Service Oriented Architectures (SOA) in IB-ARM

Background

Web services provide a standards-based method for invoking a functional service located either internal or external to the organization. Architectures utilizing such web services are referred to as service-oriented architectures (SOA). There is undeniable benefit to building systems based on well-defined, self-contained services that deliver effective business functions.

However, for many organizations there are major challenges in migrating existing systems to service-oriented architectures and in maintaining SOA-based application software after it has been developed. IB-ARM is uniquely positioned to help in both of these key application management functions.

Enabling Applications for SOA

The desire to reuse software assets through SOA is a key driver for organizations. There is huge value locked in legacy systems and an enormous benefit to reusing that software via SOA-enabled applications. IB-ARM assists in the migrating legacy software to SOA-enabled applications using the following steps:

Step 1. Understanding Existing Applications

IB-ARM has helped many clients to understand their existing applications. These applications are often older, built in legacy technologies and key staff knowledgeable about the applications have often left the company. BUT these applications still deliver significant business value and could deliver even more value if re-structured to be consumed as a Service. These applications must first be understood in order to be migrated to new SOA-enabled platforms.

For SOA migration, a key step is to identify candidates for service integration.

- online screens / maps (BMS, MFS)
- Existing message interfaces (MQ)
- Program / function API's (copybook)
- Data / file interfaces (copybook)

These candidates can all be readily identified within IB-ARM

Step 2. SOA Enablement / Migration / Modernization

Development projects are then undertaken to migrate to new platforms and technologies based on a solid understanding of the current applications. Key steps for SOA migration are:

Create starting point Business Services

IB-ARM automates the current process of extracting starting point business services from current code based on application / business knowledge and / or documentation. Extraction can be based on any pattern such as:

- By component type – maps, programs, messages, files, database tables
- By naming standards
- By data accesses – file or database access

- By screen access – sends / receives

New Application Architecture

As new Web Services are developed and integrated into the application software, IB-ARM is used to capture and map the Services components and their relationships. This provides a basis for measuring progress of the service enablement effort. Perhaps more importantly, IB-ARM provides a method of documenting and organizing the new services and their usage for ongoing maintenance and development

Measure Coverage and Progress

IB-ARM has assisted clients in this stage by measuring the progress of application development. By doing periodic refreshes during development and comparing against the previous snapshot, management can assess the progress of the development project against plans.

Step 3. Ongoing Maintenance

IB-ARM has helped many clients in maintaining their application portfolios both before and after development efforts. It is particularly important to capture application documentation developed during a migration project. This information is critical in the ongoing maintenance and enhancement of the application. It is especially critical if a significant portion of the development was outsourced; the re-patriation of the application support is much smoother when a solid base of application knowledge is provided within the IB-ARM repository.

Maintaining SOA-based Applications

For many organizations, the development of SOA architectures has resulted in a proliferation of services. The problem then becomes understanding what services are being used by what systems. In a large organization, this can be a major issue in terms of maintaining and managing the user of services. IB-ARM assists an organization by parsing service and understanding their usage.

From a conceptual point of view, a central paradigm in programming involves objects invoking other objects. There are several mechanisms for used for handling such invocations:

- Message Bus
- Message queuing (asynchronous communications)
- Remote procedure calls (RPC)
- Object request broker (ORB)

The keys to extracting the semantic interpretation of each of these invocation types are:

- Understanding the nature of the invocation
- Understanding how to parse the requesting and the responding code
- Understanding how to map the relationship between the requester and responder, usually by defining an escalation rule for the specific technology context.

Parsing Web Services in IB-ARM

The IB-ARM solution for parsing Web Services follows the conceptual approach described above.

Parsing WSDL in the Directory of Web Services

The Web Services definitions, in WSDL format, are published in a Directory of Web Services. This directory can be in UDDI, ebXML or other format. Parsing the Directory will provide a list of the published Web Services. This will provide a component list of web services providers that should be found during parsing. Since WSDL uses XML, this parsing is quite simple for IB-ARM.

Parsing Web Service Providers

Web services providers are typically implemented using programming languages designed for interaction with the web, such as Java Servlets or Application Server Pages (ASPs) that call a back-end program or object. These Web service implementations are also typically represented using a Web services description language (WSDL).

For each web service defined in the Directory, parsing will be conducted as for any other set of components and a component tree will be created based on the technology context (e.g. Java servlet, ASP). The web service definition in the directory will allow the web services to be identified as such and will allow escalation to the web service requester objects.

Parsing Web Service Requesters

Web services requesters are typically implemented using programming languages that issue HTTP requests. The requesters will issue a SOAP request to invoke a web service using the message format defined in the web services definition. The parsers will parse the web components as usual. In addition, they will identify the SOAP request and identify the web service request.

Connection between Web Service Providers and Requesters

An escalation rule is defined to map the web service provider with the requester. This will create an explicit relationship in the IB-ARM repository from the web service requester to the web service provider. Note that this can be a many-to-many relationship: a web service can be requested by many requesters and a requester can request many web services.

External Services Providers

Web Services can be provided by external organizations. In this case, the source code for the web services will not be available. A specific client convention (example: URI name, existence within UDDI directory or some other mechanism) should allow external web services to be identified by the IB-ARM parsers.

About Information Balance, Inc.

Information Balance, Inc. develops software solutions to support large scale application development and maintenance practices across mainstream technology platforms. Its flagship product **Application Road Map (ARM)** is a unique Application Portfolio Management (APM) solution. The company's clients include leading financial services, insurance, telecommunications and retail organizations across North America.

For more information, please call 416-962-5235, e-mail infobal@infobal.com, or visit our website at www.infobal.com.