

SOA Gateway Concepts and Facilities

- Introduction to the SOA Gateway
 - Why a Standards Based Service Oriented Architecture ?
 - The SOA Gateway Approach
 - How does the SOA Gateway work ?
 - Conclusions
-

Introduction to the SOA Gateway

The SOA Gateway will enable an organization to save a large percentage of their integration budgets by using a productized approach to what has, until now, been an expensive, people intensive, repetitive process to get at existing data and business logic. The SOA Gateway enables access to these assets from all of the most popular languages, products and Enterprise Service Buses (ESBs) available today. This capability has been promised a number of times over the years and has not been delivered, however, with the advent of W3C standards such as WSDL and SOAP, and specifications such as the REST architecture, this promise can now be delivered. If you are a little skeptical because you have heard all this before, read on to see how the SOA Gateway achieves this.

The Integration Problem

Why a Standards Based Service Oriented Architecture ?

A Service Oriented Architecture (SOA) can mean different things to different people. Ultimately, in its loosest form, if a program can be called in some way and give a result, it is providing a service and thus the infrastructure could be called 'Service Oriented'. At the end of the day, many organizations have been doing this for a long time by implementing the business logic and screen logic separately so that the business logic may be called from multiple places so a SOA is nothing new apart from the terminology.

The nirvana that most organizations wish to get to is where their products and applications can interface seamlessly with each other on multiple platforms. This is sometimes achieved by what are called 'Web Services'. However, this term is often also applied loosely on the basis that if a service is delivered via the Web it is a 'Web Service'. These are generally proprietary, non standard Web Services which can only be integrated once you know the proprietary mechanism they are used.

What does offer the integration that organizations desire are Web Services that are implemented using W3C standards like WSDL and SOAP or specifications such as the REST architecture. The implementation of interfaces using these standards will enable applications to seamlessly interface with each other without any programming requirement. The SOA Gateway specifically only offers access to data and business logic using recognized standards to enable simpler integration. Any reference to Web Services in the SOA Gateway documentation set will mean standards based Web Services implementing using WSDL and SOAP or using the REST Architecture.

The Standards Used

The Simple Object Access Protocol or SOAP emerged as a mechanism to enable applications to access objects on any platform without any knowledge of the platform. Along with the Web Services Definition Language or WSDL, it forms the basis for the provision of standards based Web Services. It has been accepted as the best protocol available for accessing data and applications on heterogeneous platforms without any knowledge of the platform or the operating system where the accessed object resides. It is emerging as the agreed standard for future application development and integration efforts. In addition, the SOA Gateway enables access to data and business logic using the REpresentational State Transfer or REST architecture which enables access to resources using a simple URL based approach.

Open and Simple Protocol

The protocol is open and thus is not controlled or owned by any vested interest. It has been designed to work with any software or hardware. Its greatest asset, like TCP/IP and HTTP before it, is its simplicity. It is possible to read SOAP requests and responses with the human eye and understand what they are trying to achieve. The SOAP Standard itself is extremely light particularly when compared to other standards for linking programs together such as CORBA, COM, COM+ and so on.

Platform and Operating System Agnostic

The test for any standard that claims to link software together is how easily it can be run on new platforms. In the case of SOAP, it is already in use on all of the leading platforms and operating systems available today. While there is a large amount of code available today that uses or implements SOAP in some way, it is the standard itself and not necessarily a set of code that is important. The SOAP standard can be used to solve each problem in the best possible way instead of relying on lots of infrastructure to enable SOAP. This flexibility enables the standard to be used on any platform or operating system.

No Client Foot Print Required

As all of the leading technologies have an integrated understanding of how to call SOAP Services, there is absolutely no software footprint for the SOAP Service on the client platform. This leads to easier deployment and maintenance of SOAP based clients.

IBM, Microsoft, BEA, SAP and others agree

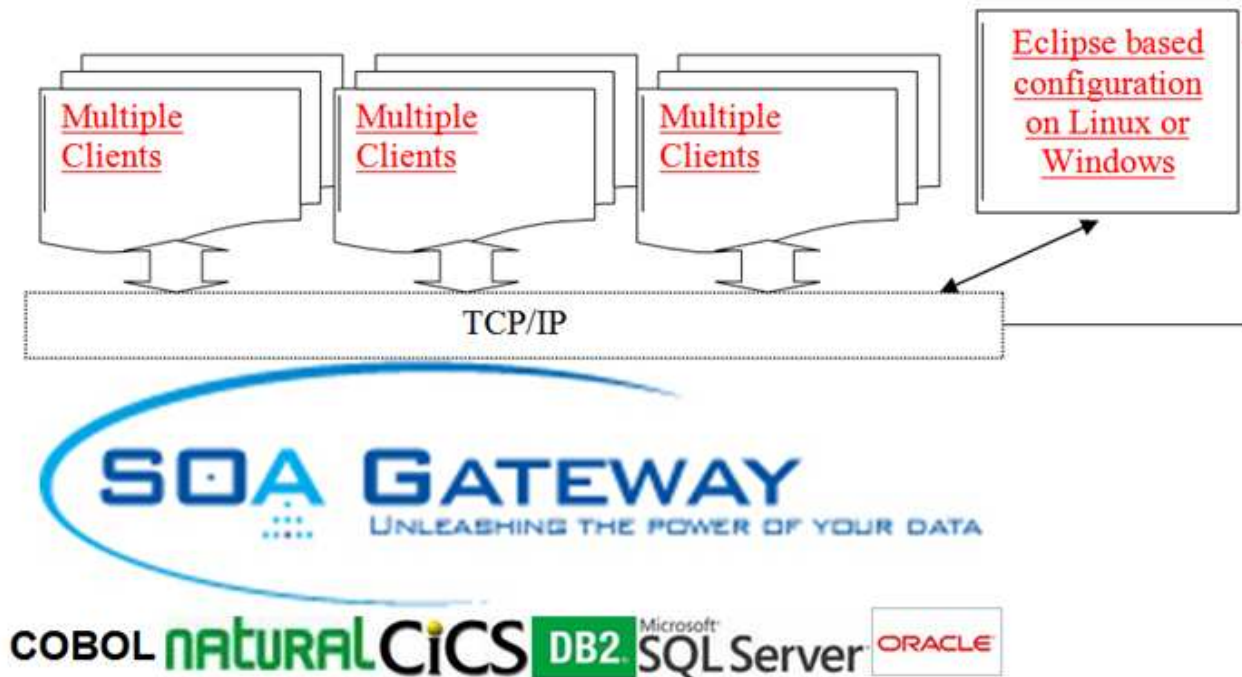
All the leading software manufacturers believe that standards based Web Services are the way of the future. All new and upcoming releases of software will include some support for SOAP. For applications that run on the desktop, this generally means that they can issue SOAP requests. For heavier duty 'server' applications, they will generally have the ability to issue SOAP requests and to service SOAP requests in some form or another.

The SOA Gateway Approach

The SOA Gateway adopts the view that data and business logic can be exposed in a simple, open and logical fashion that requires no coding on the platform where the data and business logic resides. This leads to the point where access to existing data and business logic on any existing platforms can be achieved by installing software and configuring it. This task can be undertaken by the system administrators who are normally available on site. Looking to the future, later projects can then continue to reuse the software with only a configuration requirement to make additional data available for other

application development or integration projects.

Overview



Less Points of Failure Leads to a More Resilient Solution

With this architecture, there are less points of failure. The application talks directly to the software that accesses the database or the program containing the business logic, thus the additional hop from client to middleware and middleware to server is avoided. Less points of failure leads to a more stable and reliable implementation.

Only Configuration - No Programming Effort

Once the SOA Gateway has been installed on a platform, the only additional effort that must be made to make data or business logic available is to configure the product to access the data or business logic. This can be done by the administrator or systems programmer for the existing platform instead of having to get programmers with the appropriate knowledge to develop code on that platform. The SOA Gateway provides a discovery interface for each supported database or programming language to automatically create the Web Services for you. This means that your programmers or users can focus on the business issue in hand instead of spending time on getting to the data or business logic.

Knowledge Only Required in New Programming Logic

No knowledge of different platforms and software is required apart from that which will be used for the new application or integration effort. This will lead to a less complex project as all programmers are working in the same environment and there is no coordination required with programmers on different platforms.

New Technologies Use Web Services

The Web Services technology has simplified the way program to program communication can occur. All new technologies can use Web Services and existing technologies that are being further developed will all ultimately be able to use Web Services. The Web Services standards in use are hardware, operating system and data agnostic and as such can be used from any programming language or paradigm and from any platform.

Uses Less Hardware Resources

There is less software installed on the existing system and less hops for each message backwards and forwards. This leads to a smaller footprint on the system and less usage of valuable memory and CPU resources during execution.

Specialized Service - Only Does What You Need

The SOA Gateway is a specialized service to access your data and thus does exactly what is required. There is no large body of code and functionality that is not used and must be installed as part of the product as can be the case with standard middleware solutions. This saves on time, resources and complexity leading to a more performant solution which is extremely robust.

Totally Reusable Web Services

Once data or business logic has been configured once as a Web Service, it can be reused again and again by further application development or integration projects. No further configuration or programming is required as the data or business logic is available in a standard fashion that all technologies can use.

Support Only Required for New Platform Code and Implementation

Once the system is implemented, continuing support is only required for the new client code on the platform where it has been installed. This saves the cost of having support on hand for the code and middleware that runs on the existing platform for the user.

One Simple API to access Data or Business Logic

Once programmers have learned to manipulate one database or application program with the SOA Gateway Web Services requests, any other Web Service configured in the SOA Gateway can be accessed using the same technique. This saves on training and enables programmers to be more productive with what they know instead of having to continually learn and understand different technologies.

How does the SOA Gateway work ?

The SOA Gateway has multiple different drivers to create Web Services for different databases and different programming languages. Depending on the amount of meta data available for a given database or programming language, the SOA Gateway Resource Discovery interface can automatically define the components required to wrap a database or application program as a Web Service. The following outlines the steps required to create Web Services for different types of resources:

1. In the case of relational databases, the catalog is used to determine the names of each table in the database. The user may then select the table(s) for which Web Services are to be created and the catalog can again be used to determine what columns are in each selected table. In some cases, you may not want each column in a table to be exposed for a specific service in which case it is possible to modify the view of the table associated with the Web Service and delete the unwanted columns. It is possible to have multiple different views of the same table.
2. For Adabas, the database is queried to determine what files are defined. The user may then select from the list of defined files and the Web Service will be built based on the field names for the selected files. Again if all fields are not to be shown for a given ADABAS file, the view may be edited and the unwanted fields deleted.
3. The SOA Gateway control Center can also build views from SYSOBJH offloaded data which, in the case of ADABAS, will result in views with long names.
4. For application programs written in COBOL or Natural, again it is possible to determine what programs are available from a Natural library or dataset containing COBOL programs. The administrator must then connect the programs to be wrapped as a Web Service with the COBOL or Natural copybooks or structures that map the parameters passed to the COBOL or Natural program.
5. For file systems that have no meta data internally such as VSAM, the SOA Gateway can list all VSAM files from the catalog to determine what Web Services to create. However, the administrator must then tell the SOA Gateway where to find a COBOL copybook which describes the file and the view can be created from this.
6. If there is no meta data available for a given database or application program, as a last resort it is possible to build views for a given Web Service with the DataView editor. The database file / table can then be created from the SOA Gateway Dataview.

Refer to the Resource Discovery documentation for more information about the above.

Once the Web Services is defined, it may be invoked using the URL based REST approach or it may be invoked using SOAP. The SOA Gateway can deliver the WSDL directly to the application or the WSDL may be registered with a UDDI server such as CentraSite from Software AG

Conclusions

The SOA Gateway can enable your new generation of Java, .NET and MONO programmers to be instantly productive accessing existing data or invoking existing business logic in other languages. These programmers can get access to existing data and business logic in minutes rather than weeks or months as is the case when the traditional approach is taken. No knowledge of the database or programming language being invoked is required and no footprint is required on the client system. They are simply provided with the URL for the WSDL of the Web Service that they need to access and then have everything needed to work with this data from any Web Service enabled environment.

Recommended reading

The following documentation is provided with the SOA Gateway and gives examples and tutorials about how to use the SOA Gateway once it has been installed:

Basic:

- Introduction to the Eclipse Workbench
- Configuring the SOA Gateway through the SOA Gateway Control Center

Intermediate:

- Web Service Discovery

Advanced:

- Accessing Adabas through Microsoft InfoPath
- Accessing Adabas using C# and the .NET Framework
- Accessing MySQL using C# and the .NET Framework