Lessons Learned from Migration of Web Services to Workflow Services on an Enterprise Service Bus in the Cloud: Toward Self-Organizing Service-Oriented Architecture

Mythreyee Sunil and Sam Chung

Institute of Technology
University of WA, Tacoma
1900 Commerce St., Tacoma, WA, USA 98402
{sunilm, chungsa}@uw.edu

Abstract. An ideal goal of self-organizing services requires self-composition of a set of services on demand. For this goal, emerging composition approach in cyber space needs to be understood and the understanding can help the development of autonomic service composition toward self-organizing service-oriented architecture. Modern information systems focusing on cyber properties have developed service standards using web technologies called web services. Also, technologies and engines for composing the services in parallel with the advent of cloud computing have been recently emerged and allow us to have highly available, on-demand and scalable development and deployment. An Enterprise Service Bus (ESB) provides a good platform for effective, resilient, secure and rapid deployment and execution of simple or composite services. Workflows enable service composition with improved runtime to manage the services. In this paper, we demonstrate the migration of web services to workflows on an ESB in the cloud. Based upon the empirical results, we propose the guidelines to migrate a service based application onto the cloud using emerging technologies like Workflow and ESB. We argue that these guidelines can help to discover how services can be composed to be another new service.

Keywords: Service Composition, Cloud Computing, Service Oriented Computing, Web Services, Enterprise Service Bus, Workflows

1 Introduction

Cloud Computing provides a good infrastructure for delivering computing as a service with several characteristics like scalability, high availability, location and device independence, easy maintenance, security and reliability [1]. An Enterprise Service Bus (ESB) [2] that is based on the principles of Service Oriented Computing (SOC) provides a good platform for rapid development and deployment of services, applications and data with benefits like flexible communication between components on premise and the cloud exposing the endpoints securely, effective management of components, message queues for communication that provide better resilience and load balancing and easy deployment of all components at once.

Windows Workflow foundation [3] provides a flexible framework for declarative and reactive programming. Declarative programming makes programming easier by providing a designer using eXtensible Application Markup Language (XAML). It also supports Java and other frameworks along with .NET and is designed in such a way to reuse the existing assets. Reactive programs react to a certain stimuli and have the capability to store the state when the programs go idle and resume from where they have stopped thus providing better thread and process agility. A workflow is nothing but a tree of activities where an activity could be program code such as something basic like assignments or composite like control flows such if/else or for/while loops. Activities are also serializable. Workflows can be integrated with Windows Communication Foundation (WCF) to provide effective, flexible and secure communication by exposing the workflow services and also for efficient use of system resources by providing a run time that helps persistent data between workflows and provide better process management [4, 5, 6, 7].

An ESB like AppFabric [9] provides a flexible communication framework between components on premises or on cloud. Services can be composed by storing the dependency information and these services can be reused multiple times [8]. Some of the advantages as per [9, 10] are: secure communication, loose coupling by connecting disparate applications, cost effective routing mechanism without the need to worry about firewalls etc. [11], changes in infrastructure could be easily handled since a unique Uniform Resource Identifier (URI) is allocated to all the services registered which would benefit any service provider significantly, and better scale and resiliency due to the queuing and topics service [11]

By understanding the benefits of the cloud and the SOC technologies through the migration from the web services to workflow services on an ESB in the cloud, it would be possible to rethink self-organizing Service-Oriented Architecture (SOA). Binder et al. [12] mentioned how much automated service composition is important for self-organizing SOA. However, up to our best knowledge, there have been no case studies on migration of web services to workflows on an ESB in the cloud. Also, there are no specific guidelines for the migration either. The problem statements would then be as follows:

- How can we reengineer web services to workflows on an ESB in the cloud and what would be the benefits of this approach, compared to migration to an ESB in the cloud?
- What are the guidelines of migration to workflows on an ESB in the cloud?

For the first objective, we use a case study to demonstrate migration of Library Management web services application to Library Management workflow service application on an ESB in the cloud. The second objective is to formulate the guidelines for the migration by discovering the best practices through the migration case studies.

2 Related Work

Hong Zhou et al. [13] talk about re-engineering a legacy enterprise system using ontology development and host it in a cloud computing environment. However, neither ESB nor Workflow is used. Michelbach et al. [14] describe the benefits of combining SOC and cloud computing technologies for better program design, implementation and operation which result in improved Return on Investment (ROI). Although an ESB is used, it was not used with Workflow on the cloud. Yan Liu et al. [15] discuss about using SOA for service composition for systems biology workflows. No ESB is used although Workflow is employed. No guidelines migrating web services to workflow services are proposed. Table 1 summarizes the features of our interest that are covered in these papers

Table 1. Summary of features covered in the related work

Features	Hong Zhou et al. [13]	Michelbach et al. [14]	Yan Liu et al. [15]
Re-engineer a Service Oriented System to host it on the cloud	Yes	Yes	No
ESB in the cloud	No	Yes	No
Workflow for Service Composition	No	No	Yes
Work flow on ESB	No	No	No
Migrating web services to workflow services	No	No	No
Guidelines for migrating to workflow services	No	No	No

3. Migrating a Service-Oriented System to Workflows on Windows Azure AppFabric

Service-Oriented Library Management System: The Simple Object Access Protocol (SOAP) and REpresentation State Transfer (REST)-ful web services based Library Management System consists of a service producer who is a WCF service application and a service consumer who is the Model-View-Controller (MVC) 3 web application respectively. ADO.NET Entity Framework (EF) is used for importing the data from the Library Database as per the guidelines in [16]. It works on the principle of EF and generates a `.cs' file that contains the entity classes and the object context to the library database in order to enable easy data updates using Language INtegrated Query (LINQ) queries. In order to prevent exposure of the EF classes, separate classes are written for exposing the WCF data contracts. Internet Information Services (IIS) web server is used for hosting the web services.

The migration to workflows on Windows Azure AppFabric: The following are the three steps for migration. Firstly, host the web services on the cloud to compare with the hosting of workflows on ESB in the cloud. Microsoft's Windows Azure was selected to be the cloud computing platform since it has the best support for .NET technologies and Structured Query Language (SQL) Azure is very similar to SQL Server which was the database engine for our case study. The REST Target system was chosen to be deployed on the cloud due to its simplicity and is best suited for web programming. A tutorial like [17] was used as a reference to deploy our REST Target system on Azure.

Secondly, use Service Composition using Windows Workflows. The WCF service application is converted to Workflow service application as follows using [4] as a reference:

- Selecting a messaging format: The message format could be one of these-ReceiveMessage, SendMessage, ReceiveAndSendMessage, and TransctionalSendReceive.
- Defining the contracts for the message exchange: There is no need of defining a separate Interface compared to conventional WCF services. Multiple arguments are supported for message exchange for example: Message is used if there is single argument and Parameter is used for multiple arguments which can be primitive and custom data types. It is also possible to define Service Name Space. XML and data serialization supported by default so there is no need to import 3`^{d} parry libraries like Json.Net for data deserialization
- Defining the custom activities in the workflow for service: The activities could be any of the following - designer for .xamlx file, Composite activities, Code activity, and Native code. The web configuration can be defined for custom activities.

Lastly, host the Windows Workflow (WF) Services on Windows Azure AppFabric Service Bus. The Workflow Services are hosted on the AppFabric Service bus using [18] as a reference. The following are the required steps:

- Namespace creation and Service Name/Private Key generation
- Expose the WF/WCF services using the Service bus libraries
- Enable the service to Auto Start
- Modification of the client to consume the exposed service using the service name and private key

5 Discussion and Future Work

The following are the guidelines that we discovered during the migration from web services to workflows.

- If services are built once and reused multiple times, use workflows.
- For parallel processing without object-oriented programming, use workflow.
- Use parallel workflow works for asynchronous scenarios like an expense reporting system
- Use Message Relay instead of Brokered message for synchronous scenarios

There are several benefits by migrating web services to workflows on an ESB on the cloud. Workflows provide a developer friendly declarative programming framework that facilitates to spend time on business logic rather than spending time in writing code. Workflows can be exposed effectively and securely for communicating entities. The system resources are also used efficiently due to the run time that helps persist data between workflows and provide better process management. We observed that using an enterprise service bus helps for better secure way of exposing end points against finding a way to work around corporate firewalls, access controlled lists. It also provides message queues that help in load balancing and highly scalable.

Some features of ESB that can be attempted in the future are Queues, Topics and Subscriptions and Access control lists. Also, we need to deploy our example case onto other cloud computing environments like Amazon EC2 for comparison purpose.

References

[1] Geoffrey Raine, Cloud Computing and SOA, http://www.mitre.org/work/tech_papers/tech_papers 09/09 0743/09 0743.pdf, 2009, Accessed on June 25, 2012.

[2] David Chappell, Introducing Windows Communication Foundation in .NET Framework 4, Chappell & Associates, http://msdn.microsoft.com/en-us/library/ee958158.aspx, accessed on June 25, 2012

[3] David Chappell, The Workflow Way-Understanding the Windows Workflow Foundation, http://msdn.microsoft.com/en-us/library/dd851337.aspx, accessed on June 25, 2012

[4] Maurice de Beijer, Introduction to Windows Workflow Foundation, http://msdn.microsoft.com/en-us/netframework/ff802403, Accessed on June 25, 2012.

[5] Matt Milner, A Developer's Introduction to Windows Workflow Foundation (WF) in .NET 4, http://msdn.microsoft.com/en-us/library/ee342461.aspx, Accessed on June 25, 2012.

[6] Aaron Skonnard, A Developer's Introduction to Windows Communication Foundation 4, http://msdn.microsoft.com/en-us/library/ee354381.aspx, Accessed on June 25, 2012.

[7] Workflow Services, http://msdn.microsoft.com/en-us/library/dd456788.aspx, Accessed on June 25, 2012.

[8] Jiehan Zhou, Daniel Pakkala, Juho Perala, Eila Niemeld, Jukka Riekki, Mika Ylianttila, Dependency-aware Service Oriented Architecture and Service Composition, Proceedings of the International Conference on Web Services (ICIW) 2007.

[9] Microsoft, Windows Azure AppFabric Overview.

[10] Windows Azure SDK for .NET, http://www.windowsazure.com/en-us/develop/net/, Accessed on June 25, 2012.

[11] .NET Development Center, http://www.windowsazure.com/en-us/develop/net/how-to-guides/, Accessed on June 25, 2012.

[12] Walter Binder, Daniele Bonetta, Cesare Pautasso, Achille Peternier, Diego Milano, Heiko Schuldt, Nenad Stojnic, Boi Faltings, and Immanuel Trummer, Towards Self-Organizing Service-Oriented Architectures, Proceedings of the 2011 IEEE World Congress on Services (SERVICES 2011), pp. 115-121

[13] Hong Zhou, Hongji Yang, Andrew Hugill, An Ontology-Based Approach to Reengineering Enterprise Software for Cloud Computing, Proceedings of 2010 34th Annual IEEE Computer Software and Applications Conference (COMPSAC), pp. 383-388.

[14] T. Michelbach,,B. E. Amstutz, T. T. Usrey, G. Frithl, M. Ziegler, C. Tueffers, J. Stogdill, A. Agueda Mate, Cloud Computing and Service Orientation in the SSA programme Space Surveillance services and data in the Cloud, European Space Surveillance Conference 2011.

[15] Yan Liu, Ian Gorton, Adam Wynne, and Anand Kullkarni, Scientific Workflows Composition and Deployment on SOA Frameworks, Proceedings of the 6th International Symposium on Service Oriented System Engineering (SOSE 2011), pp. 134 — 139.

[16] Mike Liu, Implementing a WCF service with Entity Framework, http://www.codeproject.com/Articles/127395/Implementing-a-WCF-Service-with-Entity-Framework, Accessed on June 25, 2012.

[17] Deploying an ASP.NET Web Application to a Windows Azure Web Site and SQL Database, https://www.windowsazure.com/en-us/develop/net/tutorials/web-app-with-sql-azure/, Accessed on June 25, 2012.

[18] Implementing Azure AppFabric Service Bus, http://www.dotnettwitter.com/2011/09/implementing-azure-appfabric-service.html, Accessed on June 25, 2012.