Why You Never Want to Build a Custom Cloud Integration

The Hidden Complexity of Connecting to Apps in the Cloud

Executive Summary

When they sign the dotted line for SaaS applications, many business executives fail to see the bigger picture. This casual act creates a lot of headaches for the IT department and buyer's remorse for them. One of the more complex problems is integrating a hybrid cloud architecture that is partly on-premises and partly beyond the firewall. Fortunately, there is more than one way to solve this problem. How do you know which approach is likely to work best in your organization? This paper will explain some of the ins-and-outs of integrating to the cloud based on our coding experience.

Introduction

When you are given the task of integrating an on-premises application with one in the cloud, there are many factors to consider before you agree. Many business executives assume that integration is the simple process of moving data from point A to point B. They don't appreciate how far and fast SaaS vendors have evolved in creating an infrastructure that is very different from traditional IT. In a hybrid

cloud environment integrating is not as easy as moving data from one database to another. Data structures differ. Rules for synchronizing various data sets vary. Authentication methods diverge.

Then there is the political consideration of connecting to applications that change four times a year on average and sometimes as often as every two weeks. In the good old days, if the application was stable, the integration just worked. Once in place, you could, for the most part, forget about it. In a hybrid cloud environment, you have to fix the connection every time someone else breaks it.

Here are some of the factors to consider when building and maintaining custom integrations.

In the Cloud APIs Change Rapidly

One of the benefits of a multi-tenant SaaS architecture is that new versions of the application can be deployed to all customers quickly and uniformly. As a result, APIs also change rapidly to keep up with new functionality and other structural changes to the core application.

Your Users Can Customize Cloud API's

Changes to the API aren't limited to the vendor. Most SaaS applications allow users to customize their instance by adding custom fields and objects. These changes may be reflected in the API. User-defined custom fields require your code to be modified to accommodate them. Usually, it will be your application administrator adding these fields. If the integration is custom coded, almost every change will mean the administrator is pulling your developers away from their current projects to make sure the integration is working correctly.

Cloud APIs are Unique

Each SaaS application has its own API, its own rules, and its own nuances. You need to understand the API not only of the target app but every application you plan to connect to it. When you choose to custom code your integration, your developers will need to learn multiple API standards and how they function with other technologies. Many APIs have their own programming languages, and even the ones based on standards such as SOAP or REST or WSDL will always have unique capabilities, schemas, authentication requirements as well as transaction rules. These nuances make every app – to some extent – unique.

Monitoring Tools

Errors happen. Systems become temporarily unavailable. What happens when the integrated applications don't respond as they should? How do you find out what went wrong and why? As you are developing your custom code, it would be prudent to include a reporting mechanism for logging and managing information about the integration. Ideally, the reporting mechanism would also alert the appropriate people when something does go wrong.

These tools are beneficial for pinpointing exactly where a failure has occurred. Since most integrations are comprised of numerous stages (e.g., connections, transformations, workflow), only detailed logs provide the context to easily determine where a fault has occurred and the best ways to fix it.

Security and Connections

SaaS applications present a unique challenge when it comes to integration because being on the internet requires Web Services to carry a lot of heavy baggage. First, there is security to consider. Today most every call to the application is transmitted over a secure internet connection (e.g., HTTPS). Some APIs require separate authentication calls before they allow other data to be transmitted.

The multi-tenant nature of SaaS applications often means that there are limitations on the number of times you can make a call to the application in a given time. Other APIs limit the amount of data you can move in one call. Compounding the complexity of connecting to these APIs is the fact that their specifications change often. Hardcoding against a WSDL all but guarantees limited scalability and poor performance. You will need to code around these limitations.

Redundancy and Performance Optimization

Network connections, and by extension web-based APIs, are never going to be 100% reliable. There will be times when calls time-out or connectivity goes down. Your custom code should provide procedures to deal with these inconsistencies. They should include processes for guaranteeing message delivery, failure notifications, and other error handling schemes. Ideally, you should also include processes for data chunking, parallel processing, and data streaming technologies to improve performance. These features allow you to break your data into pieces that the Web Service can handle, allowing you to move more data, faster – without running up against the built-in limits of the API.

Because every SaaS application has its nuances, these features will need to be customizable, including the ability to define data chunk sizes and the number of calls per minute among other things.

Conclusion

Many companies look at the upfront costs of buying an integration platform and think they can accomplish this in-house for less money and with more control. For some very technically savvy companies this might be the case. However, for most of the world custom coding is not a long-term solution. Over time, security, performance, the lack of scalability and dependence on programmers will balloon costs and create an unmanageable collection of hand-coded scripts. Even a company like Salesforce.com uses our Jitterbit platform extensively to handle its internal integration needs.

Because it is so easy to get up and running with SaaS applications, the natural tendency is to cobble together connectivity with your other systems haphazardly. It would be a mistake not to do it right. Life is too short to spend nights and weekends fixing custom integration code.