International Journal of Computer Applications (0975 – 8887)
Volume 121 – No.5, July 2015
pp27-29

Software System Integration – Middleware – An Overview

Seemal Asif
School of Aerospace, Transport and Manufacturing
Cranfield University
Cranfield
Bedford, MK43 0AL, UK

Philip Webb
School of Aerospace, Transport and Manufacturing
Cranfield University
Cranfield
Bedford, MK43 0AL, UK

ABSTRACT
The integration of different softwares written in different language and based on different platforms can be tricky. In that situation a middleware is necessary to enable the communication between different softwares. The middleware enables the software system not only to share data but also share the services. This paper gives an overview of some of middleware technologies which can be used to integrate different software systems.

General Terms
Software Integration, Middleware, SOA (Service Oriented Architecture), Web Services

Keywords
Middleware, SOA, SOAP, Web Services

1. INTRODUCTION
Software system integration is essential where communication between different applications running on different platform is needed. Suppose a system designed for payroll running with Human Resource System. In that case employees' data need to be inserted in both systems. The system integration benefits a lot in these cases where data and services needed to be shared. Web services [1] are becoming very popular to share data between systems over the network and over the internet as well. In software industry the software integration carried same steps as software development and hence demands same kind of development procedures and testing.[2] This ensures the meaningful and clear communication between the systems. Systems integration becomes inevitable in Enterprise Systems where the whole organization needed to share data and services and give the feel to user as one system. [3] The core purpose of integration is to make the systems communicate and also to make the whole system flexible and expandable.

2. SOFTWARE SYSTEM INTEGRATION

2.1 Middleware
Independently written software systems need to be integrated in large system for example industry, institution, etc. These systems need to agree on common method for integration. To get them communicate there is need to have something in between them. The middle thing is termed commonly as middleware. Figure 1 shows the middleware in between university's systems.

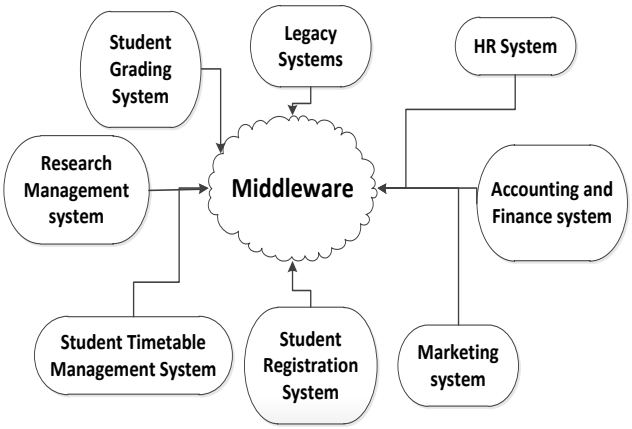

Figure 1: Middleware

CORBA (Common Object Request Broker Architecture) [4] , RMI (Remote Method Invocation), RPC (Remote Procedure Call), SOAP (Simple Object Access Protocol), Jini [5] are all middleware technologies. CORBA, RPC, RMI, Jini are network specific complies only to object based applications and networks whereas SOAP is network independent. [6] Service Oriented Architecture is the modified and better form of middleware.

2.2 Service Oriented Architecture
Service Oriented Architecture (SOA) is the architectural design and pattern which is used to provide services to different applications. Its goal is to achieve loose coupling between interacting components of applications. Web Services, Corba, Jini, etc are the technologies used to implement SOA.

In SOA a service is complete self-reliant and un-associated unit which provide some functionality for example providing data from the student grading system to the student registration system or retrieving information from the appraisal system and provide it to the payroll system.

Main benefit of SOA is that it can provide the means of communication between completely different applications (built in different technologies). The services are also completely independent and reusable and the nature of reusability provide the less time to market. All the services technologies needs to be implemented using the SOA design pattern and need to be designed on the basis of SOA to get maximum benefit.

2.3 Web Services

Web service is the SOA technology with additional requirements of using internet protocols (HTTP, FTP, SMTP, etc.) and using XML (Extensible Markup Language) for message transmission. [7] These are application components which communicate between different applications using open protocol. Open protocol is the web protocol for querying and updating information. These components can be used by different kinds of applications to exchange information. HTML (HyperText Markup Language) and XML are the basics of web service implementation. [7] In web services, service requestor search for the desired service from the service register and if found it registers the services with itself which is binding (the service binding is the attachment of service with requestor by the service provider). Service register keeps record of all available services inside system. Figure 2 describes the architecture of web services.

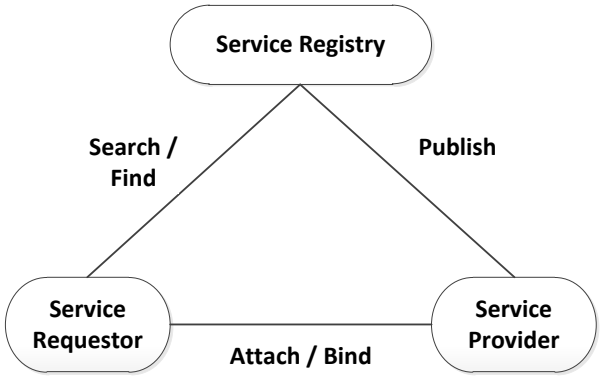

Figure 2: Web-Service Architecture

There are two main uses of web services: [8]

1. Reusability: in applications sometimes we need different things frequently for example certain forms, units conversions, calculations, reports, language translation, etc. Web services can provide solution to frequently used applications.
2. Solution to interoperability: If different kind and nature of application needs to link together web-service is there to solve this problem. It can provide a channel to share data among different applications. For example sharing certain data between HR and Finance.

2.4 WSDL (Web Service Descriptive Language)

WSDL is a language to describe web services and providing the link to access these web services. It is acting as a publisher in web service architecture. It is the XML document which is recommended by W3C (World Wide Web Consortium) in 2007. In WSDL XML describes the service and the address from where applications can access the service. [8] WSDL predecessors were COM and CORBA - IDL. It provides the following basic information about service:

1. What is service about: means what service can provide to the application and what it does?
2. How application can access the service? The name, format and protocol of service.
3. What is the location of service? The address of service i.e., URL (Uniform Resource Locator). [7]

For example if we have a service named robot_program which provides the status of robot (running program) then service interface which needs to be registered with WSDL should be service_name = robot_program, service_provides (returns) = program_running_on_robot. This is the interface of service.

2.5 UDDI (Universal Description, Discovery and Integration)

It is directory service / registry service where applications can register and look for web services. It is Platform independent framework. It is acting like service register in web service architecture. It uses HTML, XML and DNS (Domain Name Server) protocols which enables it to become the directory service. [8] It defines the keyword search, categories and classification for an application and registers it into business directory. In that way it is making the application easier to be approached by the customers online. It provides the following information about the service / business online:

1. Who is the one has the business? It requires name and contact information.
2. What is business category? Is the business belongs to manufacturing, education, health, etc.
3. What is the URI (Universal Resource Identifier)? The direct sub contact of service. For example http://www.cranfield.ac.uk/ is URL and http://www.cranfield.ac.uk/images/slider-items/homepage/cranfield_1000x500_turbines.jpg is URI which is specific image (resource) address.
4. How the given interface can work. Describe the complete process how we can use specific service resource. What is the interface and what are the parameters it is taking. [8]

2.6 SOAP (Simple Object Access Protocol)

It is a messaging / invoker in web service architecture. It is use to send messages in between the service and service consumer; and in between service consumer and service registry. It used to communicate with web service. It is a message framework which transfers the information between sender and receiver. SOAP doesn't define the service but defines the mechanisms for messaging. It binds the client to the service. Three main operations of the web services are performed by SOAP:

1. Publishing: Making the service available for other applications to use. The web services are published in UDDI (Service Registry).
2. Finding (search): Locating the web service. Finding the web service and its location.
3. Binding / Attaching: Utilising the web service.

Figure 3 describes web service architecture with its technologies like SOAP, WSDL and UDDI.

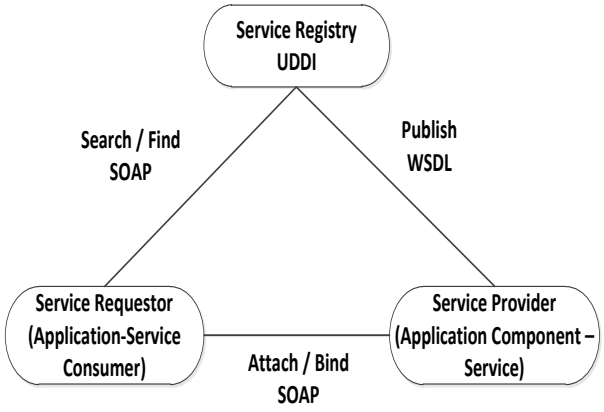

Figure 3: Web-Service Architecture with technologies

2.7 SOA – A solution to spaghetti architecture

In a fairly medium scale to large software architecture where there is need to integrate or communicate between different kinds of applications the introduction of links can make the architecture messy and it is called spaghetti architecture [9]. Figure 4 shows that problem in detail. It makes the system less flexible and expandability is the nightmare. Service Oriented Architecture (SOA) makes the architecture flexible and expandable. [9]

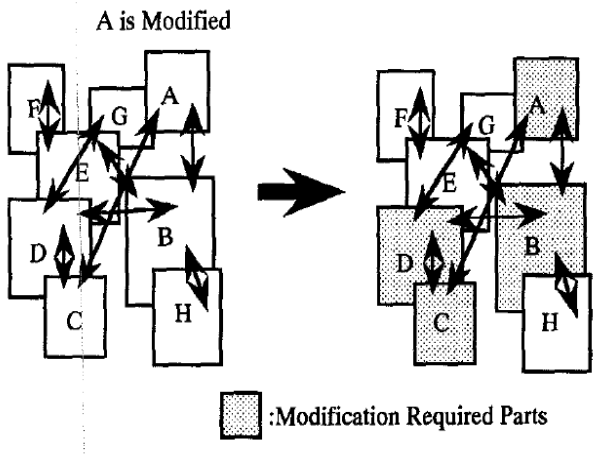

Figure 4: Spaghetti architecture [10]

3. SUMMARY

After analysing different ways of software integration it is clear that every technique and architecture has its own implications. To apply certain technique or architecture it is very important to analyse the systems to find out the suitability and get the maximum advantage out of it. This review helped the research of the system which is under development in the Aero-structure Assembly and Systems Installation Research Group of Cranfield University. The middleware is the main part in the ongoing research to integrate systems.

4. ACKNOWLEDGMENTS

This research is supported by the EPSRC Centre for Innovative Manufacturing in Intelligent Automation.

5. REFERENCES

[1] R. Pressman and B. Maxim, *Software Engineering: A Practitioner's Approach*, 8th ed. McGraw-Hill Higher Education, 2014.

[2] P. Johnson, "Enterprise Software System Integration - an Architectural Perspective," 2002.

[3] F. A. Cummins, *Enterprise Integration: An Architecture for Enterprise Application and Systems Integration*. John Wiley & Sons, Inc. New York, NY, USA 2002.

[4] A. L. Pope, *The CORBA Reference Guide: Understanding the Common Object Request Broker*, 1st ed. Addison Wesley, 1997.

[5] K. Arnold, R. Scheifler, J. Waldo, B. O'Sullivan, and A. Wollrath, *Jini Specification*, 1st ed. Addison-Wesley Longman Publishing Co., Inc., 1999.

[6] B. Yurday and H. Gümükaya, "A Service Oriented Reflective Wireless Middleware," in *Service-Oriented Computing – ICSOC 2006*, 2006, pp. 545–556.

[7] N. A. Good, "HTML and XML," in *Regular Expression Recipes for Windows Developers*, Apress, 2005, pp. 243–270.

[8] E. Newcomer, *Understanding Web Services: XML, WSDL, SOAP and UDDI (Independent Technology Guides)*, 1st ed. Addison Wesley, 2002.

[9] a. K. Harikumar and R. Lee, "A model for application integration using Web services," *Fourth Annu. ACIS Int. Conf. Comput. Inf. Sci.*, pp. 468–475, 2005.

[10] H. Wataya, K. Hayashi, J. Toyouchi, T. Aizono, T. Iimka, S. Shibao, M. Omura, and M. Oku, "Integration of Control & Information Systems by Open Autonomous Decentralized System Architecture and its Application for Distributed Manufacturing System," *IEEE*, pp. 147–154, 1997.

2015-07-01

Software system integration - Middleware - an overview

Asif, Seemal

Foundation of Computer Science

Asif S, Webb P. (2015) Software System Integration - Middleware - an Overview. International Journal of Computer Applications, Volume 121, Issue 5, July 2015, pp. 27-29
http://dx.doi.org/10.5120/21538-4547