

## BIOGRAPHICAL INFORMATION

Michael B. Hamsa  
Vice President, Technology  
GeoSpatial Innovations, Inc.

### Specific Responsibilities

Michael Hamsa joined GeoSpatial Innovations, Inc. as an owner in January of 2002.

Michael Hamsa is Vice President of Technology at GeoSpatial Innovations, Inc. His responsibilities include the management and development of GSI Pocket Designer and GSI Pocket Collector, GPS integrated Windows CE based software for the energy utility market.

### Past Experience

Inventor, US Patent 6,564,201, May 13<sup>th</sup> 2003 – Expert Designer System Virtual Plug-In Interface.

Worked at Cook-Hurlbert in Austin, Texas from November 1992 to April of 2001.

System Architect of CH Expert Designer version 3.0, a utility design tool integrated with the GIS environment. Responsible for software architecture and design decisions for all aspects of ongoing development.

Technical manager of the CH Expert Designer version 2.0 product team, a utility design tool integrated into the Smallworld GIS environment developed with Smallworld Magik. Responsible for technology decisions and software design principles.

Developer on the Enghouse GeoNet GIS implementation project at San Diego Gas and Electric, San Diego, California. Development experience involved Sybase RDBMS in the IBM AIX platform.

### Educational Information

Computer Aided Drafting, Remington College, Lafayette, La.  
Mechanical Engineering - CAD/CAM Option, University of Louisiana, Lafayette, La.

## **Publishing GIS Data through Location-Based Services**

Michael B. Hamsa  
Vice President, Technology  
GeoSpatial Innovations, Inc.  
12593 Research Blvd., Suite 303  
Austin, Texas 78759

### **ABSTRACT**

Providing location specific data to the everyday consumer is a technology that is already becoming a commodity. A practical example of this is Microsoft's MSN Direct service through which a business traveler can get the most current information on news, sports and weather specific to the city he or she is currently visiting. The information is delivered to a wristwatch without touching any buttons or changing any settings and is based on the person's location. The concepts presented in this example can also be applied to electric, gas and telecommunication companies. These companies have data in GIS systems which need to be delivered to workers in the field. Thus, location becomes a very important component in data delivery and can be used to provide concise information. For instance, a field crew scouting outages in a storm situation can have jobs dispatched directly to them as well as be told which jobs are closest to their current position. Another example might be providing a field technician with information about specific isolation valves within a 1 mile radius. This presentation will explore techniques used to distribute information stored in geographical information systems using technologies such as Location-Based Services, Web Services and Smart Clients.

### **INTRODUCTION**

Location-Based Services utilize the location of a mobile device to control and subset the data distributed over a wireless connection. Based on current market research, Location-Based Services will be a \$4 billion market in 2004 in the US and a \$30 billion market worldwide<sup>1</sup>. The findings of this research are a direct result of advancements in technology driven by increased consumer demand. Although the market has not exploded as once predicted, it continues to gain momentum. One of the first federally mandated implementations of Location-Based Services is the FCC's E911 specification in which all mobile phones must be able to provide a caller's location to 911 operators. This should improve personal safety and emergency response since an increasing number of mobile phones are in use.

With the evolution of technology and the increase of consumption of information comes the need for location aware applications. Users are demanding smaller form factor devices in the field while the amount of raw information in the back office continues to increase. These catalysts are driving the development of Location-Based Services. Location-Based Services are applications that use the location of the device to subset data. In the energy company (electric, gas, telecommunications, etc.) most asset related data has a spatial component that is stored in a Geospatial Information System (GIS). This asset data is used throughout the organization, including many users in the field. In certain situations the most up to date asset information is required, i.e. outage scenarios in which a scout is determining the status of an electric network. Since the amount of raw information is too great to simply send to the user via a conventional wireless or cellular network, location aware applications allow a subset of the asset data to be

---

<sup>1</sup> Ovum Research and the Strategis Group

requested and sent to the user's mobile device. The user spends less time in the field sifting through data because only the most applicable data is distributed to the device based on its location.

One of the leading limiting factors to remote data delivery is wireless coverage constraints. Although cellular coverage in the United States continues to improve every year, there are still large holes where wireless connections are not available. In many energy companies whose service territory covers rural area, cellular or wireless coverage is not guaranteed everywhere, thus completely wireless clients are not reasonable solutions. Furthermore, some urban setting wireless coverage is not available. Smart Client applications incorporate a "sometimes-connected" architecture in which the application continues to operate normally even if wireless or cellular coverage is not available. In these applications, synchronization with the information in back end servers occurs when a wireless connection is available otherwise data is cached so that transactions which occurred during a time when a wireless connection is not available can be synchronized when the connection is available.

### LOCATION-BASED SERVICES

Location-Based Services exploit a device's physical location (usually latitude and longitude) to provide specific information or navigational routes between two or more points of interest. This framework is especially interesting to the GIS professional since most of the data in this type of system is already spatially aware. Energy companies have long embraced GIS for asset management and facilities management operations because of its unique ability to associate tabular data to spatial data.

There are several examples of commercial Location-Based Services available today. Microsoft's MapPoint Location Server (MLS) uses Location Service Providers to acquire a device's position so that it can provide concise information to that device. Bell Mobility and Sprint are the first North American mobility operators that will provide location services. These mobility operators use the cell tower id to approximate the position of the device. In addition to Location Service Providers, Notification Providers allow content to be sent to mobile devices based upon the device's current position. For instance, a sales person in an unfamiliar territory can get information on opportunities within a specific distance. With a current subscription, it is also possible to upload company specific information that can be processed and delivered to a user based upon that user's location.

ESRI's ArcWeb Services is subscription-based content that includes street, photo imagery, weather, traffic, base map and flood plain data as well as a variety of other data categories. ArcWeb Services also provides a developer's API which adheres to the most recent standards for interoperability. ESRI's ArcWeb Service also allows Points of Interest (POI) to be uploaded, geocoded and stored. Using the Proximity Service, it is possible to return POIs within a specific radius using a device's location.

As a demonstration, Microsoft, along with the USGS, developed TerraServer to showcase the strengths of Microsoft's database platform. Currently TerraServer contains about 3.3 terabytes of USGS aerial imagery and USGS topographic maps which are public domain and freely redistributable. TerraServer also exposes a web service through which these topographic maps

and aerial imagery can be requested. Using this web service and a latitude/longitude, photo imagery for a device's current position can be requested and displayed to a user in the field.

Trimble and Nextel currently offer a package that allows location-based content to be delivered to a GPS enabled cell phone. The content is in the form of pre-planned trips, topographic and street maps, or aerial photography. For around \$10 a month, this subscription service is available to Nextel customers with qualifying headsets.

The enabling technology in Location-Based Services is the ability to acquire the device's current position. There are several methods that can be used for this purpose. However the most common method is the use of a GPS receiver. GPS has no boundaries of availability in use outdoors since it is the result of a network of 24 operational satellites orbiting in 6 orbits above the earth. In addition, the use of real-time differential correction (DGPS) yields a far more accurate position which can be in the range of a meter or less. One limitation of GPS is that it can not be used indoors since it is a "line-of-sight" technology relying on clear path to the satellites. There are a variety of manufactures that offer small devices running Window Mobile 2003 which also have integrated sub-meter GPS receivers. This type of platform provides the perfect solution for field users because of its size and reliability.

Several standards organizations have published interoperability specifications that provide the basis for location-based services development using spatial data. The Open Geospatial Consortium's OpenGIS Location Services (OpenLS) specification defines the framework for a GeoMobility Server (GMS) which provides Geocoding/Reverse Geocoding, Navigation and Mapping services for mobile terminals. The International Organization for Standardization (ISO) Technical Committee 211 is tasked with the specification for standards revolving around electronic geographic data. Its project 19132 currently proposes a set of standards specifically for Location-Based Services. Although not as far along as the OpenLS specification, it also attempts to define a set of interoperability standards that will allow mobile devices to acquire location specific data over a wireless connection.

Armed with these commercial examples and interoperability specifications in mind, it is possible to develop location-based services that publish data contained with an energy utility's GIS for consumption by mobile devices in the field.

## WEB SERVICES

Web Services are another step forward in the industry's path to interoperability and service abstraction whose recent history includes, Dynamic Link Libraries, Component Object Model, Distributed Component Object Model, Transaction Servers and the .NET Common Language Runtime. Each of these steps in evolution has promised to provide a greater level of interoperability and abstraction as well as solve the problems created in the previous step.

Dynamic Link Libraries (DLL) allowed developers to encapsulate discrete sets of functions so that they could be used by more than one application with the need to copy code. The DLL architecture was plagued with problems because dynamic linking required the exact location of the DLL which wasn't always known. Also there was an informal contract between the exporting DLL and the importing client such that the client always relied upon the same functions and the

same function signature to be exported from the DLL. If this contract was ever broken by the DLL the client may no longer work correctly.

Component Object Model (COM) was a huge step forward for interoperability because it removed the requirement of knowing where the DLL existed. This single aspect increased interoperability applications and brought service abstraction to a new level. COM was still built around the informal interface contract between service provider and consumer so that if the provider ever changed its interfaces the client may not work. Rules were put in place to maintain interface compatibility but still there were inherent problems and contract compatibility was never guaranteed. In addition, COM services were limited because the components had to be deployed to each computer and this made application maintenance and upgrade tedious.

Distributed Component Object Model (DCOM) allowed core components to be installed on a single computer and used by other client computers. This decreased the amount of resources needed to maintain and upgrade core business logic while extending the reach of the application. Now, less of the application needed to be deployed to client computers because the core business logic resided on a server while only a thin presentation layer needed to be deployed to the user's computer. Since many distributed components were becoming involved in transactions, the potential for a transaction to fail increased due to business rules. Keeping track of each process in a transaction became difficult and more code required.

Transaction Servers (COM+) incorporated the distributed qualities of DCOM while adding the ability to track transactions. Each component in a transaction was included within a Transaction Context and each component was allowed to vote on whether its part of the transaction succeeded or failed. The Transaction Context could be configured in such a way so that all components had to succeed for the transaction to succeed or that only certain components had to succeed for the transaction to succeed. If the transaction failed for any reason it was rolled back and no changes to the underlying data sources were made. This technology still depended upon the informal contract between service publisher and server consumer and if this interface contract was broken by the service publisher the client may no longer work.

The invention of the Common Language Runtime (CLR) in Microsoft's .NET Framework finally replaced the informal interface contract of the DLL and COM era with a more formal interface contract based on metadata. The CLR replaces language specific runtime environments (i.e., C and C++ Runtime, VB runtime, etc) with a single runtime environment revolving around uniform data types.

Web Services are the most current step forward in distributed interoperability and abstraction because they allow core business functions to be exposed over the internet or intranet. XML Web Services provide the greatest flexibility but there are also serialization techniques that allow object instances to be serialized on one end of the call, sent in binary or XML format over the wire, and de-serialized on the other end into their original structure. Web Services incorporate the use of Simple Object Access Protocol (SOAP), HTTP GET/POST messaging and MIME to provide a completely abstract process for exposing functionality.

Functions of a Web Service are defined for use through Web Services Definition Language (WSDL), a specification which was submitted to the World Wide Web Consortium (W3C) by Abria, IBM and Microsoft. Now at version 1.1, WSDL itself is in XML format so it is consumable by a variety of development platforms and operating systems. WSDL uses types, messages, ports, operations and bindings to define service endpoints. For maximum interoperability its underlying types system is XSD.

### *Example 1 – Sample WSDL Document*

```
<?xml version="1.0" encoding="utf-8"?>
<definitions xmlns:http=http://schemas.xmlsoap.org/wsdl/http/
  xmlns:soap=http://schemas.xmlsoap.org/wsdl/soap/
  xmlns:s=http://www.w3.org/2001/XMLSchema
  xmlns:s0=http://localhost/GSIWebServices
  xmlns:soapenc=http://schemas.xmlsoap.org/soap/encoding/
  xmlns:tm=http://microsoft.com/wsdl/mime/textMatching/
  xmlns:mime=http://schemas.xmlsoap.org/wsdl/mime/
  targetNamespace=http://localhost/GSIWebServices
  xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <s:schema elementFormDefault="qualified"
      targetNamespace="http://localhost/GsiWs">
      <s:element name="CorrectRoverFile">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
              name="bytRoverFileBlob"
              type="s:base64Binary" />
            <s:element minOccurs="0" maxOccurs="1"
              name="strRoverFileName"
              type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="CorrectRoverFileResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
              name="CorrectRoverFileResult"
              type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:schema>
  </types>
  <message name="CorrectRoverFileSoapIn">
    <part name="parameters" element="s0:CorrectRoverFile" />
  </message>
  <message name="CorrectRoverFileSoapOut">
    <part name="parameters" element="s0:CorrectRoverFileResponse" />
  </message>
  <portType name="DifferentialCorrectionSoap">
    <operation name="CorrectRoverFile">
      <input message="s0:CorrectRoverFileSoapIn" />
    </operation>
  </portType>
</definitions>
```

```

        <output message="s0:CorrectRoverFileSoapOut" />
    </operation>
</portType>
<binding name="DifferentialCorrectionSoap"
    type="s0:DifferentialCorrectionSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
        style="document" />
    <operation name="CorrectRoverFile">
        <soap:operation soapAction="http://localhost/GsiWs/CorrectRoverFile"
            style="document" />
        <input>
            <soap:body use="literal" />
        </input>
        <output>
            <soap:body use="literal" />
        </output>
    </operation>
</binding>
<service name="DifferentialCorrection">
    <port name="DifferentialCorrectionSoap"
        binding="s0:DifferentialCorrectionSoap">
        <soap:address location="http://localhost/GsiWs/DiffCorr.asmx" />
    </port>
</service>
</definitions>

```

Data Exchange specifications are also increasing interoperability between applications. Several XML based specifications have been developed for geospatial information, such as Geographic Markup Language (GML), as well as more specific specifications, such as MultiSpeak, which specify how data should be exchanged between systems like GIS, CIS, Staking, and Network Analysis. More of the industry is moving towards data exchange specification and more industry leading companies are participating in the development of these specifications.

Since Web Services provide such a remarkably abstract process for deployment, they are the perfect candidate to distribute data to remote clients using Location-Based Services technology. Using XML Web Services and standard data exchange formats, a variety of clients can be targeted. These targets can be running on virtually any operating system and can be developed with almost any language that has the ability to communicate via HTTP through the internet or intranet. Web Services are exposed to clients using a Universal Resource Locator (URL) very similar to a common website address.

Web Services allow application developers to provide rich business logic through an abstract layer. Using tools like Microsoft's Visual Basic .NET, or Microsoft's Visual C#, the web reference can be added to the project simply by supplying the URL of a Web Service. The tool then creates a set of wrapper classes that encapsulate the functions and types exposed from the Web Service. Once the wrapper classes have been generated, they can be used in code as if the services were exposed from another project or library running on the developer's computer.

Using Microsoft's TerraService Web Service as a publisher and Microsoft Visual Basic.NET and the .NET Compact Framework, it is very easy to develop code that requests imagery in bitmap

format from TerraServer. The following example requests a map tile based upon a GPS longitude and latitude and creates a bitmap out of the data stream.

*Example 2 – Requesting data from TerraServer-USA, Visual Basic .NET and the .NET Compact Framework*

```
Dim ts As com.terraserver_usa.www.TerraService
Dim center As com.terraserver_usa.www.LonLatPt
Dim abb As com.terraserver_usa.www.AreaBoundingBox
Dim mapImage As Bitmap

' Create the LonLatPt object and populate it from the GPS
center = New com.terraserver_usa.www.LonLatPt
center.Lon = GPS.Lon
center.Lat = GPS.Lat

' Create TerraService object and set the URL
ts = New com.terraserver_usa.www.TerraService
ts.Url = "http://65.54.135.103/TerraService2.asmx"

' Get the area for the center point
abb = ts.GetAreaFromPt(center, _
                        1, _
                        com.terraserver_usa.www.Scale.Scale2m, _
                        240, _
                        320)

' Serialize the bitmap from the stream
mapImage = New Bitmap( _
             New System.IO.MemoryStream(abb.Center.TileMeta.Id))
```

What is provided back to the application is a bitmap that can be displayed to the user.

One drawback to the Web Service framework is its Synchronous nature, meaning that the call is made and the applications current thread is blocked (hung) until the web service call either times-out or the call returns. This results in what appears to the user as a hung application. Given the relativity slow nature of wireless connections, this may not be acceptable. One solution now being used is an Asynchronous calling scheme in which progress can be reported back to the user while processing is occurring. Using the .NET Framework, it is fairly painless to implement an Asynchronous Web Service call, including a mechanism for callback progress reporting in which the Web Service can report actual percent complete back to the user. This is a slightly more advanced implementation, but provides the user with a better experience.

## SECURITY AND PRIVACY

Two very important aspects of open computing and data sharing are security and privacy. Securing data from competitors or malicious attacks and making sure that confidential customer

information remains private is very important. It is however becoming critical for the right information to get to the right people.

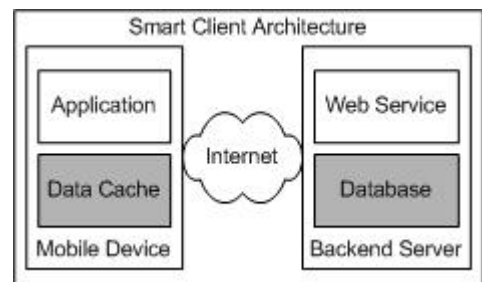
There are several ways an outside source may choose to attack. Denial-of-service attacks are designed to keep valid users from gaining access by launching synchronized calls from many computers. This type of attack over-utilizes the resources on the publishing system by providing more requests that can be processed. This effectively brings the system down. Other types of attacks, like spoofing or replay attacks, are designed to gain access to the data by impersonating a valid user's credentials or intercepting messages and replaying them. Privacy is also becoming more important in Location-Based Services, because the location of the mobile device, and its user, can be tracked.

Using the Web Services architecture, it is possible to use credentials for authorization. Web Service call can be developed so that they require a username and password which are authenticated before the data is distributed. This approach may be more appealing to IT departments because the Web Service acts as a shield between the consumer and the publisher. The consumer must present valid credentials before it is given any information in return.

Privacy will continue to be a very serious personal issue. At this point, is it already possible to track a device's position, in fact many mobile phone companies already offer services that allow a person to log onto a web site, enter a mobile phone number, and get the current position of that phone. The implications of these privacy issues will play a large roll in how successful Location-Based Services will become and how users will adopt their use.

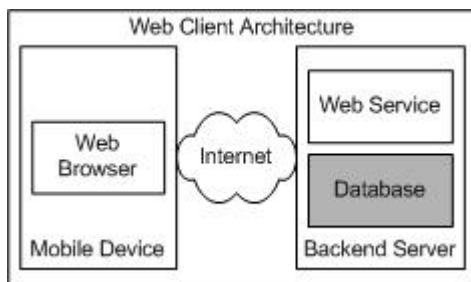
### SMART CLIENTS

The evolution of the Smart Client has been brought about because of the lack of wireless or cellular coverage. Even as we see increased coverage in the United States, there are still large holes in the network where wireless connectivity is not available. In contrast to Wireless Clients, Smart Clients contain all of the business logic, application logic, presentation layers and data required to run themselves. In the Smart Client architecture, the user interacts with the application regardless of the status of the wireless network. Smart Clients



periodically connect to the wireless network (when it is available) and synchronization data stores using technologies like merge replication or transaction replay. The obvious benefit of the Smart Client is that it provides the flexibility of a mobile application while accounting for holes in wireless coverage. The Smart Client architecture is the basis for most mobile application

development today and should remain so until a reliable wireless network with ~100% coverage has been adopted and implemented.



In contrast, the Web Client runs through a web browser on the mobile device or contains only a thin presentation layer (user interface) deployed to the mobile device. The application itself relies upon an internet/intranet

connection to run and can not operate if the connection is dropped. In addition the application's performance is limited by the speed of the connection. The appeal of the Web Client is that virtually none of the application is actually deployed to the device reducing the maintenance required to support the application. Because the weakest link in the Web Client architecture is network availability, it can be an unreliable solution.

### WIRELESS CONNECTIVITY

There are many options for wireless connectivity. At this time most common wireless long-range coverage is provided by cellular phone companies. It has a relatively slow throughput (~70-150kbps) but provides the most extensive network of coverage area. Using either GSM/GPRS or CDMA, the cell based network is used primarily by cellular phone companies to provide voice service. Recently, data throughput rates have been increasing and technologies like the GPRS EDGE network promise to bring even greater data throughput.

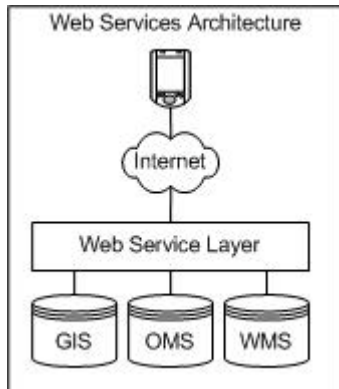
Other evolving technologies such as Ultrawideband (UWB) are proving to provide a much greater throughput (~400mbps) but as of yet do not provide the range necessary to impact wireless clients. UWB was initially developed by the military and uses low-power, short-pulse radio signals in a very short duration over a wide range of frequencies. Initially, it looks like UWB may very well replace Bluetooth in the Personal Area Network arena because of the increased data throughput rates. However UWB may have a hard time replacing the current standard 802.11x networks because of its relatively short range which is less than 10 meters.

Another interesting networking technology is the Meshed Network, which is currently being developed for internal use at Intel. Intel's IMote technology uses several Bluetooth devices, called Motes, placed within range of each other to create a "Meshed Network". Each Mote, about the size of a quarter, can contain a variety of sensors, ranging from vibration sensors to temperature and humidity sensors, that can detect and transmit data back to a centralized 802.11x hub. The data "hops" from IMote to IMote through the mesh until it reaches the network gateway which accesses the mainstream network. If a Mote fails, another Mote within range can fill the hole in the network and the mesh is automatically reconfigured. Motes were developed to monitor the condition of fabrication plants and clean rooms without requiring someone to visit each sensor individually and download the data recorded since the previous reading. Although this technology is proving to be a reliable networking tool, its use for long-range networking is still undetermined.

### PUBLISHING LOCATION SPECIFIC GIS DATA

Combining the Smart Client architecture with Web Services and Location-Base Services provides the greatest flexibility and most reliable application platform. Users have the ability to connect through the wireless network, when it is available, and retrieve data that is relevant to that user's current location. In this scenario, a subset of the entire GIS system is consumed, decreasing the time in which a user is required to be connected. In addition, since only a subset of the data is provided to the user, it is much easier for the user to digest the information. The reach of this architecture can be extended to Work Management Systems (WMS), Outage Management Systems (OMS) and even Enterprise Resource Planning (ERP) systems. When data in each of these systems becomes spatially aware, by association with GIS data, it too can be retrieved based on location.

For instance, an OMS predicts an outage and associates it to a specific device on an electrical network in the GIS. When a field crew scouting outages requests predicted outages for scouting, that crew's current location can be exploited so that a map containing the closest predicted device failure on the system can be provided to the crew, along with driving directions. The crew isn't bombarded with unnecessary information and they are able to make better decisions more quickly. The crew may also be provided with a list of critical customers that are within 2 miles of the current position so each of these customers can be visited and evaluated. Another example might be as simple as facility inspections; when a crew is performing inspections in a specific area, its location can be used to provide facilities within a 2 mile radius which have not been inspected in the last two years.



The facilitating technologies in these scenarios are Location-Based Services, Web Services and Smart Clients. Because Web Services allow data in many different systems to be combined together, they can provide a seamless data source to remote users. As individual systems underneath the Web Service Layer are switched out, the client is shielded because the web service provides an abstract data source and can be altered to target the new system. The Web Service can be developed in such a way that the mobile device's location plays an integral role in how data is subset. By providing users with concise information based upon location, better, more informed decisions can be made in the field in less time.

## BIBLIOGRAPHY

- Box, Don *Essential COM*. Massachusetts: Addison-Wesley, 1998
- Box, Don and Sells, Chris, p. *Essential .NET: The Common Language Runtime*. Massachusetts: Addison-Wesley, 2003
- Kureshy, Arif, p. *Architecting Disconnected Mobile Applications Using a Service Oriented Architecture*. Microsoft, MSDN Online, September 2004
- Martin, James and Odell, James J., p. *Object Oriented Methods, A Foundation*. New Jersey: P T R Prentice Hall, 1995
- McCarthy, Jim, p. *Dynamics of Software Development*. Washington: Microsoft Press, 1995
- McConnell, Steve, p. *Rapid Development: Taming Wild Software Schedules*. Washington: Microsoft Press, 1996
- Rogerson, Dale, p. *Inside COM*. Washington: Microsoft Press, 1997
- Taylor, David A. Ph.D., p. *Object Oriented Technology: A Manager's Guide*. Massachusetts: Addison-Wesley, 1981
- World Wide Web Consortium (W3C), *Web Services Definition Language (WSDL) 1.1*. March 15 2001