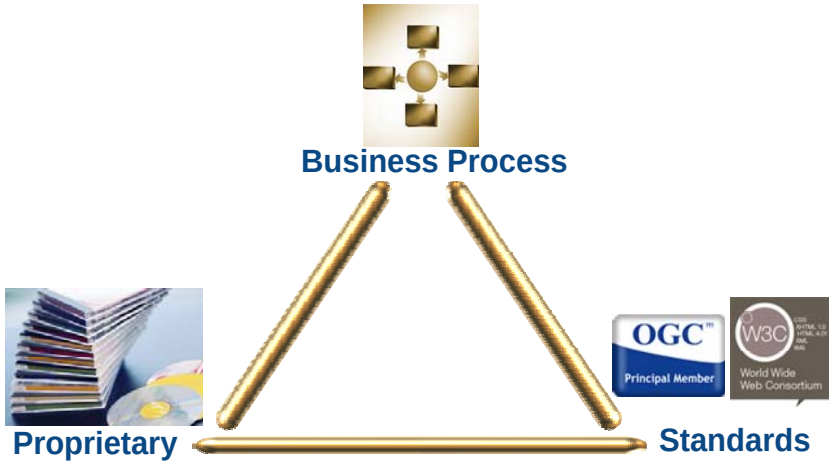
Figure 19
How ESRI Can Facilitate SOA Deployment



The actual deployment phases for each organization will vary depending on its specific needs; however, some general SOA deployment phases are shown below:

- Point-to-Point Integration
  - Understand the reality of where most implementations are now.
  - Provide standardized interfaces (SOAP/WSDL/XML).
    - Utilize adapters on legacy systems with proprietary interfaces.
    - ArcGIS 9.2 offers standard interfaces out of the box.
  - Establish a single sign-on and security infrastructure.
  - Balance the needs of business processes versus functionality offered with proprietary services versus standards-based services (shown in figure 20).
    - ESRI supports both standard OGC services and proprietary solutions that offer additional functionality.
    - Choose your solution based on your business process needs.

Figure 20
Choosing Services



- Loosely Coupled Services
  - Create a governance framework.
  - IT can list human service owners that manage versioning, ensure compliance with enterprise requirements, and more.
  - Establish registry/repository services.

- UDDI/ebXML
  - ♦ ArcIMS 9.2 provides OGC CS-W 2.0.1 and OASIS ebXML interfaces.
  - ♦ ArcGIS Server 9.2 generates WSDL description files for UDDI.

■ Reliable, Discoverable Services

- Build messaging infrastructure for large SOA deployments.
  - ♦ ESB and application servers fill this role.
  - ♦ Small-scale SOA deployments may utilize direct, synchronous XML.
- Expand service management infrastructure.
  - ♦ Web Services Distributed Management (WSDM) applications.

■ Composable, Reusable Services

- Start incorporating service workflow management.
- Use Business Process Execution Language.

■ Enterprise SOA

- It provides principles of service orientation throughout the organization, allowing true agility.
- Service federation enables groups to establish a trust relationship, automating sharing.

3.8 Scalable ESRI SOA Solutions

ESRI's ArcGIS Server technology supports any size implementation from a single server solution incorporating external Web services to distributed, high-availability, large enterprise solutions. Details on how ESRI-specific components scale may be found in the technical reference document *System Design Strategies.* This section provides an overview of how ESRI components are deployed relative to other SOA components.

3.8.1 Overview of ArcGIS Architecture

An ArcGIS Server system may consist of the following components:

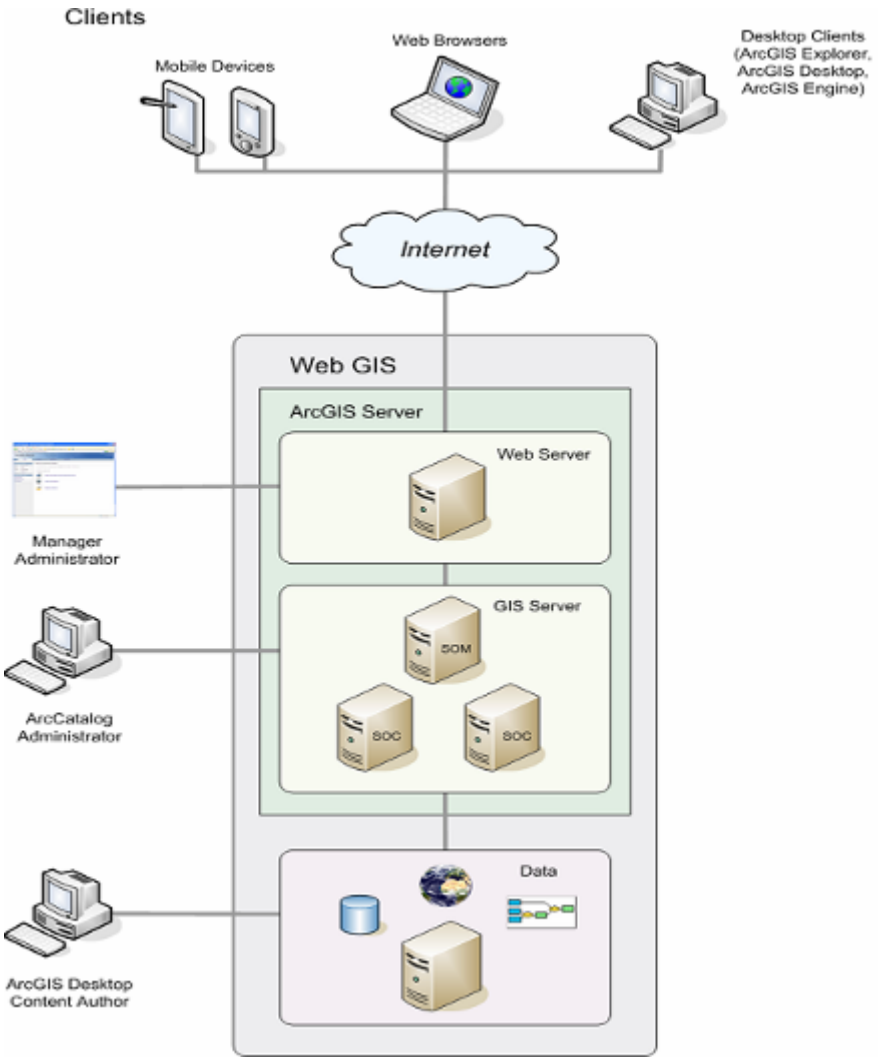
■ GIS server

- The GIS server hosts your GIS resources, such as maps, globes, and address locators, and exposes them as services to client applications.
- The GIS server is composed of two distinct parts: the server object manager and the server object container (SOC). The SOM manages the services running on the server. When a client application requests the use of a particular service, the SOM assigns the request to an available server object. There is only one SOM per GIS server.
- The SOM connects to one or more SOCs. The SOC machines—also referred to as container machines—contain, or host, the services that the SOM manages.

J-9662

Depending on your configuration, you may run the SOM and SOC on different machines and also have multiple SOC machines. Figure 21 shows an SOM machine connected to two SOC machines.

Figure 21
ArcGIS Server System Architecture



- Web server
  - The Web server hosts Web applications and Web services that use the resources running on the GIS server. Interfaces for clients include
    - Web Service End Point—SOAP/HTTP Web service access
    - Tile Handler Interface—Allows clients to directly access cache files stored in a Web server virtual directory

- ♦ Web Application Development Framework—Out-of-the-box Web mapping application functionality

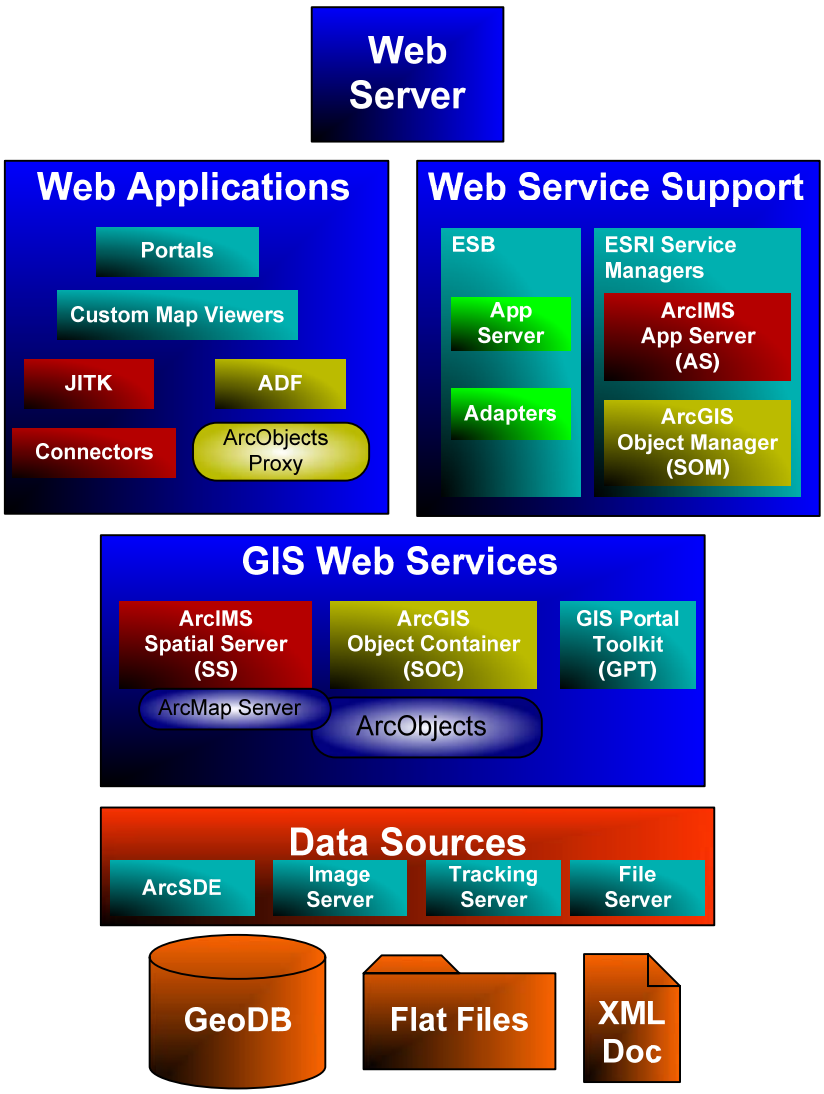
- Clients
  - Client applications are Web, mobile, and desktop applications that connect over HTTP to ArcGIS Server Web services or ArcGIS Server local services over a LAN or WAN.
- Data server
  - The data server contains the GIS resources that have been published as services on the GIS server. These resources can be map documents, address locators, globe documents, geodatabases, and toolboxes.
- Manager and ArcCatalog administrators
  - ArcGIS Server administrators can use either Server Manager or ArcCatalog to publish their GIS resources as services.
  - Server Manager is a Web application that supports publishing services, administering the GIS server, creating Web applications, and publishing ArcGIS Explorer maps on the server.
  - ArcCatalog includes a GIS server node that can be used to add connections to GIS servers for either general server usage or administration of a server's properties and services.
- ArcGIS Desktop content authors
  - To author the GIS resources, such as maps, geoprocessing tools, and globes, that will be published to your server, you will need to use ArcGIS Desktop applications such as ArcMap, ArcCatalog, and ArcGlobe. Additionally, if you're creating 2D map or 3D globe caches to increase rendering performance, you will need to use ArcCatalog to create the cache.

3.8.2 General ESRI SOA Component Deployment Locations

The scalable architecture of ArcGIS Server lends itself to numerous deployment options. Large deployments may require multiple Web servers or reverse proxies, Web application servers, Web service support boxes (core of the SOA infrastructure), Web service boxes (focus processing business logic), and data sources, while smaller implementations might want to consolidate these resources onto one or two machines. Each of these main component groups is broken down into separate components in figure 22.

The way you deploy ArcGIS Server depends on what you want to do with it. If you are using the product for development or testing purposes, you don't need an extensive deployment; however, if you are publishing GIS services to a large community of users, you need to give extra consideration and resources to factors such as processing loads, single points of failure, and security.

Figure 22
ESRI SOA Component Deployment Locations



3.8.3 Single Server Deployment Example

ESRI provides the starting framework for a geospatial SOA so that new or small implementations are not dependent on first purchasing an expensive SOA framework for business logic components to be layered onto.

Customer Needs

- Provide a low-cost, low-maintenance solution.
- Web editing and server-side geoprocessing are required for Internet users.
- Need to be able to layer in datasets from ArcWeb Services along with other OGC Web services.

- Five users are GIS professionals utilizing ArcInfo 9.2.
- Customer prefers Windows-based solutions.

System Configuration

- ArcGIS Server 9.2 Advanced Workgroup edition
- ArcWeb Services and data
- SQL Server Express 2005 (free)—10 local users, unlimited Internet users
- Windows 2003 R2 64-bit Standard edition
- Can purchase a single server solution along with software as a preconfigured package

How Needs Are Addressed

- Utilizes free SQL Server Express 2005, which supports up to 10 local users with direct connect and provides Internet access for additional users.
- The ArcGIS Server Web ADF wizard, within the Manager application, walks customers through setup of Web applications that can utilize and fuse datasets from ArcWeb Services, OGC services such as WMS, and local datasets created by the GIS professionals.

3.8.4 Java Enterprise ESRI SOA Deployment Example

Since ESRI utilizes IT standard Web service protocols, implementations can utilize additional third-party SOA infrastructure software to meet large enterprise needs. Java customers can utilize J2EE 1.4 certified Web application servers.

Customer Needs

- Incorporate live data feeds and represent filtered information geospatially.
- Currently utilize BEA for portal and message management.
- Geospatial Java Specification Request-168 portlets.

System Configuration

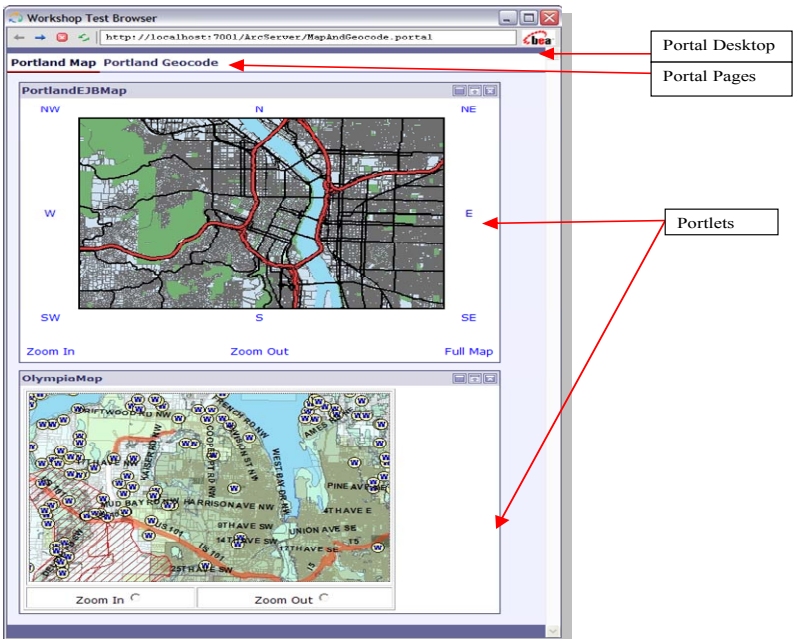
- BEA AquaLogic
  - Portal, ESB, UDDI
- Oracle Real Application Cluster (RAC) 10*g*
  - Solaris 10

- ArcGIS Server 9.2 Advanced Enterprise for Java
  - Windows 2003 R2 64-bit Standard Edition
  - Utilizes ArcGIS Enterprise JavaBeans ADF
- Tracking Server for ArcGIS
- Eclipse IDE

How Needs Are Addressed

- Tracking Server can consume live data feed needs and directly filters and populates data into the geodatabase for display via geospatial portlets in the BEA portal.
  - Note that many customers currently find the Java Specification Request-168 standard too restrictive in the portlet (see figure 23 for an example of a geospatial portlet).

Figure 23
Example BEA Portal with ArcGIS Portlets



3.8.5 .NET Enterprise ESRI SOA Deployment Example

The ArcGIS ADF is built on the latest Microsoft .NET Framework 2.0 platform, allowing for scalable Microsoft-based enterprise solutions.

Customer Needs

- Single sign-on (SSO) solution.
- Wire-speed WS-Security.

- Incorporate geospatial webparts into SharePoint.
- Remote in-the-field users need to update geospatial information and obtain latest information.
- For viewing, 500 GB of new raster data is available weekly.
- Integrated editing of geospatial applications within IDE.
- Agile development process and supporting framework for modeling and incorporating custom functionality.

System Configuration

- Microsoft SharePoint 2003
- Microsoft UDDI
- ArcGIS Server 9.2 Advanced Enterprise for .NET
  - Microsoft Windows 2003 R2 64-bit Enterprise Edition
  - ArcGIS Mobile
- Enterprise Edition SQL 2005
- ArcGIS Image Server
- Microsoft Visual Studio 2005 IDE
- Enterprise Architect
- Microsoft BizTalk
- DataPower XML Security Gateway XS40

How Needs Are Addressed

- SSO needs are addressed by new Active Directory Federation Services as part of Windows 2003 R2.
- External access Web service security is addressed by utilizing XML Security Gateway as a security enforcement point for XML and Web service transactions including encryption, firewall filtering, digital signatures, schema validation, XML access control, and XML Path Language (XPath).
- Customer is able to create a custom, easy-to-use Web application for mobile users with the ArcGIS Mobile ADF, allowing for incremental downloading and uploading of in-the-field edits.

- ArcGIS Image Server solution allows customer to incorporate large dynamic raster datasets as a backdrop image without waiting for data to be preprocessed.
- Web services and application integrated into Visual Studio 2005 IDE via both ESRI ADF and Microsoft UDDI.
- For custom code development, case studies are modeled in UML via the Iconix agile development process with Enterprise Architect. This is then utilized to help facilitate code generation in Visual Studio as part of a model-driven architecture.
- SQL 2005 Enterprise is utilized to replicate UDDI registries across the organization and to external shared registry partners.

3.9 ESRI SOA Summary

Organizations today are being challenged to be more efficient, accurate, and accessible. To deal with these challenges, IT departments are moving increasingly toward SOAs to provide a framework for technology, policies, and practices by which they can be organized in an effort to deliver the right services to the right people at the right time in the right place.

GIS is a proven and valued technology that plays an important role in today's SOA strategies and initiatives. With desktop and server-based GIS solutions, such as ArcGIS Desktop and ArcGIS Server, organizations can integrate GIS into their existing workflows and solve today's challenges of providing open access to common geospatial data, services, and applications from within and beyond.

By providing key technologies and supporting interoperability with standards such as .NET 2.0, Enterprise JavaBeans, WS-I-compliant Web services, and OGC services out of the box, the ArcGIS Server technology platform is ideally suited to SOA deployment.

Appendix A: Glossary of Terms

asynchronous (W3C)—An interaction is said to be asynchronous when the associated messages are chronologically and procedurally decoupled. For example, in a request-response interaction, the client agent can process the response at some indeterminate point in the future when its existence is discovered. Mechanisms to do this include polling and notification by receipt of another message.

choreography (W3C)—Defines the sequence and conditions under which multiple cooperating, independent agents exchange messages to perform a task to achieve a goal state.

DCOM (Distributed Component Object Model)—A set of Microsoft concepts and program interfaces in which client program objects can request services from server program objects on other computers in a network.

HTTP/HTTPS (Hypertext Transfer Protocol/Secure Hypertext Transfer Protocol)—Standard communication protocol for Web servers; Web services most commonly rely on this protocol, layering others on it (e.g., SOAP messages transmitted over HTTP).

IIOP (Internet Inter-ORB Protocol)—A protocol that makes it possible for distributed programs written in different programming languages to communicate over the Internet.

J2EE (Java 2 Enterprise Edition)—A Java platform designed for the mainframe-scale computing typical of large enterprises.

J2ME (Java 2 Micro Edition)—A technology that allows programmers to use the Java programming language and related tools to develop programs for mobile wireless information devices such as cellular phones.

JMX (Java Management Extensions)—A set of specifications for application and network management in the J2EE development and application environment. JMX defines a method for Java developers to integrate their applications with existing network management software by dynamically assigning Java objects with management attributes and operations.

loose coupling (W3C)—Coupling is the dependency between interacting systems. This dependency can be decomposed into real dependency and artificial dependency:

- Real dependency is the set of features or services that a system consumes from other systems. The real dependency always exists and cannot be reduced.

- Artificial dependency is the set of factors that a system has to comply with to consume the features or services provided by other systems. Typical artificial dependency factors are language, platform, API, and so forth. Artificial dependency always exists, but it or its cost can be reduced.

Loose coupling describes the configuration in which artificial dependency has been reduced to the minimum.

managed code—Code that has its execution managed by the .NET Framework Common Language Runtime.

MDA (model-driven architecture)—An approach to IT system specification that separates the specification of functionality from the specification of the implementation of that functionality on a specific technology platform.

orchestration (W3C)—An orchestration defines the sequence and conditions in which one Web service invokes other Web services to realize some useful function (an orchestration is the pattern of interactions that a Web service agent must follow to achieve its goal).

protocol (W3C)—A set of formal rules describing how to transmit data, especially across a network. Low-level protocols define the electrical and physical standards to be observed, bit and byte ordering, and the transmission and error detection and correction of the bit stream. High-level protocols deal with the data formatting including the syntax of messages, terminal-to-computer dialog, character sets, and sequencing of messages.

proxy (W3C)—An agent that relays a message between a requester agent and a provider agent, appearing to the Web service to be the requester.

quality of service (W3C)—Quality of service is an obligation accepted and advertised by a provider entity to service consumers.

reference architecture—A reference architecture is the generalized architecture of several end systems that share one or more common domains. The reference architecture defines the infrastructure common to the end systems and the interfaces of components that will be included in the end systems. The reference architecture is then instantiated to create a software architecture of a specific system. The definition of the reference architecture facilitates deriving and extending new software architectures for classes of systems. A reference architecture, therefore, plays a dual role with regard to specific target software architectures. First, it generalizes and extracts common functions and configurations. Second, it provides a base for instantiating target systems that use that common base more reliably and cost-effectively.

registry (W3C)—Authoritative, centrally controlled store of information.

REST (Representational State Transfer)—An approach for getting information content from a Web site by reading a designated Web page that contains an XML file that describes and includes the desired content.

SAML (Security Assertion Markup Language)—XML standard for expressing authentication and authorization.

service—A service is an abstract resource that represents a capability of performing tasks that form a coherent functionality from the point of view of provider entities and requester entities. To be used, a service must be realized by a concrete provider agent.

service description—A service description is a set of documents that describes the interface to and semantics of a service.

service interface—1. A service interface is the abstract boundary that a service exposes. It defines the types of messages and the message exchange patterns that are involved in interacting with the service, together with any conditions implied by those messages.

2. A logical grouping of operations. An interface represents an abstract service type, independent of transmission protocol and data format.

service intermediary (e.g., ESB, BizTalk)—A service intermediary is a Web service whose main role is to transform messages in a value-added way. (From a messaging point of view, an intermediary processes messages en route from one agent to another.) Specifically, a service intermediary is a service whose outgoing messages are equivalent to its incoming messages in some application-defined sense.

service-oriented architecture—A set of components that can be invoked and whose interface descriptions can be published and discovered.

service semantics—The semantics of a service is the behavior expected when interacting with the service. The semantics expresses a contract (not necessarily a legal contract) between the provider entity and the requester entity. It expresses the effect of invoking the service. Service semantics may be formally described in a machine-readable form, identified but not formally defined, or informally defined via an out-of-band agreement between the provider and the requester entity.

session—A lasting interaction between system entities, often involving a user, typified by the maintenance of some state of the interaction for the duration of the interaction. Such an interaction may not be limited to a single connection between the system entities.

SOAP (W3C)—The formal set of conventions governing the format and processing rules of a SOAP message. These conventions include the interactions among SOAP nodes generating and accepting SOAP messages for the purpose of exchanging information along a SOAP message path.

synchronous—An interaction is said to be synchronous when the participating agents must be available to receive and process the associated messages from the time the interaction is initiated until all messages are actually received or some failure condition is determined. The exact meaning of *available to receive the message* depends on the characteristics of the participating agents (including the transfer protocol it uses); it may, but does not necessarily, imply tight time synchronization, blocking a thread, and so forth.

tModel—A data structure representing a service type (a generic representation of a registered service) in the UDDI registry. Each business registered with UDDI categorizes all of its Web services according to a defined list of service types. Businesses can search the registry's listed service types to find service providers. The tModel is an abstraction for a technical specification of a service type; it organizes the service type's information and makes it accessible in the registry database.

UDDI (Universal Description, Discovery, and Integration)—Standard for registering and locating Web services.

Web service (W3C 2002)—A Web service is a software system identified by a uniform resource identifier (URI), whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML-based messages conveyed by Internet protocols.

Web service (W3C 2004)—A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processible format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

WSDL (Web Services Description Language)—Standard means of describing a Web service.

WS-I—The WS-I is an open industry organization chartered to promote Web services interoperability across platforms, operating systems, and programming languages. Specifically, WS-I creates, promotes, and supports generic protocols for the interoperable exchange of messages between Web services.

WS-Security—Set of standards for providing security capabilities within XML messages; incorporates both XML-Encryption and XML-Signature.

XACML (Extensible Access Control Markup Language)—XML standard for access control policies for Web resources.

XML-Encryption—Standard for encrypting data and expressing the result within an XML message.

XML-Signature—Standard for digitally signing portions of or entire XML documents.

Appendix B: OASIS SOA Key Principles

The OASIS standards group has recently provided a reference model for SOA so that there is general consensus on its key aspects:

- **Contract and Policy—**A policy represents some constraint or condition on the use, deployment, or description of an owned entity as defined by any participant. A contract, on the other hand, represents an agreement by two or more parties.
- **Execution Context—**The set of technical and business elements that form a path between those with needs and those with capabilities and that permit service providers and consumers to interact.
- **Interaction—**The activity involved in making use of a capability offered, usually across an ownership boundary, to achieve a particular desired real-world effect.
- **Real-World Effect—**The actual result of using a service, rather than merely the capability offered by a service provider.
- **Service Descriptions—**The information needed to use, or consider using, a service.
- **Services—**The means by which the needs of a consumer are brought together with the capabilities of a provider.
- **Visibility—**The capacity for those with needs (consumers) and those with capabilities (producers) to be able to interact with each other.

Appendix C: List of Acronyms and Definitions

ADF	Application Development Framework
AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
ASP	Active Server Pages
BPEL	Business Process Execution Language
CAD	Computer-Aided Design
CIO	Chief Information Officer
CORBA	Common Object Request Broker Architecture
COTS	Commercial Off-the-Shelf (software)
CRM	Customer Relationship Management
CS-W	Web Catalog Service
DCOM	Distributed Component Object Model
DGINet	Distributed Geospatial Intelligence Network
DNS	Domain Name System
EAI	Enterprise Application Integration
EDA	Event-Driven Architecture
EIC	Enterprise Information Consumer
EIS	Enterprise Information System
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
ESRI	Environmental Systems Research Institute, Inc.
ETL	Extraction, Transformation, and Loading
FGDC	Federal Geographic Data Committee
FISMA	Federal Information Security Management Act
GDB	Geodatabase
GeoDRM	Geospatial Digital Rights Management
GIS	Geographic Information System
GOS	Geospatial One-Stop
GPT	GIS Portal Toolkit
GUI	Graphic User Interface
HTTP	Hypertext Transport Protocol
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IIOP	Internet Inter-ORB Protocol
IIS	Internet Information Server
ISAPI	Internet Server Application Programming Interface
ISO	International Organization for Standardization

IT	Information Technology
J2EE	Java 2 Enterprise Edition
J2ME	Java 2 Micro Edition
J2SE	Java 2 Standard Edition
JCA	J2EE Connector Architecture
JDBC	Java Database Connectivity
JDK	Java Development Kit
JAX-R	Java API for XML Registries
JITK	Java Integration Toolkit
JMX	Java Management Extensions
JSP	JavaServer Pages
JTX	Job Tracking for ArcGIS
KML	Keyhole Markup Language
LAN	Local Area Network
MDA	Model-Driven Architecture
MOM	Message-Oriented Middleware
MOU	Memorandum of Understanding
NIEM	National Information Exchange Model
OASIS	Organization for the Advancement of Structured Information Standards
OGC	Open Geospatial Consortium, Inc.
OMG	Object Management Group
OpenLS	OpenGIS Location Services
OS	Operating System
OWL	Web Ontology Language
OWS	Open Web Services
P2P	Peer-to-Peer
POC	Point of Contact
POX	Plain Old XML
QA	Quality Assurance
QMS	Quality Management System
RDBMS	Relational Database Management System
RDF	Resource Description Framework
REST	Representational State Transfer
ROI	Return on Investment
RPC	Remote Procedure Call
SAML	Security Assertion Markup Language
SDE®	Spatial Database Engine™
SDK	Software Development Kit
SDSFIE	Spatial Data Standard for Facilities, Infrastructure, and Environment
SEI	Software Engineering Institute
SLD	Styled Layer Descriptor

J-9662

SNMP	Simple Network Management Protocol
SOA	Service-Oriented Architecture
SOC	Server Object Container
SOE	Server Object Extension
SOM	Server Object Manager
SSL	Secure Sockets Layer
SSO	Single Sign-On
SVG	Scalable Vector Graphics
SWF	Small Web Format and Shockwave Flash
UDDI	Universal Description, Discovery, and Integration
UI	User Interface
UML	Unified Modeling Language
URI	Universal Resource Identifier
VB	Visual Basic
W3C	World Wide Web Consortium
WAN	Wide Area Network
WCF	Windows Communication Foundation
WCS	Web Coverage Service
WEA LAS	WebSphere Everyplace Access Location Aware Services
WFS	Web Feature Service
WMS	Web Map Service
WS	Web Services
WS-BPEL	Web Services Business Process Execution Language
WSDL	Web Services Description Language
WSDM	Web Services Distributed Management
WSE	Web Services Enhancements (Microsoft)
WS-I	Web Services Interoperability Organization
XACML	Extensible Access Control Markup Language
XML	Extensible Markup Language
XPath	XML Path Language

Appendix D: Comparison of Java and .NET Components

ESRI customers have the flexibility of using a Java or .NET framework to support their enterprise environments. This appendix provides a brief comparison of how components relate to each other between the two frameworks.

Java 2 Enterprise Edition is a specification made up of many component specifications related to developing distributed applications in the Java programming language. J2EE components are used when writing Web-based applications and traditional client-server applications, and to connect to legacy resources like relational databases using a standard API. To those from a .NET development background, the Java Servlets and JavaServer Pages technologies are the components of most interest.

Java Servlets are Java classes that run as an extension of a Web server like IIS or the Apache Web server. A Java Servlet is analogous to an Internet Server API (ISAPI) filter, the .NET HttpHandler class, or a cgi-bin program/script. A Java Servlet runs when a client browser invokes a specifically configured URL directly or indirectly. A servlet has access to all the information in an HTTP request and can handle the request directly by providing content to be returned to the client. Alternatively, a servlet can redirect the client browser to another resource. Most J2EE Web applications use servlets primarily as targets of HTML forms to handle user input and process it accordingly. Generation of the response page is typically delegated to JavaServer Pages.

Development Platform	HTML Document	Code Emitting HTML	Portal Subcomponents
Java	JavaServer Pages (Java)	Java Servlets	Java Specification Request 168 Portlets
.NET	.NET (C#, Visual Basic .NET)	.NET HttpHandler class or ISAPI filter	Web Parts

JavaServer Pages are analogous to .NET pages. That is, they are HTML pages that also contain scripting elements that execute on the server when a user requests the page. A key difference between .NET pages and JavaServer Pages is that .NET pages use a .NET language (such as C# and Visual Basic .NET) as their scripting language, whereas JavaServer Pages use the Java language. Typical JavaServer Pages contain snippets of Java code and some special HTML-like tags defined in the Java Specification Request, interleaved with standard HTML to provide a combination of static and dynamic content. The difference between Java Servlets and JavaServer Pages is conceptually similar to the difference between the .NET HttpHandler class and a .NET page. In both cases, the former is a piece of code that can be used to emit HTML directly or redirect to other resources, and the latter is an HTML document that can contain embedded code.

Appendix E: Geospatial SOA Case Study

ArcGIS Server, IBM WebSphere Process Server, and heritage business information systems

Business Environment

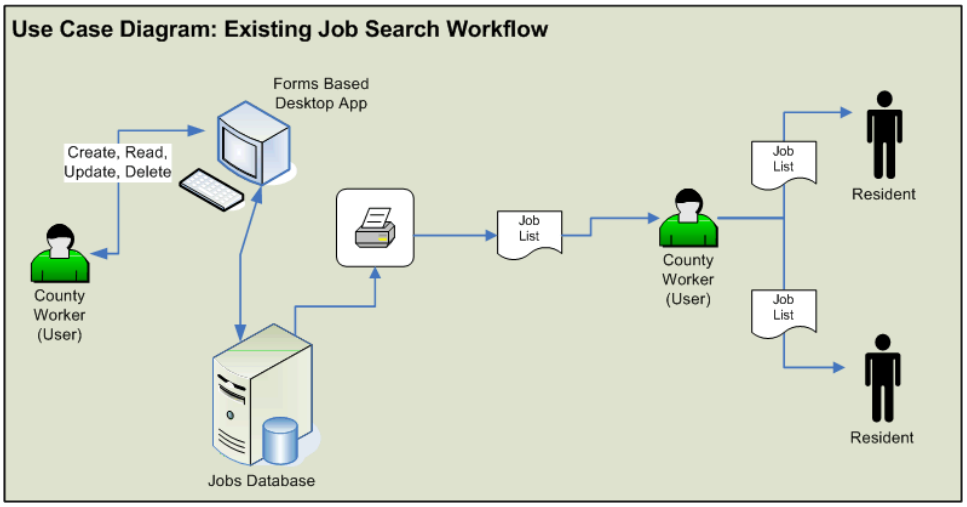
The business environment for this solution is a county government. Within the county, the Human Services Department is responsible for providing a number of resident-facing services as part of its mission. One such service is providing assistance in locating potential job opportunities for residents searching for employment. The county cooperates with local businesses for job placement services based on applicant interests, experience, and skills.

Existing Capabilities

Workers in the Human Services Department currently interact with a heritage, or legacy, client-server database system to store and retrieve information about available jobs within the county. As depicted in the use case diagram in figure 1, county workers interact with the county jobs database through a desktop forms-based client application. They are satisfied with the desktop application as a means to enter and manage job information.

The legacy database system outputs a simple text list of available jobs with associated job information. Workers find this output format unintuitive, hard to use, and not supportive of their efforts in locating the best potential job opportunity for a given candidate. Workers indicate that they often spend more time working with the jobs list than they feel should be necessary because its utility is so limited.

Figure 1
Existing Job Search Workflow



Desired Capabilities

Workers expressed a desire to use a common Web browser to create job listings for residents. In addition to the information from the jobs database, workers desire the ability to create a map display of the returned job locations as well as the location of the job seeker's home address. This capability will improve the system by providing a visualization of the job location in relationship to the job seeker's domicile. This capability enables location to factor into the employment search and decision. During interviews to gather system requirements, users were very enthusiastic about adding location information in the solution and continued defining location-based requirements to include enabling the system to generate transportation routes between the job seeker's residence and selected job opportunities. They also want travel routing algorithms to consider whether or not the job candidate would drive a personal vehicle to and from the job location or if he or she would take public transportation resources. Finally, users expressed a need to include the location of day care facilities as another factor in the identification of available jobs. When assisting residents in locating employment opportunities, county workers often are told that job seekers want child care in close proximity to the job location. The management team in the Human Services Department then defined a business rule stating that for those job candidates who want day care facilities located close by their work site, the county will provide job leads that have a licensed day care facility within two miles of the work location.

Solution Overview

With desired capabilities in hand, the system requirements are framed within a services-oriented design, development, and implementation strategy. The county CIO established architecture guidance and policy that established SOA as the architectural approach. Recognizing the advantages of the SOA approach, the Human Services Department set out to instantiate the new Job Finder application in a manner that aligns with the tenets and principles of SOA. Thus, the new Job Finder services-oriented business application was designed, developed, and deployed in accordance with the CIO's architectural policy and guidance.

Solution Design

Based on the requirements-gathering process and an examination of the existing and desired workflow elements of the planned Job Finder business process, the services-oriented solutions team identified four main tasks for this effort as follows:

1. Service definition and discovery.
2. Design and develop new portfolio services.
3. Design and develop composite services.
4. Design and develop a Job Finder Web client.

The remainder of this case study illustrates the approach and methods used to design, develop, and deploy the Job Finder services-oriented business application using ArcGIS Server and IBM WebSphere Process Server and leveraging heritage business information systems.

Service Definition and Discovery

The project team created an initial list of business functions to deliver the desired capabilities and workflow. Business process storyboards were developed and presented to county workers and department managers to ensure all project stakeholders shared a common vision of the solution. With a high-level definition of the business and technical requirements along with a shared vision of the solution by all stakeholders, the design team initiated a review of the county centralized service registry to discover published business services available across the organization.

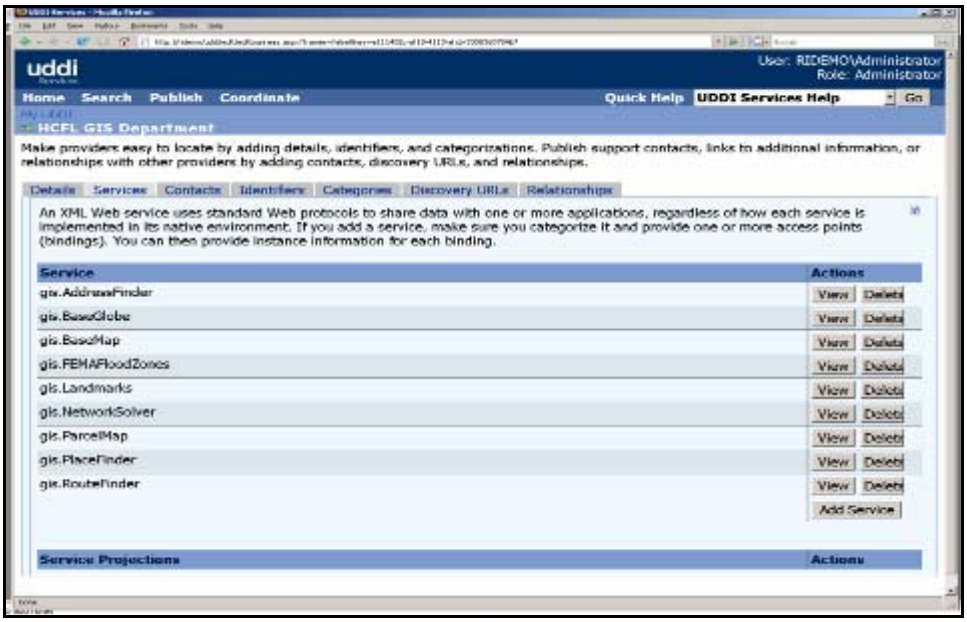
The ability to discover and subsequently reuse applicable services across multiple business systems and processes is one of the underpinnings of SOA. The creation and use of the county government service registry is a key component of the CIO's SOA governance program. Participating business, technology, and other service partners publish and describe the business services they offer in a centralized registry so that the services are effectively managed, discovered, and used. A central registry framework promotes service manageability, discovery, and reusability through a standards-based publish and discover model. The ability to publish, discover, and reuse existing services reduces the time required to locate, understand, and implement available capabilities and is a key advantage to employing a services-oriented approach. Reusability also reduces development time, development-to-deployment cycles, and software maintenance efforts.

The county's service registry employs WSDL to publish, describe, and discover services within a UDDI online registry. Although SOA business systems can be built without the use of a centralized registry, such an approach undermines the value of the SOA. A service registry and governance policies that mandate its use by both providers and consumers increase service use, reuse, and SOA value.

County Registry Search for Service Discovery

Using the county UDDI registry, the development team discovered a set of common GIS and business services applicable to the needs of the Job Finder application. Figure 2 provides a view of the county's UDDI service registry. Depicted in this view are GIS services published and maintained by the county GIS Department. These services are discoverable in the registry because they are published and are being used by other business applications within the county enterprise application portfolio.

Figure 2
UDDI Online Service Registry



In the Job Finder business scenario, the development team can reuse the published services discovered in the centralized UDDI service registry and shown in Table 1:

Table 1
Published Reusable Business Services

Service Name	Owner	Description
gis.BaseMap	GIS Department	This service provides a county basemap consisting of features and objects from the county's enterprise GIS system. The county's ArcGIS Server publishes this service to multiple business applications. In the Job Finder solution, this service will provide the basemap imagery and feature information.
gis.NetworkSolver	GIS Department	This service is a service area algorithm that uses county street centerline information from the enterprise GIS system and calculates optimum route information between two or more points. The service considers travel restrictions, obstacles, public transportation nodes and routes, and other travel time distance factors. The county's ArcGIS Server publishes this service to multiple business applications. In the Job Finder solution, this service will provide travel distance, travel time, and detailed vehicle or public transportation route information between the job seeker's residence and the job location to include a travel stop at a day care facility if required by the job seeker.
crm.CitizenLocator	Human Services Department	This service provides the registered residential address based on the county's tax system. The service accepts a resident's county identification number as input and returns the residential street address on file. In the Job Finder solution, this service receives the resident's unique tax identification number and returns a well-formed address record for the job seeker's place of residence.
crm.JobsList	Human Services Department	This service provides a listing of available jobs within the county. The listings include all relevant job-related information. Jobs are grouped according to defined job categories. These categories provide the integer value that is passed to the service to filter the jobs based on a given job category. The county jobs database is a legacy system that is not natively Web services enabled. A WebSphere ODBC adapter is used to expose the results of a job search as a standards-based Web service. In the Job Finder solution, this service will provide the information pertaining to available jobs to include job location information.
crm.BusinessFinder	Human Services Department	This service provides the registered address of licensed businesses in the county. The service accepts a number of identifying attributes such as business name, business type, and business address to create a list of businesses and their addresses based on the information provided in the service invocation message. In the Job Finder solution, this service will provide a list of registered day care providers in the county based on a business-type database query.

Design and Develop New Portfolio Services

After discovering services in the registry that were deemed reusable for the Job Finder solution, the design team identified one additional business function needed in the Job Finder solution. The team identified the need for a service that performs a geocode operation on the residential address information returned from the crm.CitizenLocator service. A description of this new service is provided in table 2.

Table 2
Description of New GIS Business Service

Service Name	Owner	Description
gis.GeocodeService	GIS Dept.	This common service is used to send an address to the server and to return a geographic location—such as a point representing a house. This GeocodeService can be used to match a single address, an array of addresses, or a list of addresses contained in a file. In the Job Finder solution, this service will transform resident address information into geographic x- and y-values for mapping.

Designing Services for Reuse

Enterprise technology services should be flexible, meeting current business needs and providing a foundation to meet new requirements. Designing flexible services enables the enterprise architecture to remain agile and accessible to business change and business opportunity. Service flexibility achieves the services-oriented design objective of service reuse, which is the most prevalent way that SOA adopters have recognized the SOA value proposition of faster development cycles, reduced development costs, and reduced software bugs.

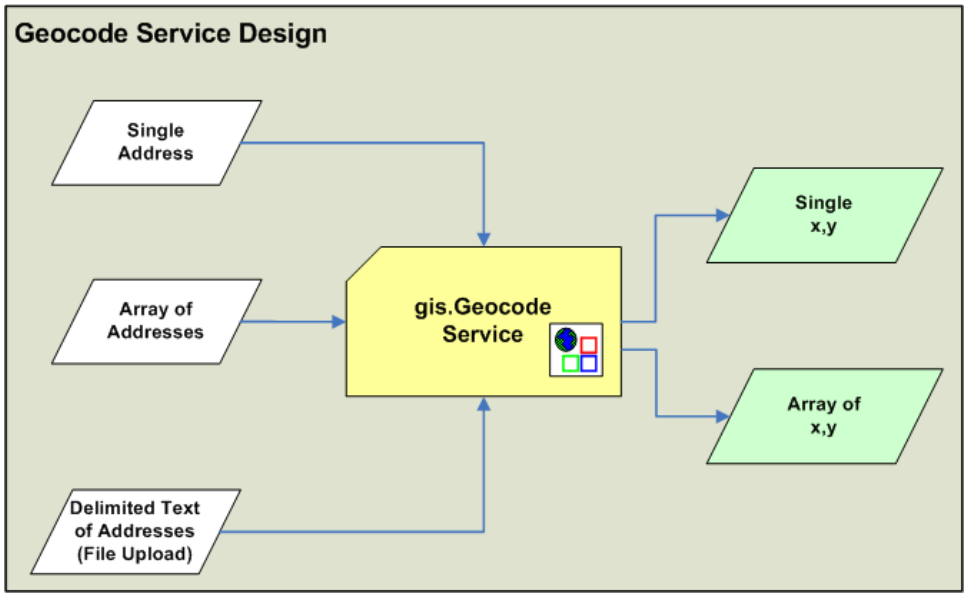
Designing the Geocode Service

In the Job Finder solution, the design team needed to define a simple geocoding service that accepts a well-formed residential address and converts the address into geographic (x and y) location information that can be used by Web and desktop mapping applications.

Adhering to the SOA design principle of service reuse, the development team considered other possible business uses of this type of information among different business units and across the organization. The team initiated the service design process by brainstorming different ways geocoding information is or might be used countywide in current and future solutions. The team quickly realized that it should expand the usability of this service by extending it to perform batch processing on addresses contained in a file or passed to the service as an array of addresses. The service logical design view for the county geocoding Web service is depicted in figure 3.

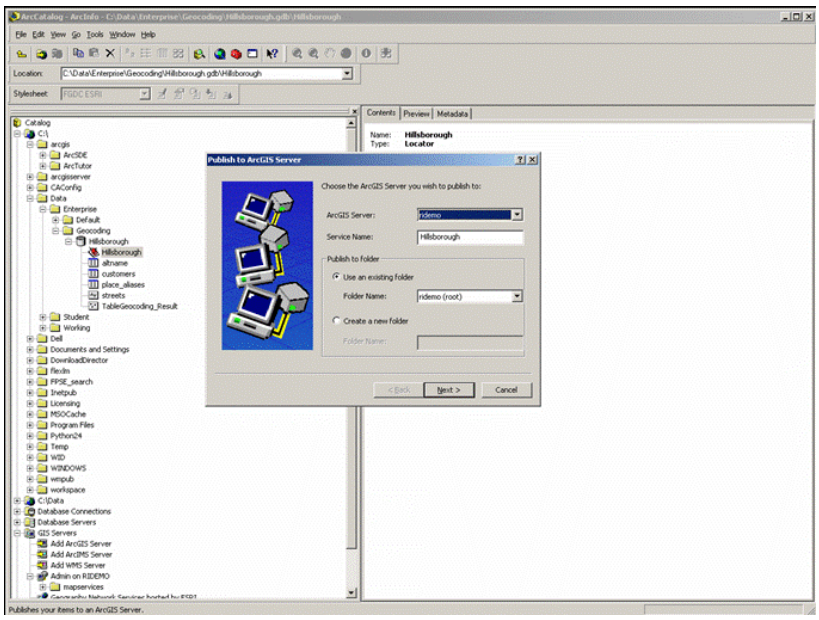
J-9662

Figure 3
Geocode Service Design



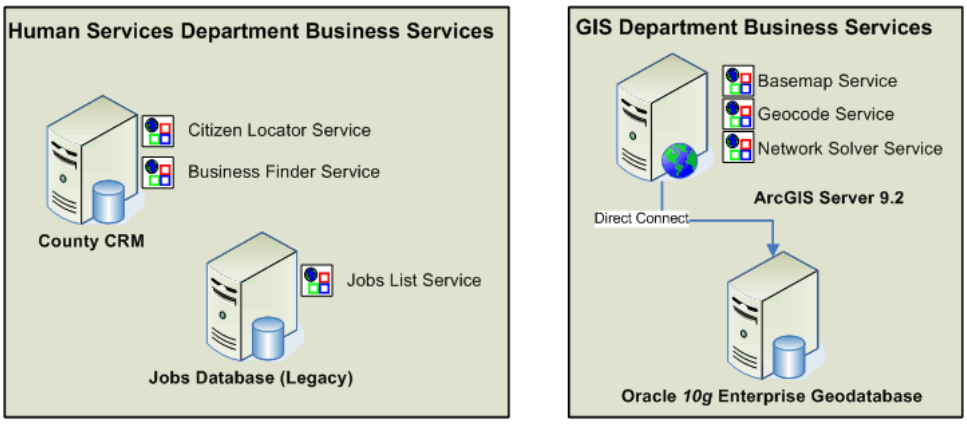
With the geocoding service designed, GIS analysts created a standards-based geocoding service using the geospatial tradecraft tools and functionality of the ArcGIS technology platform. Once the county geocoding service was instantiated according to the services-oriented design, GIS server administrators used the Web-based administration tools in ArcGIS Server to publish the geocoding service to the enterprise architecture as a standards-compliant GIS Web service as shown in figure 4. After being published to the architecture, the GIS Web service is discoverable in the county UDDI online registry and is prepared for use in the Job Finder solution as well as in all other current and future county solutions that need flexible geocoding capabilities.

Figure 4
Publishing Geocode Service to ArcGIS Server



At this point in the project, the design team has selected multiple reusable services from the enterprise service portfolio and developed a new GIS service. Figure 5 illustrates the business and technology services used in the Job Finder solution and how the services align to different business units.

Figure 5
Service Business Alignment



The distribution and alignment of services across business units and the technology architecture ensure that the line of business responsible for the information has the authority and responsibility to manage, monitor, and support its business data and systems. These business owners provide access to the information, data, and products by exposing their underlying systems through a standards-based interface for other systems

to discover, consume, and use. Also of note in this service alignment diagram is the fact that each business unit is using different underlying enterprise technologies to manage and use its data. When using SOA integration patterns and techniques, differences in underlying implementation technologies do not pose an obstacle because the use of SOA interface standards hides the underlying technical details about the service from the requesting service consumer. Abstracting the underlying technology makes it easier to invoke and use the service without knowing or worrying about underlying integration details. The implementation neutral characteristic is another key design objective of SOA that contributes to creating an agile IT architecture. This approach also means that organizations do not have to adhere to a single computing environment across the organization to achieve information sharing and integration.

Design BPEL Composite Services

As depicted in figure 5, the team has identified or constructed all the business services needed to support the Job Finder solution. Each service is independent in function and design, and each is a fully functioning unit of logic without any dependencies on another service or services. According to SOA design principles, a service is a well-defined business function that is encapsulated as a reusable software asset.

Services in an SOA must also be "composable," where one service enlists one or more additional services to complete a workflow consisting of a series of business tasks. In this way, each designed service has two capabilities:

1. The service can enlist other services in order to accomplish a function without foreknowledge of what services it will incorporate.

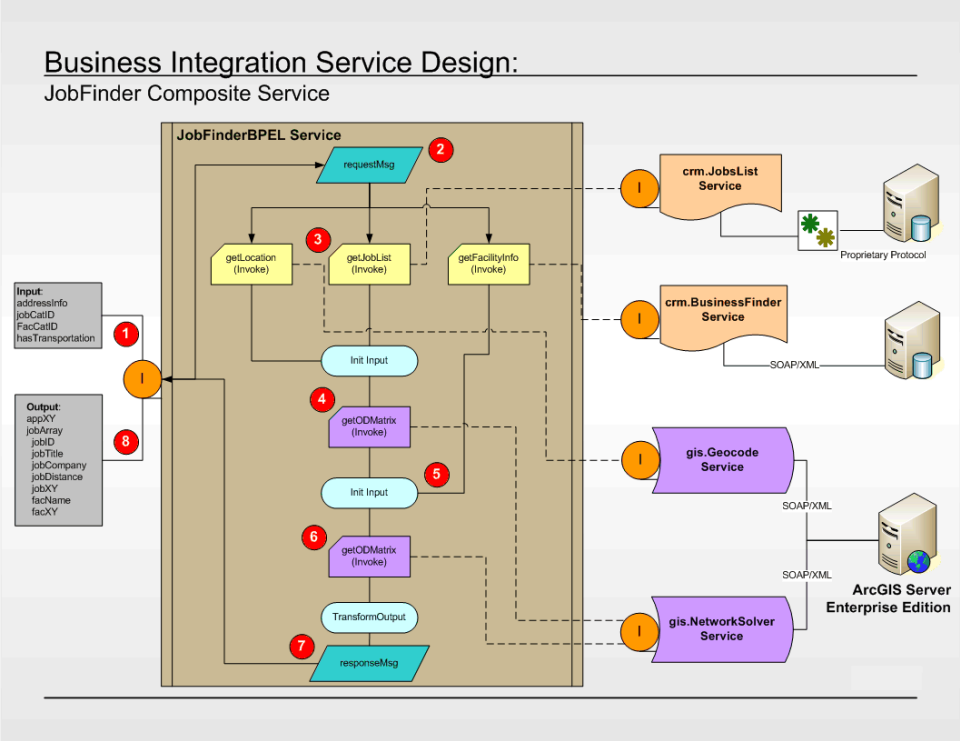
2. The service can be pulled into a service composition by another independent service without foreknowledge of how it will be incorporated.

Service composition enables the automation of business processes and incorporation of business rules in workflows. When services are composed into an ordered set of interconnected functionality, a composite application is created that solves a business need. A composite application is an application element built by combining multiple services within the service-oriented architecture. As services are the underlying building blocks of SOA solutions, service composition is the glue that connects services together in an ordered manner to accomplish business automation. This prescribed interaction, or defined connection of independent services, is referred to as service orchestration. Through the use of service orchestrations, services-oriented solution environments become inherently extensible and adaptive. So integrative and key to SOA, orchestrations themselves are instantiated as XML-based services and abide by the same standards for interoperability as business services do.

The primary industry specification that standardizes orchestration services is BPEL. BPEL is the standards-based platform for orchestrating independent services for business collaboration and business automation. BPEL is an XML-based language for the formal specification of business processes and business interaction protocols. In service orientation, BPEL is used to order and choreograph the interaction of reusable services similar to the way that a conductor carefully choreographs musicians in an orchestra to interact with one another via music. BPEL services are also referred to as business integration services.

The design of BPEL services is accomplished by enterprise SOA architects. Figure 6 depicts the design of the BPEL module, or business integration service, used in the Job Finder SOA solution. The following summary steps through the flow of the BPEL service to illustrate how this module integrates multiple services from disparate systems and technologies into an orchestrated and well-defined automated business workflow.

Figure 6
Job Finder BPEL Service



Below is a summary of this process flow:

1. The Job Finder Web application invokes the Job Finder Composite Application service by passing an XML message according to the operations described in the WSDL service interface. The WSDL service interface defines the methods, parameters, data types, and binding information for the BPEL service.

2. The BPEL service receives the client request message and initializes variables used in the choreographed workflow.

3. Three Web services are invoked in parallel to accomplish three distinct business services: get a list of day care facilities; get a geocoded value for the job seeker's address; and get a list of jobs currently open from the legacy jobs database.

4. The output from the geocoding service and the jobs list is sent to the NetworkSolver GIS service. This service acts on the location information contained in both the service results and calculated distance, travel time, and

routes to potential job locations. The output of the first use of the NetworkSolver service becomes one of the inputs to the next operation in the business process.

5. The list of day care facilities is passed to the NetworkSolver GIS service. This time, this GIS service determines which jobs have a day care facility located within two miles of the job site in accordance with the business rule. If the client request message indicates that day care is not needed (Boolean false), then this second pass through the NetworkSolver service is not executed.

6. The NetworkSolver GIS service identifies the returned jobs that satisfy the business rule of having a licensed day care facility located within two miles of the work site. The returned jobs are assembled and prepared to be included in the response message.

7. The response message is a well-formed XML message that is defined as the output in the BPEL WSDL service interface. The response message is the output of the business process and is defined in the WSDL service interface of the BPEL service.

8. In the final step of the workflow, the requesting Web client receives the well-formed SOAP/XML response message to its service invocation sent in step 1 of the workflow. The Web client is responsible for receiving the response message, parsing it, then rendering the results in the presentation Web client application.

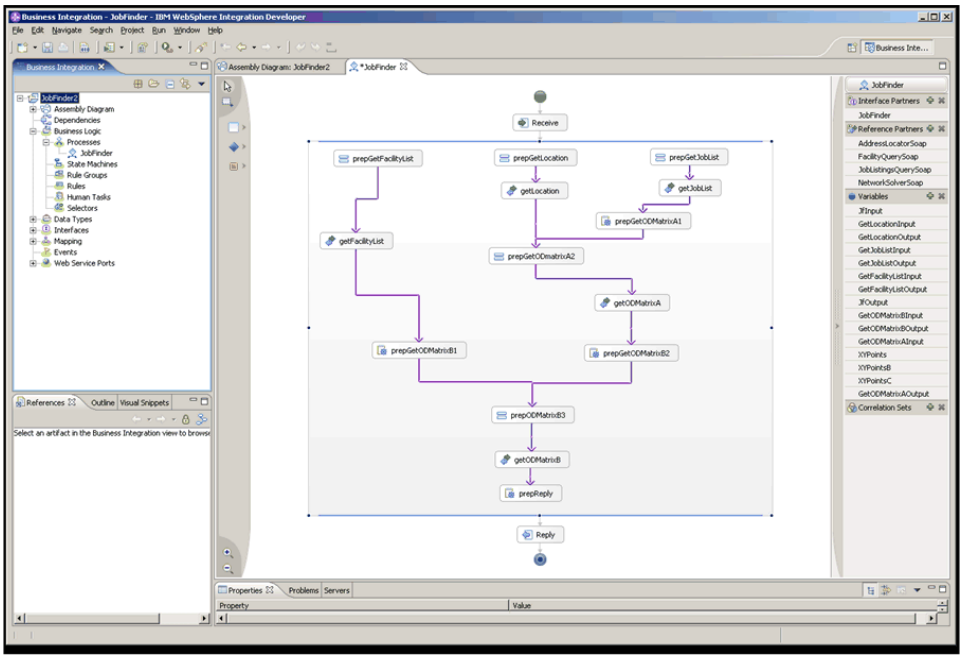
Once the BPEL service design is complete and validated against the desired workflow and information requirements, an integration developer is tasked to create the BPEL service in the integration environment. In this case study, WebSphere Process Server is the enterprise software being used as the integration platform. The BPEL service and other services in the Job Finder solution are enabled and exposed to the enterprise via the integration platform. This next section discusses a high-level view of the development process for the Job Finder BPEL service.

Develop BPEL Composite Services

The integration development team created the Job Finder BPEL service using the WebSphere integration development (WID) environment, a plug-in development module to the Eclipse open source IDE. WID is used to code, test, and validate the functionality of the BPEL service. As depicted in figure 7, WID is a visual and intuitive development environment for constructing the business integration choreography service.

Within this BPEL module, multiple disparate systems and data sources are sharing and exchanging information through standard messages. These standards-based messages are defined and governed by SOA interoperability standards. These architectural messaging standards are the key advantage of an SOA approach as an integrated technology architecture that is focused on meeting the current business needs while remaining flexible enough to change as business needs evolve and new business opportunities arise. The visual programming interface provided in WID assists the integration developer in quickly creating the business integration service. This intuitive development environment is also beneficial when the business process needs to be modified based on evolving business needs and changes.

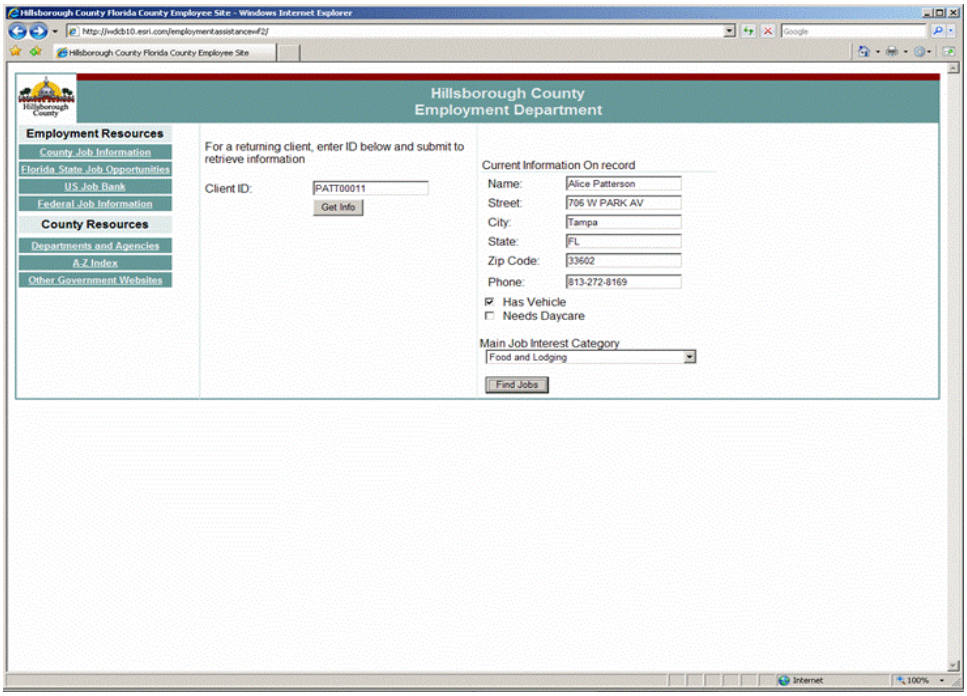
Figure 7
WebSphere BPEL Development Environment



Develop Job Finder Web Client

The last component for design and development is the Web client interface that county workers will use to interact with the Job Finder solution. The ArcGIS Web ADF .NET for the Visual Studio 2005 development environment is used to create the Web application. A view of the Web client application main page is provided as figure 8. The county worker populates the contents of this form by calling the CitizenLocator service to retrieve address information for the resident. The worker completes the form by selecting the job category of interest, indicating whether the job seeker is using public transportation and if the job seeker requires day care. Each of these selections is completed using check boxes or a dynamically populated combination drop-down selector. The county worker submits the completed form to the network as a SOAP/XML message. The input message invokes the Job Finder business process by calling the BPEL service on the enterprise (WebSphere) integration platform.

Figure 8
Job Finder Web Client Search Form

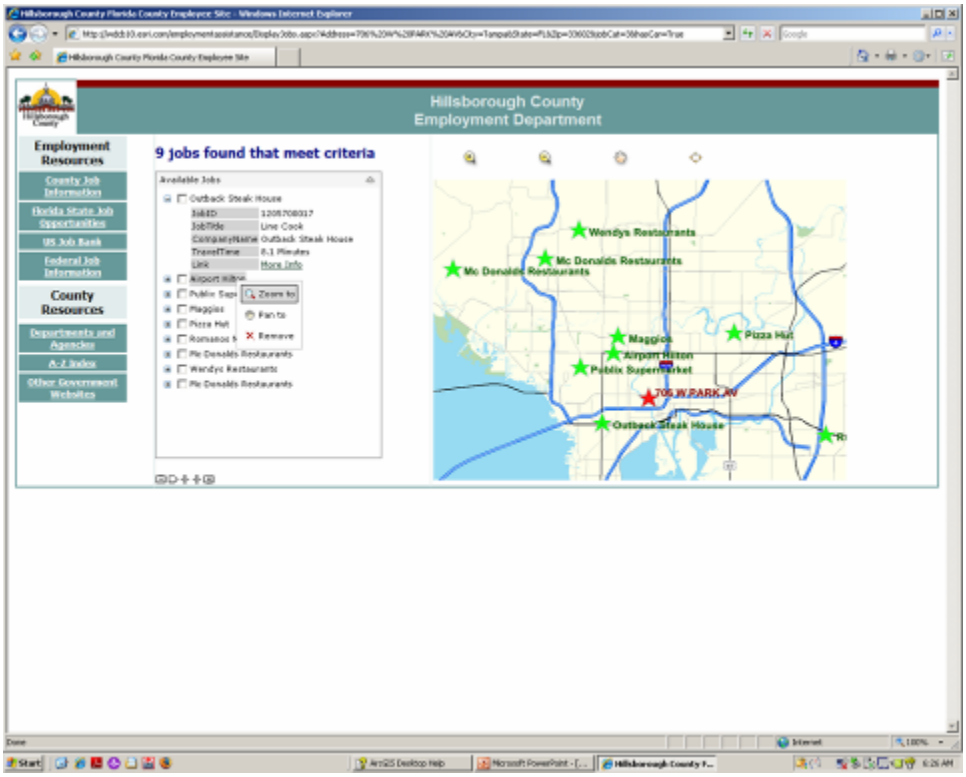


After the main page form is submitted, the Job Finder BPEL process runs through the orchestrated business process to create a list of potential jobs for the requesting candidate. Once the filtered list of jobs is created and assembled via the integration middleware, WebSphere passes the output of the BPEL business process back to the Web client as a SOAP/XML response message. Since this is a short-duration request that runs very quickly, the client is configured to make a synchronous call to the BPEL business service. When the business service completes execution, its results are immediately returned to the requesting client for rendering.

Map and geodata services from ArcGIS Server are responding to the GIS service requests contained in the BPEL business service. The CRM systems are responding to the requests as well. Upon receipt of the response message from the integrated business process and from the map and geodata services, the Web ADF client parses the SOAP/XML messages and dynamically populates the data and map controls to create an easy-to-use and intuitive Web page display of the text and map-based information that is now represented as an integrated solution. An example of the Web client results page is shown in figure 9. The client contains a map control consisting of the job seeker's residence, location of the jobs, and the detailed travel routes along public or private transportation networks. The client also contains a data grid that makes the textual information about the job, its location, travel time distance, and business name readily available to the job seeker. Job results are sorted based on travel time distances calculated by the GIS server.

J-9662

Figure 9
Geoenabled Job Finder Search Results



Conclusion

This case study has walked through a business modernization project that allowed an outdated business process to be significantly improved through the use of Web services, geospatial information systems and data, integration middleware, and an SOA design approach to enterprise architecture. Two business units with disparate underlying technologies and different workflows were integrated through a services-oriented design approach to synchronize systems, data, processes, and people. The success of the Job Finder services-oriented solution created new levels of cooperation and interest across the organization. Business leaders realize there are many more application scenarios that can benefit from services orientation and are providing creative project ideas to the county CIO for consideration.

Appendix F: References

Listed are references used to produce this paper.

- Erl, T. 2005. *Service-Oriented Architecture—Concepts, Technology, and Design.* Crawfordsville, IN: Prentice Hall.
- ESRI. 2007. *Transforming Enterprise GIS with ESRI and IBM.* Redlands, CA:ESRI.
- Krafzig, D., K. Banke, and D. Slama. 2005. *Enterprise SOA: Service-Oriented Architecture Best Practices*. Crawfordsville, IN: Prentice Hall.
- Manes, A. T. 2003. *Web Services—A Manager's Guide*. Pearson Education, Inc.
- Newcomer, E., and G. Lomow. 2005. *Understanding SOA with Web Services.* Addison-Wesley.
- OASIS. 2006 (August). Reference Model for Service-Oriented Architecture 1.0, Committee Specification 1.
- Rosenberg, D. 2005. *Agile Development with Iconix Process: People, Process, and Pragmatism*. Berkeley, CA: Apress.

ESRI
380 New York Street
Redlands, California
92373-8100 USA

Phone: 909-793-2853
Fax: 909-793-5953
E-mail: info@esri.com

For more than 35 years, ESRI has been helping people make better decisions through management and analysis of geographic information. A full-service GIS company, ESRI offers a framework for implementing GIS technology and business logic in any organization from personal GIS on the desktop to enterprise-wide GIS servers (including the Web) and mobile devices. ESRI GIS solutions are flexible and can be customized to meet the needs of our users.

For More Information

1-800-GIS-XPRT (1-800-447-9778)

www.esri.com

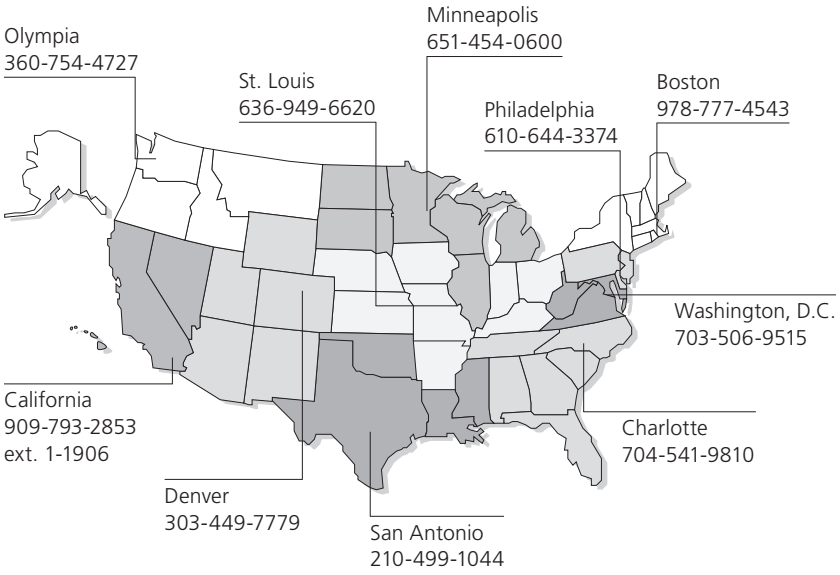
Locate an ESRI value-added reseller near you at

www.esri.com/resellers

Outside the United States, contact your local ESRI distributor. For the number of your distributor, call ESRI at 909-793-2853, ext. 1-1235, or visit our Web site at

www.esri.com/distributors

ESRI Regional Offices



ESRI International Offices

Australia
www.esriaustralia.com.au

Belgium/Luxembourg
www.esribelux.com

Bulgaria
www.esribulgaria.com

Canada
www.esricanada.com

Chile
www.esri-chile.com

China (Beijing)
www.esrichina-bj.cn

China (Hong Kong)
www.esrichina-hk.com

Finland
www.esri-finland.com

France
www.esrifrance.fr

Germany/Switzerland
www.esri-germany.de
www.esri-suisse.ch

Hungary
www.esrihu.hu

India
www.esriindia.com

Indonesia
www.esrisa.com.my

Italy
www.esriitalia.it

Japan
www.esrij.com

Korea
www.esrikr.co.kr

Malaysia
www.esrisa.com.my

Netherlands
www.esrinl.com

Northeast Africa
202-516-7485

Poland
www.esripolska.com.pl

Portugal
www.esri-portugal.pt

Romania
www.esriro.ro

Singapore
www.esrisa.com

Spain
www.esri-es.com

Sweden
www.esri-sweden.com

Thailand
www.esrith.com

United Kingdom
www.esriuk.com

Venezuela
www.esriven.com

GSA Federal Supply Schedule
No. GS-35F-5086H

Printed in USA