# SaaS and Integration Best Practices

● Henry Hai ● Seitaro Sakoda

*(Manuscript Received January 23, 2009)*

**The rising adoption of software-as-a-service (SaaS) applications by enterprise organizations has been driven by deep dissatisfaction with on-premise applications, which require organizations to purchase and deploy infrastructure, overstock on licenses, and pay for expensive resources for customizations, upgrades, and on-going maintenance. The large upfront investments combined with unpredictable costs and an immeasurable return on investment have prompted organizations to seek cheaper less-risky alternatives. Many have found that SaaS applications, which require minimal or no infrastructure and maintenance, can be deployed quickly and have a predictable cost model representing less risk and a faster return on investment. The new demand has led to rapid innovation in SaaS applications, SaaS platforms, third-party SaaS add-ons, and SaaS integration tools. However, enterprise organizations still have the burden of integrating these applications with their back-office systems and on-premise applications, without which the SaaS applications have little to no value. Complex enterprise integration requirements challenge even the best SaaS solution providers today; there are still limitations and pitfalls to be wary of. In this paper, we describe some SaaS integration best practices, present a case study, and highlight emerging integration technologies that can help ease the burden of integrating SaaS applications.**

## 1. Introduction

Today's businesses need to be agile. Constant change comes from every direction: mergers and acquisitions, re-organizations, new regulations, economic pressures, competitive pressures, and fluctuations in customer demand. Unfortunately, most applications are not designed for flexibility, and the unpredictable costs and resources needed to maintain the infrastructure and extend these applications are constraining the business instead of enabling it.

These problems are not new. The application service providers (ASPs) of the 1990s, which provided shrink-wrapped applications to business users over the Internet, made early attempts to offer robust packaged applications more quickly and easily. Unfortunately, they tended to offer few economic benefits over their locally installed counterparts and had limited ability for customization, data sharing, and integration with other applications. Customers were still required to pay for perpetual licensing, application upgrades, security, performance, availability, and unused capacity. Any customizations to the applications were costly and reserved only for larger customers. Customizations also required the ASPs to manage the development, testing, and deployment cycles differently for every customer.

Today, with advances in infrastructure and software technology, a new breed of application is being produced: robust Web-based applications designed as centralized, shared-instance, multi-tenant applications called software as a service

(SaaS). Like ASPs, up-and-coming SaaS solution providers take responsibility for managing the service including security, performance, availability, reliability, and scalability. However, the similarities stop there. In mature SaaS offerings, multi-tenancy applies to all tiers of the software architecture: all tenants share the same codebase and instances of the application, which enables large economies of scale. Given that the underlying application codebase cannot be changed for each tenant's customizations, the application is abstracted into loosely coupled fine-grained configurable components. This lets customers quickly change the presentation, logic, and database layers of the application through on-screen clicks without code modification, compilation, or deployment. Moreover, the loose coupling allows SaaS providers to release frequent application upgrades, giving all customers seamless access to the latest version of the service bundled with their unique customizations. The economies-of-scale savings are passed on to customers though a low-cost pay-as-you-go subscription model. This model also has the benefit of permitting customers to regard the service cost as an operating expense rather than a capital expense, which allows all the financial risk to be taken over an extended period of time instead of up front. Small and medium businesses, in particular, favor SaaS because of their lack of capital, infrastructure, and the resources typically required to run packaged applications on their premises (known as on-premise applications).

Enterprise organizations are beginning to adopt SaaS. Pockets of enterprise business users are purchasing independent SaaS applications without the consent of their overworked counterparts in the information technology (IT) department. They are taking on the responsibilities of customizing and administering the applications themselves. Organizations are also experiencing multiple instances of SaaS applications among their various departments with the IT department having little or no control over them and they need consolidation and integration. As SaaS solution providers gain strength and begin to offer better customization and integration capabilities, the business is beginning to call for IT department involvement. On the flip side, IT managers who are facing recent failures of large-scale on-premise packaged application implementations and tight budgets are beginning to recognize the low-risk and quick time-to-market advantages of the SaaS delivery model. This coalescence is producing some of the largest enterprise SaaS implementations to date.

SaaS is growing by leaps and bounds. Large independent software vendors that have traditionally offered on-premise applications are creating new SaaS offerings in response to market demand. Some of the larger traditional on-premise players offering SaaS applications are SAP in enterprise resource planning (ERP); Informatica in extract-transform-load (ETL) processes, data quality, and enterprise application integration; Business Objects in reporting and business intelligence; Oracle in customer relationship management (CRM); and Microsoft in business productivity tools and CRM. Many of these traditional vendors are migrating their on-premise offerings to a SaaS model, but they face the prospect of cannibalizing their existing market and the challenge of transitioning their legacy architectures to a multi-tenant model. Most are currently offering both while they build up their SaaS capabilities. Some of the pure SaaS players leading the market are Salesforce.com in CRM and application platforms, Netsuite in ERP, Google in business productivity tools and application platforms, and Workday in human capital management.

SaaS has driven a new implementation methodology. With no infrastructure to be purchased, set up, or maintained, a SaaS application is immediately available for customization. A declarative development model enables changes to be quickly designed,

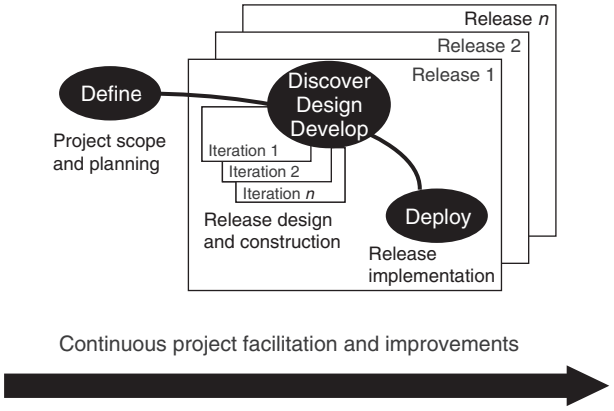


Figure 1
SaaS implementation stages.

implemented, and shown to users for feedback. Releases can be smaller and more frequent as a result of logical partitioning of functionalities and the decreased overhead for testing and deployment. SaaS has enabled a shift from traditional waterfall-like development models consisting of single-phase sequential steps to a more agile methodology consisting of multiple phases and multiple iterations within each phase. Iterative methodologies help maximize user adoption by incorporating user feedback early in the discovery, design, and development stages of the implementation. The SaaS Practice of Fujitsu Consulting has developed a five-stage methodology for SaaS implementation (define, discover, design, develop, and deploy), as shown in **Figure 1**. It is part of Fujitsu Consulting's Macroscope, which is an integrated suite of business and IT methods that help Fujitsu's clients gain maximum value from their business transformation initiatives.

## 2. Integration best practices

Integration is critical to the success of any application, especially within enterprise organizations. Its prerequisites are a bridge between SaaS applications and data warehouses, entitlement, authentication, billing systems, a system-of-record, and on-premise applications. With many on-premise applications, customers can integrate anywhere in the stack. In many cases, this integration is across layers in a manner not intended by the application's vendor but performed nevertheless because interfaces are unavailable, because organizations do not have the necessary skill set and do what they know how to do, or simply because it is faster and cheaper. Customers may also decide to purchase bolt-on or point solutions from third-party vendors that cannot be upgraded or supported over the long term. These reactions may be convenient in the short term, but they raise issues for both the application vendor and customer by creating brittle connections that are easily broken by software upgrades, additional integrations, or customizations applied to the application. The SaaS approach to integration leverages a set of standard Web service application programming interfaces (APIs) published by the SaaS solution provider. All data integration is executed though these APIs over the Internet, enabling SaaS solution providers to continuously provide upgrades to functionality without breaking existing integrations. Complex enterprise integration requirements challenge even the best SaaS applications today: there are still limitations and pitfalls that organizations must be wary of. The cost of integration can be 30–45% of the overall SaaS implementation, which often increases project durations. While organizations will encounter some new integration challenges with SaaS, they are still faced with many of the same challenges of traditional application integration.

### 2.1 Upfront considerations

Enterprise organizations in industries such as financial services, health care, and life sciences are subject to extensive government, industry, and internal IT data security regulations that are more challenging to address with SaaS solutions but must be considered early in the process. For example, some government data privacy laws ban any information about its

citizens deemed confidential from being stored outside the country. If the SaaS application solution provider's data center is located outside the country, then it cannot be used. In the financial services, industry organizations are required to provide detailed audit trails of all correspondence with customers and prospective customers. These audit trails are typically required to be archived by the organization for a certain period of time and in such a manner that they cannot be tampered with. If an organization's E-mails are sent through a SaaS solution provider, the organization may not have records of these messages on their internal systems and may have little or no control of the archival process or access control for them. We must also not forget organizational IT data security policies, which may dictate what types of data cannot be replicated outside the organization's four walls, different levels of audit trail that must be maintained for changes to that data, or data that must always be encrypted when stored. Lastly, there are the business requirements of the application that may impose complex data visibility rules that are unique to the organization's business process and cannot be changed. SaaS applications that have rigid visibility models may require additional work to shoe-horn these rules into the application, which often compromises the integrity of the data or the application.

The ease of use of Web service integration available in most development tools may entice organizations to build their own point-to-point integration, but unless a common data integration framework is used, this is not recommended. Most enterprise organizations will have some internal standard application for data integration. Many of the larger traditional on-premise integration vendors such as IBM DataStage, Informatica, and Ab Initio already have Web service capability and are also providing specialized connectors for SaaS applications. The connectors can significantly reduce the integration effort while taking advantage of any specialized functionality offered by the SaaS solution provider. In some cases, these connectors cannot be used because of incompatible integration software versions, so some of these integration vendors offer stopgap measures through standalone appliances or subscription-based pricing until the organization has caught up and upgraded to the appropriate version. Some SaaS solution providers provide integration as part of their service offerings by providing an integration server with prebuilt connectors to common on-premise applications, which further reduces the integration effort and cost. Lastly, there are new emerging offerings dubbed integration as a service, which are analogous to SaaS applications in integration space. They can be used for Cloud-to-Cloud, Cloud-to-on-premise, or on-premise-to-on-premise integrations. We discuss integration as a service in more detail in the next section.

## 2.2 Integration architecture and design

Once the fundamental integration requirements have been established, the process of designing the integration can begin. Given that SaaS application integrations typically occur over the Internet, the integration architecture must consider the locations of the different on-premise source and target systems within the organization's network. Understanding the locations, connections, and protocols between these components and the other systems that must be traversed for Internet traffic will offer immediate insight into interoperability, security, scalability, and performance concerns. Furthermore, defining performance metrics for each integration upfront is paramount to the success of the project and can help define how the integration is designed and implemented. In the world of SaaS integration, Web service integration and mashups are common, but they introduce factors that an organization has little control over such as the Internet and the different service levels of the SaaS solution

providers. Initiating a proof of concept early in the integration discovery period or even as early as the sales cycle will yield valuable data about the approach and performance when combined with measurable success metrics.

There are a few integration patterns that are commonly applied to SaaS implementations. The functionality offered by SaaS applications can be leveraged only if the data to be acted upon is stored within the SaaS application's data tier. This is fundamental to most SaaS applications and often leads to data replication and synchronization between the on-premise system of record and the SaaS application. A clear understanding of the desired functionality will yield the minimum data elements requiring synchronization and replication. Most SaaS solution providers offer APIs that can facilitate batch synchronization of data through simple Create, Read, Update, and Delete (known as CRUD) statements. More sophisticated SaaS solution providers offer event-driven near-realtime data synchronization capabilities.

Mashups are another common integration pattern used to support information sharing, information enrichment, and collaboration in SaaS applications. In particular, mashups are used to enrich SaaS application data with realtime back-office data presented in concert seamlessly to the user. This is an alternative to the approach of replicating back-office data in the SaaS application. It is essential to match the integration capabilities of a SaaS solution provider with the integration requirements and with the pros and cons of each approach.

## 2.3 Working with large volumes of data

There are often cases in which enterprise organizations have large volumes of data required in the SaaS application. It is important to work with the SaaS solution provider early in the process to understand any potential ramifications. Foresight into initial sizing and growth of the SaaS application data is necessary to ensure that the organization will not exceed its data storage limits and incur additional costs. Some SaaS solution providers offer data archiving services that can reduce the storage overhead but may change the functionality of the application. Organizations should check that the volume of data will not detrimentally affect application performance, stability, and functionality. Since all data integrations for the SaaS application occur through the API and many SaaS solution providers impose rate limits on the number of API calls that an organization can perform within a certain period of time, it may be necessary for the SaaS solution provider to raise those limits. Some SaaS solution providers support different modes of operation for large organizations and have enhanced methods for expediting the load of large data volumes. This can be critical if an organization has a small time window for execution because of dependent integration jobs or global time constraints.

## 2.4 Extension of service oriented architecture governance

As organizations create more business functionality and integrations with SaaS applications they must also extend and adapt their service oriented architecture (SOA) governance practices accordingly. SaaS solution providers are continually upgrading their capabilities and creating more sophisticated APIs. Some even offer organizations the ability to create their own custom business services. Governance must be extended to cover matters such as ensuring that the SaaS solution provider maintains backward compatibility in their APIs and continues to support any custom-built business services through change management processes and testing; to validate compliance with IT standards, especially around auditing and security; and to monitor and track the quality of the SaaS services to make certain that integrations do not suffer faults or failures. There are a number of different tools available in the SOA

industry, and there are many emerging ones for SaaS applications that can help facilitate SOA governance.

## 3. Integration on demand

As more organizations adopt SaaS applications with greater interoperability, an emerging integration-on-demand market has been created, which calls for an integration service that provides all the benefits and architecture of SaaS. These services allow non-technical end users to customize and deploy integrations through menu-driven wizards configuring data sources and targets, mappings, transformations, integration processes, and the scheduling of integration jobs. The service provides a natural model for Cloud-to-Cloud integration, a simplified model for on-premise-to-Cloud integration, and a conventional model for on-premise-to-on-premise integration. It also presents a platform for easily incorporating new connectors and value-added services, such as data cleansing and validation, into organizations for a significantly reduced cost compared with on-premise products in a pay-as-you-go fashion. The value proposition for integration as a service is high for small and medium businesses that lack integration capabilities, resources, and tools.

These services come with a browser-based integration design and with management tools having pre-built connectors to SaaS applications, databases, and on-premise applications. They offer services such as data replication, data synchronization, data quality assessment, and custom integration. For some on-premise integrations, an agent must be deployed within an organization's infrastructure with access to the source or target systems. These agents are self-contained integration servers that possess the components required to execute integration processes from end to end. The agent receives its instructions from the integration-as-a-service solution provider on the basis of the organization's integration processes as well as any necessary connectors or software updates. The agent provides regular reports on the health and integration status to the integration-as-a-service solution provider, which can then be monitored and managed by the organization. However, the integration itself is communicated only among the on-premise agent, the source on-premise system, and either a target SaaS application or another target on-premise system. Some solution providers offer virtual agents to organizations that do not wish to deploy any infrastructure but require access to on-premise systems for on-premise-to-Cloud integrations. Some of the different integration scenarios are illustrated in **Figure 2**.

The pricing model for integration-on-demand offerings is much more flexible and affordable than that for their on-premise counterparts. Organizations are generally charged a flat monthly fee for the connectors that they require regardless of the usage, data volumes, or deployed agents. With some solution providers, this flat fee also covers unlimited use of all connectors, which makes the cost much more predictable and significantly reduces the deployment risk.

In enterprise space, we can expect organizations to continue to leverage their corporate integration tools. Some will maintain a hybrid model of an existing on-premise integration system combined with an integration service because it can offer a broader spectrum of services and connectors more quickly and cheaply than their on-premise counterparts. On the other hand, the on-premise integration systems will continue to offer complex integration capabilities that their on-demand counterparts cannot provide. Moreover, as with the adoption pattern of SaaS applications in enterprises, savvy business users are using these on-demand integration tools sometimes below the radar of the IT department, which raises significant security concerns about unauthorized client applications accessing corporate data sources. Eventually, as these tools become more sophisticated and

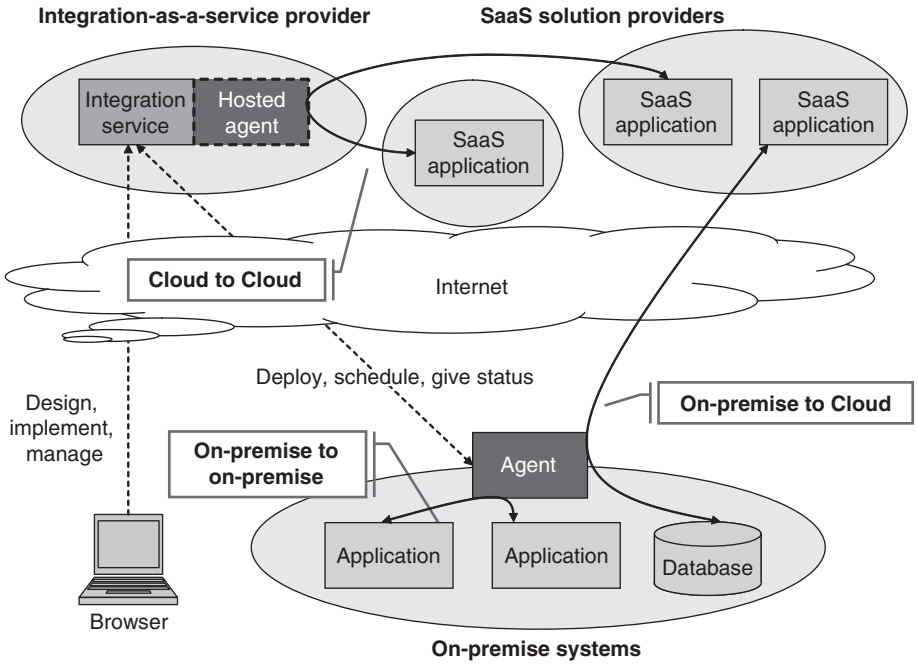


Figure 2
Integration scenarios.

organizations move more applications to the Cloud, SaaS application solution providers will begin to incorporate these services into their own offerings and supply out-of-the-box integration between different SaaS applications at the click of a button.

## 4. Enterprise integration case study

A large retail banking firm replaced multiple on-premise CRM applications with a single SaaS CRM package. There were several different reasons that drove the firm to replace their existing CRM applications. Some of the applications had been so severely customized that they could no longer be easily supported or upgraded to take advantage of new releases. These applications were from different vendors and had been implemented by different departments, which created partitioned IT knowledge, business processes, rules, and data. A merger with another large financial organization drove the need to share client information easily across different business units. Finally, there was an organization-wide initiative to leverage a newly created enterprise data warehouse to drive a singular view of the client across all applications.

The goals for using the new CRM application were to improve relationship management, streamline the creation and distribution of opportunities, enable the prospecting of new business, increase the sharing of information among business units, and permit a 360° view of a customer by any sales representative. At the same time, the firm did not want to repeat past mistakes related to customization and costly maintenance. The benefits of the SaaS model and the robust integration capabilities of the CRM package played a key factor in the successful realization of the stated goals.

The integration with the data warehouse required batch synchronization of visibility, client, and financial account data, which consisted of millions of records. Dynamic data such as account balances was available on demand through a realtime interface to the data warehouse from the CRM application. Opportunities and alerts

were fed to the CRM system in real time and then routed to the appropriate sales representative in the nearest retail branch for up-selling and cross-selling opportunities. Sales representatives were able to generate prospecting lists by querying their data warehouse in real time through the CRM application, and these lists were then brought into the CRM system so that the representatives could continue to manage them. The sheer volume of data and complexity of these integrations required the use of an on-premise ETL tool capable of pushing millions of records to the SaaS CRM provider over the Internet, which was probably the most challenging element of the project.

Overall, the project was a success and the firm was able to accomplish its business goals through the SaaS CRM package and it significantly reduced its maintenance costs. However, the complexity of the integrations required contributions from many different teams, required an open channel of communications with the SaaS solution provider, and involved development efforts, tools, and challenges comparable to traditional application integration.

## 5. Conclusion

Although SaaS has dramatically transformed the application lifecycle, it has not significantly reduced the complexities of integration with SaaS applications. Organizations are still faced with vendor-specific APIs and varying levels of integration capabilities from different SaaS solution providers. Since most enterprise organizations require integrations to the back office, many of the same traditional integration challenges are encountered. Integration-as-a-service solutions are beginning to simplify integrations, especially in the Cloud-to-Cloud space, but they do not yet offer the same level of capabilities as their on-premise counterparts. SaaS solution providers are starting to focus more on pre-built integrations, which will help reduce the cost and complexity of integration. However, system integrators will continue to play a key role in preserving the SaaS value proposition while creating durable value-added integrations.

**Henry Hai**
*Fujitsu America Inc.*
Mr. Hai received a B.S. degree in Computer Science and Philosophy from Rensselaer Polytechnic Institute, New York, USA in 1996. He joined the SaaS Practice of Fujitsu Consulting Inc. in October 2003 and has been engaged in implementing and integrating SaaS applications for enterprise customers. He has been with Fujitsu America Inc. since its merger in April 2009.

E-mail: henry.hai@us.fujitsu.com

**Seitaro Sakoda**
*Fujitsu America Inc.*
Mr. Sakoda received a B.S. degree in Physics from Kyoto University, Kyoto, Japan in 1998 and an M.S. degree in Physics from the University of Tokyo, Tokyo, Japan in 2000. He joined Fujitsu Ltd., Tokyo, Japan in 2001 and engaged in system development and technology support. He moved to Fujitsu Consulting Inc., New Jersey, USA in 2004 and has been engaged in technical consulting and system development. He then joined the SaaS Practice of Fujitsu Consulting as a technical leader. He has been with Fujitsu America Inc. since its merger in April 2009.

E-mail: seitaro.sakoda@us.fujitsu.com