Managed Web Services using WS-Manageability

Giner Alor Hernández, José Oscar Olmedo Aguirre

Research and Advanced Studies Center of IPN (CINVESTAV).
Electrical Engineering Department. Computing Section.
Av. Instituto Politécnico Nacional 2508, Col San Pedro Zacatenco. 07360 México, D. F.
e-mail: gineralor@computacion.cs.cinvestav.mx, oolmedo@delta.cs.cinvestav.mx

Abstract. As Web services become pervasive and critical to business operations, the task of managing Web services and implementations of the Web services architecture is imperative to the success of business operations. Web services manageability is defined as a set of capabilities for discovering the existence, availability, health, performance, and usage, as well as the control and configuration of a Web service within the Web services architecture. In this work, we have developed an initial prototype of the WS-Manageability specification, which illustrates how manageability functionality can be seamlessly added to a collection of Web services in a totally non-invasive manner to the services being managed. A basic manager is also included, which allows pluggable manageability aspects of the managed web services to be configured.

1 Introduction

The emergence of Service-Oriented Architectures (SOA) [1] and industry commitment to Web services and Grid computing [2] requires new management capabilities in order to truly achieve the visions and goals of these initiatives. Because, both Web services and Grid environments, leverage the loosely coupled nature of services, deployed across heterogeneous platforms and ownership domains and can dynamically discover and interact with each other during solution design or execution, the manageability of such environments is more critical and challenging to the business they support. Web s :rvices can also cross administrative domains leading to multiple aspects of control requiring a greater emphasis on declarative and real-time management through the use of management policies and service level agreements. Having this into account, we have developed an initial prototype of the WS-Manageability specification, which is able to manage resources (as Web services) remotely from a central location to make management practical. In our prototype the underlying technology used to manage the resources is JMX (Java Management eXtension) [3], but conceptually could be extended to support other management technologies such as CIM (Common Information Model) [4] and SNMP (Simple Network Management Protocol) [5].

The rest of this paper is structured as follows. In the next section we provide the main concepts of WS-Manageability specification. In the following sections we present the architecture of our prototype developed and describe the functionality and its main components. Then we describe the future directions and the work to do. Finally, we emphasize the contributions of our work.

2 Web Services Manageability

The WS-Manageability specification states that Web services management concerns are to be partitioned into "Topics". A topic covers a functional capability that supports management of a particular problem or management domain. For example, the state management for a manageable resource is a management topic. Defining manageability within topics enables incremental and modular development and support of manageability capabilities. The functional capability of a topic can be described, or modeled using a combination of three aspects: properties, operations, and events. Next, we describe the five topics from WS-Manageability specification, which are:

- **Identification:** provides the functional capability to uniquely identify the resource being managed. This static information may include properties that are not required for unique identification, including descriptive and semantic information.
- **State:** provides the functional capability to manage the actual operational state of a resource (i.e. up and down states in a lifecycle) [6]. The State topic defines properties about the operational state, operations to influence a change in state and events indicating when a state change has occurred.
- **Configuration:** provides the functional capability to manage the collection of properties whose values may influence the behavior of a resource. These properties may be changed by the resource or may be changed by the manager which may cause behavior changes in the resource [6]. Operations to access and change the configuration along with events indicating configuration changes are also part of the configuration topic.
- **Metrics:** provides the functional capability to manage metrics for a resource. The metric functional capability includes reset operations for metrics, and metric collection controls. Metrics are raw atomic, unambiguous, quantifiable information. The value of the metric captures the information at a point in time. Generally, these values are numeric, but may be strings as well. Metrics can be contrasted with derived metrics that are calculated using a formula and metrics [6]. For example, average response time during the last hour of execution is an example of a derived metric.
- **Relationships:** provides the capability to query associations that the resource (e.g., endpoint) participates in with other resources. Relationships cover the associations that may exist between a resource and other resources of the same or disparate type. Relationships are important to management for problem isolation, root cause analysis, and impact analysis where resources are related in some way. Relationship types that are defined between resources can have common or very specific semantics, requirements, and implications for management [6]. For this reason it will be common to have topics that include operations that will create relationships that are viewed through this topic.

Within each of these topics are the details which make up the specification. These details are divided for each topic into the following three "aspects".

- **Properties:** provide a way to advertise or surface state information about a resource (e.g., endpoint). They are expressed as named elements in an XML infoset and may be either of a simple type or a complex type. Properties are manipulated directly through operations defined as part of the management interface (e.g. get/set operations) or by events that occur outside of the influence of the manager (e.g. a configuration file being updated). Property change events may be emitted when a property's value is changed. Which property changes cause events to be emitted must be described. Emission of property change events may also be controllable by a manager to permit pausing and resuming the emission of property change events.
- **Operations:** are methods to control the resource or to retrieve state information. Operations may cause temporary or permanent changes in the Web service's behavior.
- **Events:** describe the indications of changes with respect to the resource. An event description is a set of information that describes the change. The information structure of an event description is defined by a named complex type. Notifications are messages containing the event descriptions that are transported to an interested party.

In the next section, we describe the architecture and its components of a prototype that we developed, which covers the main concepts from WS-Manageability specification such as identification, state, configuration, metrics, properties and operations.

3 Web Services Manageability Prototype

We have developed a prototype, which clearly illustrates the spirit and intent of the WS-Manageability specification. The underlying technology used to the implementation is JMX (Java Management eXtension). The JMX architecture consists of three levels: instrumentation, agent, and distributed services. JMX provides interfaces and services adequate to monitoring and management systems requirements [7]. This functionality involves abstracting resources by using components called MBeans (Managed Beans) and remote instrumented resources accessibility through JMX connectors. An MBean is a Java object that represents a manageable resource, such as an application, a service, a component, or a device [7].

The main component of our architecture is a JMX Bridge, which acts as a bridge between the world of resources managed by JMX and web services. In our prototype, the Web services interfaces to JMX are available. Rather than provide a JMX specific web service interface, our prototype provides a web service interface to a manageable resource. Under this approach, the resources can be implemented on different technologies because only is necessary to define a web service interface for a resource. For doing this, we use MBeans to represent a resource being managed. The JMX

Bridge is given information which identifies or describes the MBean instance which represents a specific managed resource. The JMX bridge generates a WSDL document describing the web service, a Java class which acts as the web service implementation, and an Axis deployment descriptor. The resulting Java class must then be compiled, made available to an application server and deployed to Axis as a web service. The generated WSDL can be used to make dynamic calls, validate static calls, or generate a client proxy class. Fig. 1 shows the general architecture of our prototype. In Fig. 1, a generic management handler has been implemented and added to application service's handler chains to gathers statistics.

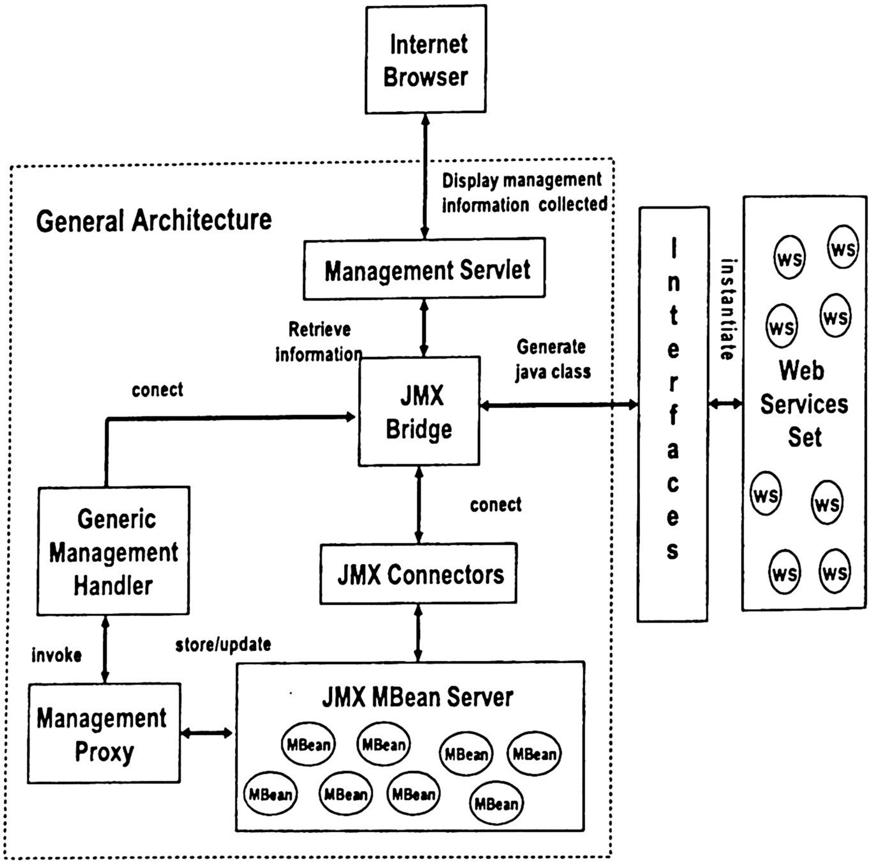
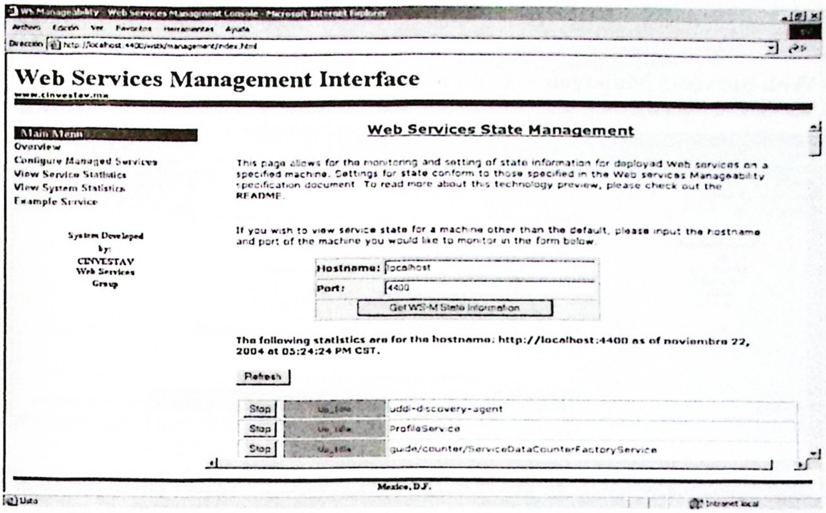


Fig. 1 General Architecture of prototype based on WS-Manageability.

The handler invokes a management proxy which stores, or updates the statistics in MBeans which are instantiated and managed by the JMX compliant MBean server.

Fig. 2 shows a screenshot where the JMX compliant MBean server deploys the stored resources as web services. Also, a management servlet was developed which provides a view of management information collected. The MBean Server and its supporting classes are instantiated by the application server's class loader, which allows this single MBean Server instance to gather MBeans for all management instrumented web services.


WS-Manageability - Web Services Managment Console - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Dirección http://localhost:4400/wstk/management/index.html
Main Menu
Overview
Configure Managed Services
View Service Statistics
View System Statistics
Example Service
Get WS-M State Information
Refresh
Stop
Listo
Intranet local

Web Services Management Interface

www.cinvestav.mx

Web Services State Management

This page allows for the monitoring and setting of state information for deployed Web services on a specified machine. Settings for state conform to those specified in the Web services Manageability specification document. To read more about this technology preview, please check out the README.

If you wish to view service state for a machine other than the default, please input the hostname and port of the machine you would like to monitor in the form below.

Hostname: localhost
Port: 4400

The following statistics are for the hostname: http://localhost:4400 as of noviembre 22, 2004 at 05:24:24 PM CST.

Stop	Up_Idle	uddi-discovery-agent
Stop	Up_Idle	ProfileService
Stop	Up_Idle	guide/counter/ServiceDataCounterFactoryService

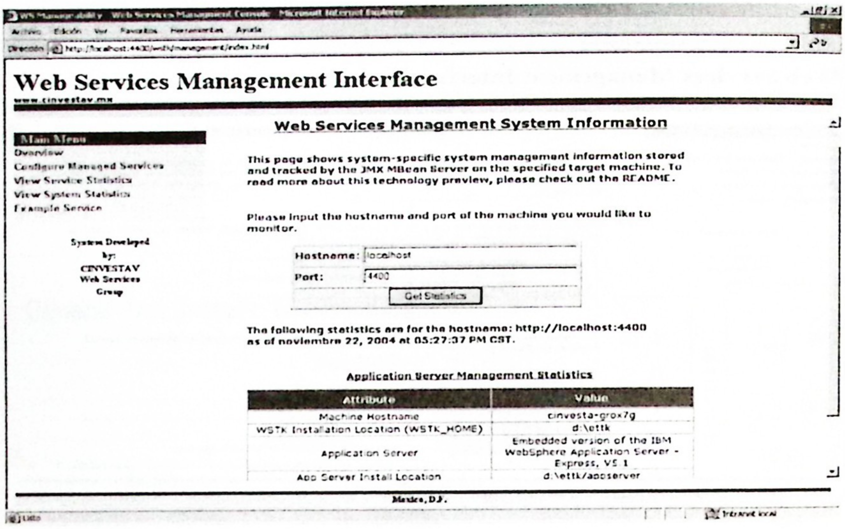
System Developed by: CINVESTAV Web Services Group

Mexico, D.F.

Fig. 2 The JMX compliant MBean Server deploying registered web services.

The most significant of this work is a JMX-based management web service which allows to store MBeans within a JMX MBean Server which can be accessed and manipulated from an AXIS-enabled remote console. Under this approach is possible to monitor the status of JMX-enabled services and components through an AXIS (SOAP) client interface. The MBean server tracks global information and statistics about web services. For doing this, a web-based interface was developed, which displays this data based on a hostname and port pair specified in a simple form embedded in a servlet. Collectively, this functionality represents the direction towards active management of JMX-enabled data and application resources over a standard AXIS (SOAP) interface. Furthermore, MBean tracks global information regarding the configuration and use of the server for which information is being requested. This global information can be tracked and retrieved by a remote console for the purpose of monitoring and remotely managing application servers and web services in an enterprise environment. Fig. 3 shows a screenshot of the kind of information that JMX compliant MBean Server deploys. Among the kind of information collected in the JMX compliant MBean Server is included the following:

- The brand and name of the application server being used
- The version of the server being used
- The hostname of the server
- The HTTP port number currently in use by the server
- Total number of managed SOAP services deployed
- Total number of RPC calls to all managed services combined


WS Manageability - Web Services Management Console - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Dirección http://localhost:4400/wsth/management/index.html
Main Menu
Overview
Configure Managed Services
View Service Statistics
View System Statistics
Example Service
Get Statistics
Listo
Intranet local

Web Services Management Interface

www.cinvestav.mx

Web Services Management System Information

This page shows system-specific system management information stored and tracked by the JMX MBean Server on the specified target machine. To read more about this technology preview, please check out the README.

Please input the hostname and port of the machine you would like to monitor.

Hostname: localhost

Port: 4400

System Developed by: CINVESTAV Web Services Group

The following statistics are for the hostname: http://localhost:4400 as of noviembre 22, 2004 at 05:27:37 PM CST.

Application Server Management Statistics

Attribute	Value
Machine Hostname	cinvesta-grox7g
WSTK Installation Location (WSTK_HOME)	d:\ettk
Application Server	Embedded version of the IBM WebSphere Application Server - Express, V5.1
App Server Install Location	d:\ettk/appserver

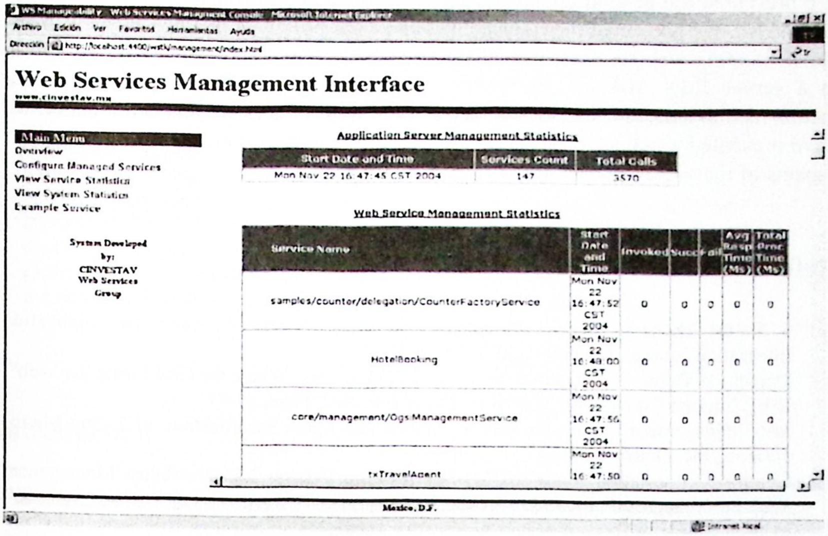
Mexico, D.F.

Fig. 3 The JMX compliant MBean Server deploying global information.

This illustrates an approach to managing web services by instrumenting AXIS through the use of handlers to provide a JMX-based systems management interface. In this case, AXIS handlers were developed and added to the handler chains of our various web services to allow statistics about those services to be gathered. The JMX MBean Server which tracks the service statistics is instantiated globally within the application server's JVM allowing statistics to be tracked across all instrumented web applications. Fig. 4 shows a screenshot of the kind of information that JMX compliant MBean Server deploys when the web services are invoked. Under this approach no code changes to the web services them-selves are required to enable this functionality. The handlers simply need to be added to each web service's handler chain.
Finally, for each instrumented web service invoked, the following four statistics are tracked:

- Current state
- Total number of successful invocations
- Number of failed invocations (request received the server but resulted in an exception)

- Average response/transaction time for successful requests



WS Manageability - Web Services Management Console - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Dirección http://localhost:4400/wstk/management/index.html

Web Services Management Interface

www.cinvestav.mx

Main Menu
Overview
Configure Managed Services
View Service Statistics
View System Statistics
Example Service

System Developed by: CINVESTAV Web Services Group

Application Server Management Statistics

Start Date and Time	Services Count	Total Calls
Mon Nov 22 16:47:45 CST 2004	147	3570

Web Service Management Statistics

Service Name	Start Date and Time	Invoked	Succ	Fail	Avg Resp Time (Ms)	Total Proc Time (Ms)
samples/counter/delegation/CounterFactoryService	Mon Nov 22 16:47:52 CST 2004	0	0	0	0	0
HotelBooking	Mon Nov 22 16:48:00 CST 2004	0	0	0	0	0
core/management/OgsiManagementService	Mon Nov 22 16:47:56 CST 2004	0	0	0	0	0
txTravelAgent	Mon Nov 22 16:47:50	0	0	0	0	0

Mexico, D.F.

Fig. 4 Graphic interface where the MBean server deploys the current state, total number of successful and failed invocations, and the average response/transaction time for each requests of the stored web services.

4 Future Directions

Our prototype shows only the basics of what a JMX-based management system is capable based on WS-Manageability of. However, many items are possible for future development, including:

- Further implementation of the WS-Manageability specification
- Migrating from AXIS Handlers to Web Services Bus Filters.
- Provide facilities and instrumentation inside web services to expose and illustrate their own management capabilities.
- Extend management capabilities in web services to include automatic and manual operations (the ability to change the behavior, or state of a service), and events and alerts (the ability to send notifications when some criteria is met allowing for more proactive management of the service).
- Provide a mechanism to control which services are managed and how much information should be collected both as a install time and runtime configuration setting.

5 Conclusions

In this work we have presented a prototype of the WS-Manageability specification. We presented the general architecture and described the functionality of each component. Also, we presented the implementation of our prototype in a non-invasive manner to existing web services, which is easy to apply across all deployed Web services in a server JMX MBeans. Furthermore, we propose a JMX MBeans server as a mechanism to store the statistics gathered when web services are invoked. Finally, we have presented a manager interface, which can be easily extended as more topics and aspects of the WS-Manageability specification.

References

1. S. Pallos Michael. "Service Oriented Architecture: A primer". Enterprise Application Integration Journal. December 2001. Pages 32-35.
2. Domenico Talia. "The Open Grid Services Architecture: Where the Grid Meets the Web". IEEE Internet Computing. November-December 2002. Pages 67-71.
3. Java Management Extensions Instrumentation and Agent Specification, v1.2. Sun Microsystems, Inc. October 2002.
4. Common Information Model (CIM) Specification Version 2.2. Distributed Management Task Force, Inc. June 14, 1999
5. M. Feit Sidnie. *SNMP: A Guide to Network Management*. McGraw Hill Series on Computer Communications. September 1, 1994. ISBN: 0070203598.
6. Potts Mark, Sedukhin Igor, Kreger Heather. "The Web Services Manageability Specification 1.0". IBM, Talking Blocks, Computer Associates. September 10, 2003
7. Sun Microsystems, Java(TM) Management Extensions (JMX TM) Reference Implementation v1.2. http://java.sun.com/products/JavaManagement/download.html