International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:268
Vol.2, No.3, March 2013 DOI:10.15693/ijaist/2013.v2i3.48-51

Enterprise Integration Using Boomi Tool

Mr. Vishal R.More
Dept. of Computer Science and Engineering
G.H.Raisoni College of Engg & Mng. Amravati
SGBAU Amaravati University Amravati

Mr. M.M. Bartere
Dept. of Computer Science and Engineering
G.H.Raisoni College of Engg & Mng. Amravati
SGBAU Amaravati University Amravati

Abstract— **Application Integration is an age old concern for any enterprise, as no application can live in isolation. Typically integration projects always involve a set of distributed applications with multiple data formats. These applications interact through multiple transport protocols over multiple platforms. These diversifying parameters often makes architecting and implementation of enterprise integration extremely complex.**
The conventional approaches of enterprise integration (EI) are often quite expensive both in terms of time, effort and money. Most of the traditional integration middleware are very costly and need lot of upfront investment. It also involves lot of hand coding to implement the integration scenario using these middlewares.
Following in line with today's business mandates for IT teams such as productivity, scalability, elasticity, reduced cycle time and baked in quality, we have to look into alternate set of EI mechanism. This mandates trigger the emergence of a set of new set of technologies and platforms in the EI domain.
In the current paper with the help of an example, we would like to present our experiences with some of these modern generation EI technologies along with pros and cons of each of them.

Keyword: integrate distributed applications using tools as well as SQS (tool: apache camel, boomi, SQS service provided by amazon)

I. Introduction

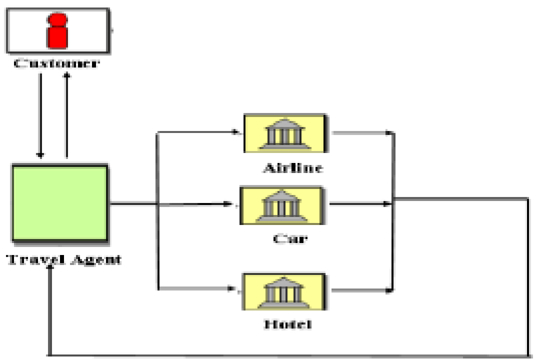
Enterprise Integration involves a set of distributed applications. Each of these applications supports various data formats such as CSV, XML, flat file etc. These applications communicate through multiple transport protocols such as HTTP, FTP, JMS etc. The diversity among the applications makes the implementation of EI project pretty challenging.
Traditionally, companies had limited choices for solving application integration problems. Either they can use complex software platforms (such as BPM, EAI, etc.) or write customised code, or use web service. Application complexity along with platform complexity makes the task at hand even harder. The typical challenges faced by the Integration projects are:

- ***Integration of Distributed Systems:*** Complexity rises when multiple applications talk over multiple transports on multiple platforms.
- ***Greater Learning Curve*:** Companies had to spend a lot of effort on educating the employees to learn various complex EI Tools and platforms.
- ***Expensive and Time Consuming*:** Quite often developing and maintaining custom code required for EI is time consuming and labour intensive

Travel Agent Application is a famous example in the EI domain. In the subsequent sections with the help of Travel Agent application we will describe how we have used the new breed EI technologies to solve some of the challenges of enterprise integration.

II. Introduction to travel agent application

To make a travelling plan, the customer usually calls a travel agent to book his Air tickets, Car and Hotel. The following figure represents the overview of a Travel Agent application:



The message flow of the application is as follows:
1. Receive the customers travel request
2. Split the request into three parts: airline, hotel and car.
3. Obtain the cost and other details from all the three components.
4. Aggregate these responses.
5. Send the result back to the customer.

We had used Enterprise Integration Patterns to model the Travel Agent scenario.

III. Implementation options

The EIP based model provides the specification of the Travel Agent Scenario. Now what are the options for the actual implementations? For implementation we explored couple of modern tools / platforms in the EI domain. In the subsequent sections we will discuss such implementation options:

- Cloud based Reliable Messaging Fabric Amazon Simple Queuing Service,
- Cloud based on demand integration as a service

.A. Implementing with SQS from Amazon

Amazon SQS is a reliable messaging fabric for sending asynchronous messages. SQS acts as a buffer between the component producing and saving the data and component receiving the data for processing. Using SQS we can build highly scalable applications.

SQS ensures delivery of each message at least once, andsupports multiple readers and writers interacting with the same queue. Since SQS is deployed in Amazon data centre and available as a service we can avail the service only whenever it is required that is on demand basis. You can also increase the load as per the current requirement. That way it allows us elastic scalability.

Figure 4 shows the implementation of travel agent using Amazon SQS. The Travel Request Queue acts as a communication link between the Customer and the Travel agent. Travel Agent polls the Queue to see for the availability of any messages by the customer. On receipt of any messages, it would execute the functionality specified in the message.

In the similar way when the travel agent sends the request to the Airline, Hotel and Car rental , these request initially go to their respective queues i.e. Airline, hotel and Car rental Queues. In the end the consolidated response is sent to the Travel Agent Response queue.

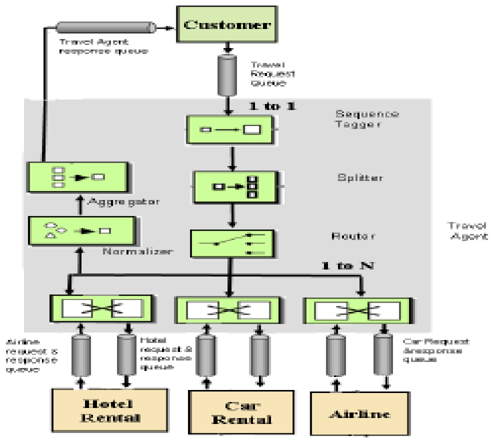


Figure 4. Implementation of Travel Agent using SQS

All major EI patterns implementation in SQS have to be either written by hand or we can use an existing API such as Camel. For example in case of Travel Agent shown above we had to hand code the router, Translator, branching because they were not provided by SQS

A possible solution to this will be to use existing APIs like Camel which we have already seen in the previous section. SQS can act as a reliable messaging fabric for Camel .In case of travel Agent we can use the simple DSL api of Camel to implement the router, splitter and translator.

Pros:

- Amazon SQS gives freedom from big infrastructure investment allowing vendors to focus only on business related problems
- All the non functional requirements such as availability, scalability, reliability are taken care of by Amazon.
- SQS is pay as per use service. We pay for the time period we
- It gives API in multiple programming languages.

Con:

- Major drawback with SQS is that the learning curve is much higher. We need to have seasoned programmers to do the coding.
- Cost of developing and maintaining the code base is time consuming and labour intensive.
- If our Application needs to run 24/7 , using SQS will be quite costly

B. Implementing with Boomi

As SQS based implementation involves lot of hand coding, we explored integration-as-a-service which is completely configurable.

Boomi connects any combination of Software-as-a-Service (SaaS), cloud, and on-premise applications without installing and maintaining any software package or appliance.

With Boomi we can implement the integration scenarios through declarative programming. Thus the implementation can be done much faster. Boomi requires no coding, therefore no APIs are required. Its core competency is just Integration.

Boomi works in three stages:

1. *Build*: The Build Tab is where we design our application. Boomi provides set components which provide similar functionality as that of EI patterns.

2. *Deploy*: The Deploy Tab is where we select the atom on which the application will run. An atom is a runtime engine that executes the entire integration process. For integrating on-premise applications the atom resides in our machine. For integrating Off-premise applications the atom resides at the Boomi side. One atom can have multiple processes running on it.

3. *Manage:* The Manage Tab is where we view the atom status, Process Execution results, and troubleshoot Process errors. It provides a full fledge dashboard to view atom results.

The figure below shows the implementation of Travel Agent using Enterprise Integration Patterns in Boomi.

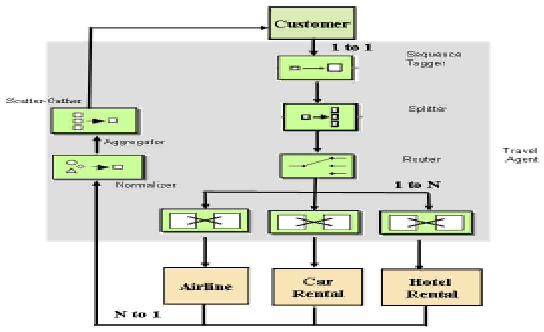


Figure 5. Implementation of Travel Agent using Boomi

As specified in the message flow, travel agent has to retrieve the detailed travel request, from the customer. The *sequence tagger* will tag the sequence in our request i.e. in first in first out order. After that the Travel Agent needs to split the request using a *splitter* into three different parts- Airline, Hotel, and Car. After splitting the request, it is routed to the specified location using a *router*. We have used message *translator* to communicate our request to the Airline, Car and Hotel Booking as they all may accept different formats. Once the Airline, Car and Hotel reply with their cost/day and other details, we use a *normalizer* to process the request that arrives in different formats. The *Scatter-Gather* aggregates the individual cost into a single response file for the Customer.

For all these patterns Boomi provides a set of Components. Boomi has a start connector in which we have to mention the type of connection (GET /SEND) and the type of operation (Flat file/ CSV/Database/[any off-premise connectors]).

Figure below shows the integration of Travel Agent to Airline, Car and Hotel process using Boomi components.

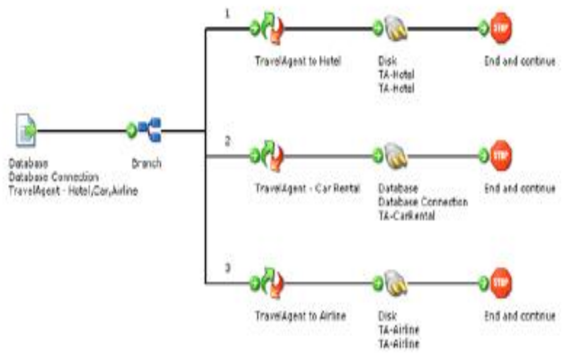


Figure6. Travel Agent-Airline Hotel,Car

We have used Database connector (Travel Agent) as the start connector. We have used the branch component of Boomi which splits our request in three parts. Data maps have been used which performs the translation.

Pros:

- Boomi is on demand configurable framework which requires no coding.
- It is a drag and drop application which ensures ease of use.
- No hardware or software installations are required
- Boomi has unlimited scalability that supports simple to complex business logic

Cons:

- As Boomi is cloud based integration solution, confidentiality may be a concern.
- If our Application needs to run 24/7 , using SQS will be quite costl

IV Security concern of boomi

Boomi Atmosphere to address security at three distinct points: the network and facilities infrastructure, the application and platform layer, and at the data level. This three-tiered security approach ensures that your data is never exposed to unauthorized parties, remains safe in transit between applications, and that you can access your data whenever and wherever you want.

A. Network & Facilities Infrastructure Security

The Boomi infrastructure has been deemed SAS 70 Type II compliant as per the audit requirements of the American Institute of Certified Public Accountants. The configuration of the data center includes SAS 70 Type II attestation, best-of-breed security (routers, firewalls, IDS and DDoS protection), redundant IP connections to world class carriers terminated on our carrier grade network, redundant UPS power, diesel generator backup, and HVAC facilities, and multipoint monitoring of key metrics alerts for both mission critical and ongoing maintenance issues.

B. Application & Platform Security

The Boomi Atom has been carefully architected with your security in mind. Because the Atom can reside on your network or be hosted in our data center, it is important that there are extensive security measures in place in order to prevent any compromise in your data or the Atom. During installation, the Atom download and all of its contents are verified and authenticated by the Boomi data center before deploymentThe Atom communicates information to the Boomi AtomSphere in two modes,ongoing/automatic communications and user-initiated communications. During ongoing/automated communications the Atom transmits operational information to the Boomi AtomSphere data center, including online status to monitor uptime, tracking information to catalog process executions, integration process configuration updates, and periodic checks for updates to the Atom code. User initiated communications occur only upon the request of an authorized Boomi AtomSphere user and include logging information about a specific integration process, capturing error messages and failures for diagnostic purposes, and retrieving connector schema to define mapping and rules for new integration processes

C. On-Premise Data Communication Security – After the Atom is deployed behind the firewall, the Atom will be in contact with the data center for ‘tracking’ and ‘status’ information. No inbound firewall ports need to be opened in order for the Atom to communicate with the data center. The Atom always initiates the connection and there is never ‘pushing’ of data from the data center to the Atom. When the Atom initiates the connection to the data center, it authenticates the data center before sending data using an SSL handshake and a digital certificate.

D. Data Communication Security Standards – To ensure the security of data in transit, Boomi AtomSphere makes use of the latest and most stringent data communication security standards. All communication from Atom to data center uses SSL 128 bit encryption. All outbound communication from Atom to data center is HTTPS. Atoms use a standard SSL Handshake to authenticate with platform.boomi.com.

E. Password Encryption Security – When a user registers and activates an account, Boomi generates a private/public x509 key (PKI). We store both the public certificate and the private key in our secure data center. When creating a Connector, users will be prompted to create their password. The password is encrypted and stored for the account. Only the account owner can decrypt with the password needed to unlock the encrypted private key. When atoms are deployed, the entire encrypted string is deployed to that Atom and the credentials supplied unlock the password at runtime.

F. Certificates – Certain AtomSphere Connectors use certificates in order to ensure security when transmitting data across a communication protocol. Connectors such as FTPS, SFTP, HTTPS, AS2, and many others require the use of certificates in order to encrypt data and channels and to verify the digital signature of the person sending the data. The Certificate Component can use an existing key obtained from a certificate authority such as Verisign/Thawte or a key generated by Boomi.

G. Data Security

It is important to note that at no point during the integration process does Boomi store data. Boomi AtomSphere is engineered to optimize interoperability of applications and facilitate your integration processes without saving your data in our data center, unless specifically configured to do so.

IV. Conclusion

Many companies view integration as a series of big, messy projects that require extensive resources and multiple tools from a variety of different vendors. However, through small, strategically planned, tightly-scoped projects, our business can achieve immediate benefits and create demonstrable short-term value – particularly during periods of financial crisis – while preparing for continued prosperity through growth and expansion in the future.

The integration solutions highlighted in this paper can be considered as a continuous spectrum for solving EI problems. These solutions can be considered by many to be quick fixes. All these solutions have their own pros and cons. We need to carefully choose an appropriate solution depending on the current context

For quick implementation, we may use DSL based open source integration middleware i.e. Camel. If we don't want to invest much on infrastructure we can make use of SQS or Boomi. Again if we don't have seasoned programmers and we need to go for integration projects, we can use Boomi as the solution.

Organizations must strive to develop an aligned integration strategy that is accessible across each of the layers taking the risk factor and cost into account. By applying these technologies based approaches and solutions an enterprise can be assured of best likelihood of success.

Uses Cases

Kelly-Moore, one of the recognized companies in the paint industry, they needed to give their sales organization one view of all customer records, and it needed to happen quickly. They had recently selected Salesforce as their CRM solution due to its flexibility and ability to grow along with their needs and business requirements. Kelly-Moore needed to quickly and capably integrate information from ERP and POS systems into their new CRM system before launching to their sales organization. And They used Boomi tool for their integration.

Emtec relies on three systems to manage their back-office (Salesforce.com (CRM), SAP (ERP),VARStreet) However, the systems were working independently of one another. For instance, quotes were generated in VARStreet and the same data needed to be logged as an opportunity in Salesforce.com so that it could be tracked in the sales pipeline. This process of using systems that sit in their own silos resulted in a high incidence or errors, duplicate data entry and an inability to generate reliable, up-to-date data and analysis of what was really goingoninthebusiness. Emtec selected Boomi tool for seamless integration

References

[1]. http:// www.boomi.com/solutions/taleointegration
[2]. http://camel.apache.org/ldap.html
[3]. http://janstey.blogspot.com
[4]. http://www.enterpriseintegrationpatterns.com
[5]. http://aws.amazon.com/sqs/
[6]. https://ondemand.boomi.com/application/login/app.php
[7].http://www.boomi.com/dell/business/smb/application-integration/
[8].http://www.automationanywhere.com/solutions/application integration

Authors Profile

Mr.Vishal R. More received the **B.E.** degree in information technology engineering from the Sipna College of Engineering, Amravati, SGBAU University, Amravati, India, in 2010.Currently doing **M.E.** in computer science and engineering from G. H. Raisoni College of engg. And mangt Amaravati. in SGBAU University of Amravati, India. His research interest includes Enterprise integration using boomi tool which integrate different application by using tools

Prof. M.M. Bartere (Guide) received the **B.E.** degree in computer science and engineering from HVPM College of engineering Amravati, SGBAU University of Amravati, India, in 2007. And **M.E.** in computer science and engineering from Sipna College of Engineering Amravati, SGBAU University, Amravati,