The Essential Cloud Integration Checklist

By Jordan Braunstein, Visual Integrator, Inc
Jordan.Braunstein@visualintegrator.com
www.MileHiCloud.com

With the current paradigm shift towards cloud computing, it's evident many companies are optimistically investing in cloud solutions. While some corporations are starting their private cloud infrastructure, other early adopters are driving out applications by developing directly in public cloud platforms, and even more are procuring software capabilities from the litany of Software as a Service (SaaS) vendors. No matter what your level of "Cloud Maturity" is or which deployment models you choose (SaaS, PaaS, or IaaS), there are certain essential architectural considerations when designing cloud environments. This is especially true for companies sharing information from their corporate applications into the cloud and vice versa-integrating existing systems with the cloud can be a daunting task. Sharing information with the cloud seems to be a concern for many CIO's, since they haven't established a level of trust with their cloud yet. This article explores the more critical characteristics of integrating to and from the cloud, and how to ensure your solution is stable, scalable, and interoperable. Consider this your Essential Cloud Integration Checklist.

(1) Security

Security is by far the #1 concern of IT departments when considering venturing into the cloud. Most of IT's concern centers around the fear of exposing private or sensitive information to non-validated users. Nobody wants to be the next company featured on CNN for losing their customer's data to an outside intruder. However, many of these same doubters would be surprised to learn that most Cloud providers have hosting and data centers that are far more secure than their own company's on premise center. Overcoming the risk of losing sensitive data is best accomplished through education, cloud provider research, and contractual stipulations. In other words, make sure your cloud provider follows certain standards (some of which are outlined below), ensure you have the proper service level agreements to protect your company in case the cloud has a failure, and educate any doubters to cloud's security capabilities. Standards such as Secure Socket Layers (SSL), Security Assertion Markup Language (SAML), and for encryption, authentication, and single-sign on (SSO) should be considered for any Cloud security architecture.

Cloud is based on a best-of-breed approach, and it's common for companies to be tied in to multiple cloud vendors for their cloud solutions. If you buy-into a multiple cloud provider approach, how do you go about managing users logging into each provider's proprietary, distributed, and multi-technology environments? This can become even more difficult when the cloud is off premise or a public Cloud. Policy Assertion standards such as Security Assertion Markup Language (SAML) will allow your systems to integrate via a single-sign on token and share security policies across technologies. This assumes your cloud provider supports SAML, which is an important consideration. Without direct SAML support, secondary options can be custom-based session tokens through cross-reference tables.

It's also important to ensure your cloud provider supports SSL and other encryption techniques in case any sensitive Data needs to be exchanged with the cloud. This is especially true with public clouds, where the data can be crossing a public wire as it flows to/from the cloud, and the data needs to be masked from any potential interceptor.

Access Control of software source control objects in the cloud is also important during design time or runtime-- objects, screens, and content artifacts should be protected to disallow any threats of viewing the source code, business rules, opportunity to change the configuration of an object, or running or executing the object by a disallowed user.

These security approaches become increasingly more important, since many cloud providers practice Multi-tenancy, which is the ability for providers to host multiple customers on a single resource. Examples could include your company's assets being hosted on a shared server, database, and disk drive as your most feared competitor. The cloud provider is responsible for separation of concerns and ensuring nothing is compromised.

(2) **Interoperability.**

Sending information to/from the Cloud is an important consideration because the Cloud can be both an authoritative source of information and a consumer of existing on-premise enterprise information. This information can be process, data, or business centric, but is still required to integrate with the Cloud to complete a business process. Normally, this integration needs to be electronic, automated, and seamless, so, its important for the cloud to have Application Programming Interfaces (API's) that are remotely accessible to other systems off the cloud (or on other clouds). These API's provide the channel or method for sending information into the Cloud, pulling information out of the Cloud, pushing information out of the Cloud, or modifying information in the Cloud. The most common technique for integration is webServices that comply with the WS-standards, and specific industry standards. This will allow the organization to leverage services and comply with architecture styles such as Service Oriented Architecture (SOA) to share information across technologies and platforms.

(3) Presentation Layer:

Working with different Cloud vendors, especially SaaS vendors, means you will have to familiarize with each vendor's proprietary User Interfaces (UI). This can be a daunting task and cause a lot of "swivel chair integration" for the end users who have to work in multiple UI's across cloud applications. Instead of swiveling and hand-jamming information across multiple systems, the preferred approach is to create a universal look and feel application that provides the "single version of the truth". This is best accomplished through approaches such as Composite Applications and Mash-ups that are design patterns that integrate disparate information sources into a single application screen or portal. This architecture has many benefits, such as: simplifying working environments, increasing end user efficiently, and protecting the business process from less human errors. However, this architecture design pattern is not always simple to implement, as it requires the Cloud vendor to expose information via a real-time remote API, preferably webServices. Other considerations include support for standards such as Web Services for Remote Portlets (WSRP), and Java Spec 168 (JSR 168) to embed remote content into the consuming portal screen.

(4) Federated Search:

With so much information sprinkled throughout the enterprise, it's becoming increasingly more important for companies to provide features to catalog, index, and expose content for search in enterprise systems. The Cloud is an enterprise system and the information contained within must be searchable. How is this accomplished if the Cloud is off-premise? The simplest approach is for the

Cloud vendor to index their own content and expose the searchable content via a remote webService API. This will allow companies to integrate the Cloud content with any pre-established searching software they have already standardized on and prevent end users from having to use multiple search boxes to find their information. Companies will then reap the benefit of "single box search", as having to swivel between multiple search User Interface's can be a frustrating experience to locating content and information. This concept is known as "Federated Search" since the content being searched can be hosted anywhere across the enterprise or the cloud, but the user doesn't need to worry about that as the search design complexities are abstracted from them—they simply have simple search UI that allows them to find associated content.

(5) Functionality and Usability

Working with multiple Cloud vendors presents other unique challenges that include the functionality of the software, workflow capabilities, and performance management. For example, how do I maintain a low click stream across all Cloud providers when they have unique taxonomies and page wire frames? They have different click stream architectures. How do I have a Workflow process that spans across multiple Cloud providers—especially true if the Cloud providers each have their own proprietary workflow tools? There needs to be a "master workflow" state engine that can be a master process flow across systems and leveraging their individual workflow capabilities. How do I manage performance or identify bottlenecks across multiple providers? Ping-ponging packets between on-premise systems, public cloud, and private clouds can cause inefficient data flows that cause long wait times. Ultimately, the end-users will suffer from cloud systems that doesn't account for moving data between corporate firewalls and cloud providers.

(6) Standards

Standards are important for enforcing consistency and simpler governance models. This is especially important when different technologies, approaches, and vendors are involved in a cloud system. It's important to rely on standards such as WSRP, SAML, W-3 webServices and more to simplify the integration and management amongst your portfolio of cloud providers.

Leading Practices:

While integrating with the cloud presents itself with unique set challenges that include: off-premise hosting, federated and distributed location of cloud providers, reliance on providers for standard and technology support, cross-firewall integration, and potential performance issues, there have evolved some early leading practices.

- Use a Service Oriented approach for integrating to and from the cloud
- Attach contracts to each service for management, monitoring, and governance
- Leverage standards wherever possible
- Limit the number of network hops when possible
- Integrate through webServices or remote API's
- Rely on native cloud provider Workflow tools, and implement cross-technology workflow when processes cross technologies

- Strive for consistent look and feel, click stream, and overall usability through webServices and standards