Integrating with the Cloud – Practices That Work

Glen T. Ryen

Prisio Technologies

Abstract

With business applications moving to the cloud, organizations are increasingly running their applications in an on-premise/cloud hybrid mode. This session covers how cloud applications (or SaaS, Software as a Service) can co-exist with on-premise applications and the practices used to integrate the applications. Integration options using presentation layer, application, and data integration technologies are discussed. Some integration best practices will be highlighted.

Introduction

The continued growth in the SaaS/cloud applications market comes with many new benefits to adopting organizations, but also new difficulties. Companies face many unique challenges in trying to integrate cloud applications with their on-premise applications, to extract maximum business benefit from both. This paper aims to highlight some of those challenges and discuss some high-level integration options, followed by listing some integration best practices specifically relevant to cloud applications.

This paper will be comprised of the following sections:

- **SaaS Applications – "The cloud"**
  - Definition, Features, and Benefits
  - Integration Challenges
- **Cloud Integration Options**
  - Presentation Layer Integrations
  - Application Integrations
  - Data Integrations
- **Best Practices**

This paper does not aim to provide detailed, step-by-step instructions for any one particular area or topic, or excessive details of implementation technology, architecture, or design patterns. It instead attempts to provide applications managers or business analysts with high level overviews that summarize the important aspects of each topic, with pointers to relevant resources for the reader to follow up on as applicable.

Much of the discussion and examples will assume organizations are existing customers of Oracle's E-Business Suite ERP (Enterprise Resource Planning) application and/or database products, and will consider those the on-premise applications that readers will be looking to integrate to any of a myriad of cloud-hosted applications. These could be:

- CRM (Customer Relationship Management) applications, like Saleforce.com
- HCM (Human Capital Management) services, like Workday
- Talent Management applications, like Taleo
- Other ERP/Financial systems, like NetSuite or Concur

But the discussion is equally relevant to any other custom or packaged on-premise application, or if your back-end Oracle ERP is hosted by a third-party outside of your organization's network.

SaaS Applications – "The Cloud"

To discuss the challenges and implementation options specific to integration between cloud and on-premise applications, it helps to begin with a common frame of reference. Specifically, what features or characteristics make an application a cloud application (or SaaS)?

Definition, Features and Benefits

For the purposes of this discussion, the following are the key characteristics of a cloud application:

- It is externally hosted, outside your organization's network.
- The application provider manages the availability, performance, scalability and security of the service. It is not managed or controlled by your internal IT department.
- The application is offered on a pay-as-you-go subscription model.
- All subscribers to the software run in a shared instance, also known as a multi-tenant architecture.

The benefits of the cloud-based delivery model can be compelling, as evidenced by the continued rise in its popularity in the enterprise application space. Having a single code base for all of its customers allows the SaaS provider to enjoy economies of scale. Testing only needs to be performed for one configuration, while traditional on-premise application vendors need to support or certify their products on a variety of different configurations – whatever combinations of operating systems, database versions, and application servers their client may be using. This allows SaaS providers to generally deliver much more frequent application upgrades.

Since customization of that shared code base is not an option for all of the tenants, SaaS providers generally offer their software as loosely-coupled configurable components. This allows for some degree of personalization of the data, business logic, and presentation layers.

Larger SaaS providers also generally offer pre-built connectors for the most popular ERP and/or middleware vendors (like Oracle), to facilitate integration. SaaS providers may also offer frameworks for integration and customization development. This allows them to actively develop a partner ecosystem or marketplace (like Salesforce.com's AppExchange) where the SaaS provider allows and encourages independent third-party developers to build and market their own integration solutions.

Integration Challenges

While the defining characteristics of a cloud application listed above create many benefits for adopting enterprises, those same benefits also create challenges. In addition to the usual complexity inherit in most enterprise application integration projects, a hybrid cloud and on-premise model creates specific challenges that aren't encountered when all the software is hosted and controlled in-house.

Crossing through firewalls limits the communications protocols that can be employed. Data is moved at internet speeds instead of internal network speeds, which means reduced throughput (important for the larger data volumes that may be hosted and processed by the cloud application) and increased latency (slowing response times for end user actions that need to involve both applications). Data volumes that are part of your subscription package may be capped, as may be the frequency at which you're allowed to call a given API (Application Programming Interface) or set of API's that you need to create, read, update, or delete that data. If your integration design causes you to exceed either or both of your negotiated limits, you'll be incurring additional charges.

In addition, that shared instance/multi-tenancy architecture that creates so many advantages for the cloud delivery model also creates disadvantages compared to on-premise, IT-controlled software. Having a single code base leads to API's constantly changing, often causing compatibility issues in integration code that you don't encounter when your organization controls the testing and release cycle.

The location and ownership of data could also be problematic, particularly for organizations subject to strict government or industry regulations – like those involved in financial services, health care, or life sciences. Data privacy laws may dictate that certain information can't be stored outside your country(ies) of jurisdiction, or your SaaS provider may just not have audit trail creation and retention capabilities that meet your organization's requirements – requirement either dictated to your company via legislation or mandated by your own IT department.

And while most SaaS providers offer integration tools and/or services, they will be of varying degrees of maturity. Pre-built connector may not be available for your on-premise system, or (in the case of packaged applications) they may not be compatible with your version of that system. The available connectors may not fit with your existing integration architecture, becoming just another place for your IT depart to have to check for critical interface run-time information – like access governance, transaction monitoring, exception handling, and metrics for SLA (Service Level Agreement) compliance (such as service reliability, availability, and performance).

You'll also find varying levels of capability in the data services offered by various SaaS providers. You need to evaluate upfront if the offerings from your cloud application provider meet your requirements for data replication/synchronization, granularity of data access control, and data cleansing/validation (handling inaccuracies and inconsistencies, viewing data lineage, etc.).

Cloud Integration Options

With those definitions in mind and challenges in mind, here are three high-level options that you can consider when designing and implementing your integration between your on-premise and cloud-based applications:

- Presentation Layer Integrations
- Application Integrations
- Data Integrations

And keep in mind that these are not mutually exclusive. Your specific integration needs may dictate a combination of integration approaches, or it may evolve over time.

Presentation Layer Integrations

Generally the simplest of the three implementation approaches, presentation layer integration is also very popular for several good reasons. This approach will require the least amount of effort and expertise to deliver some quick functionality wins to the end users. Often referred to as mashups or composites, presentation layer integration encompasses client presentation mashups and service mashups.

Client presentation mashups start out as integration at just at the visual level. They allow users to view some real-time information from two or more applications in a single browser window. A simple example could be a list of the most recent open orders for a customer, pulled from an on-premise order management system and displayed alongside the customer account information in a cloud-hosted CRM system. The data wouldn't be editable, but could be important information to the user making a decision or taking an action.

Slightly more complex would be service mashups, where the presentation or business logic layer can be configured to consume web sites or feeds. These mashups may also invoke an API to complete a business process by kicking off functionality in another application. The applications aren't integrated at a data level, but if the information required for completing the action is all available in the source application (or can be mapped or derived), then that information can be sent to the target system with the API request.

This type of lightweight integration can be accomplished relatively quickly. They may be as simple as some HTML or JavaScript code, or slightly more complex pre-built or custom API calls that are built (by your internal IT department or a third-party provider) according to your SaaS provider's mashup/composite development framework.

While you can improve a lot of areas of a business process that spans on-premise and cloud applications with this approach, you are limited in the amount process or workflow changes that you can affect. In addition, having separate data silos limits your ability to do analysis on the combined information.

Application Integrations

Where your business requirements dictate support for multi-step business processes, where functionality spans on-premise and cloud applications, the second option provides for that – application integrations. This approach provides significantly more capability over presentation layer integrations, at the cost of more complexity (often significantly so).

Application integration is characterized by an event-driven architecture and/or real-time/near real-time connectivity. Business logic and data are shared between the cloud and on-premise systems, such that an action or event in one system triggers the next step in the workflow, carried out in a second system. The communication may be one-way or bi-directional. An opportunity tracked in your SaaS CRM system and updated to a "Won" status could trigger the creation of a Sales Order or Invoice in your ERP system. And the resulting Order or Invoice number may be communicated back to the CRM system right away (synchronously) or at a later point (asynchronously, perhaps via a nightly batch process sending new Order or Invoice information for the day back from the ERP system to the CRM application).

This style of integration generally involves some form of middleware or an ESB (Enterprise Service Bus) handling communications. It is often facilitated by pre-built connectors or connector development frameworks from your SaaS or middleware/ESB provider. From a technical standpoint, the messages will be sent according to an agreed standard – like Web Service invocations in SOAP (Simple Object Access Protocol) envelopes, REST (REpresentational State Transfer) API calls over JMS (Java Messaging Service), etc.

For Oracle EBS customers, this is where Oracle Fusion Middleware products could come into play. The Oracle Fusion middleware suite include an ESB for Event-Driven Architecture, the SOA (Service Oriented Architecture) Suite for full-life cycle management of web services, the BPEL (Business Process Execution Language) Process Manager for process orchestration, and other tools. Or your organization may already have invested in another established middleware product, like TIBCO, webMethods, IBM Websphere, or Microsoft Biztalk.

Data Integrations

Compared to presentation layer and application integrations, data integrations occupy more of the middle ground on the cost vs. complexity spectrum. They can be employed effectively on their own to replicate data between your SaaS and on-premise applications, they can be augmented by presentation layer integration, or they can support or enable more complex application integration.

Data integration usually involves bulk or batch data loads, for initial migrations of historical data (when deploying SaaS functionality) and/or ongoing on a fixed schedule (hourly, nightly, or as often as the business requirements dictate). This can be accomplished with ETL tools (Extract, Transform, and Load) born out of the data warehousing software market, if your organization already uses those tools. Oracle Data Integrator is one such product in that market space, as are Informatica's PowerCenter and PowerExchange, Ab Initio, and IBM's InfoSphere DataStage.

Or for more real-time synchronization, replication engines that employ CDC (Change Data Capture) technology could be used to propagate on-premise updates to/from your SaaS provider (if they support it). This type of "trickle-feed" software tracks insert, update, and delete operations in a source database by directly accessing the database's transaction (redo) logs, forwarding that new and changed data on to other systems for replication. Oracle GoldenGate is one example of this type of software.

With data integrations, data services can (and often should) be employed. These services can be simple abstraction layers that a developer codes to, to loosen the coupling between two data sources. Separating the data access from the implementation details (the protocols, file formats, and metadata) makes it easier for you to adapt when some of

those implementation details change (like they almost certainly would, if you change SaaS providers at some point). These services could be implemented as web services over HTTP or JMS.

Data services could also involve calls to de-duplication and cleansing tools, also available via a SaaS delivery model (like Data.com). Or you may consider acquiring on-premise cleansing tools to programmatically evaluate your data, tools like ActivePrime CleanCRM or many others. However you tackle the problem, data cleansing is not just good business practice in general. It's essential for many data-intensive integration projects, CRM in particular. Your organization needs to focus on correcting or deleting any incorrect, incomplete, duplicate or improperly formatted data. And that is an on-going effort, not a one-time project.

Best Practices

When integrating applications in a hybrid on-premise/cloud environment, you need to focus first and foremost on the business process(es) you want to enable. When operating or viewing data in a single system, how current must that information be to be valuable to the process? What are all the applicable data access, retention, and auditing requirements that the process will need to adhere to? The requirements of the process should drive the selection of the software tools and techniques employed, not vice versa. Too often in integration, the tools readily in hand are employed without first questioning whether or not they are the best fit for solving the problem.

Second, be sure to spend the time necessary in the discovery process to really understand your data. Your organization may have many disparate data sources that you're looking to integrate, and even packaged applications like Oracle, SAP, etc. are likely to have been customized over the years in your environment. The more complex and numerous your data sources, the more you should also consider automated data discovery tools like Queplix, Eloqua, and others. The information gathered could build into your data dictionary, or you may already have MDM (Master Data Management) tools that you will leverage for this purpose.

However you decide to do it, be sure to allow enough time to focus on the on-going data quality in your integration – cataloging, deciding on which applications should be the system of record for what information, etc. And don't overlook the need to retain an acceptable level of control over your data, both for regulatory/internal IT policy compliance as well as business continuity purposes. Ensure you're protected if you later on decide to switch SaaS providers, or even if your provider goes out of business.

Third, spend some time defining your performance metrics. Document upfront the acceptable response times for user actions (where you're planning presentation layer or application layer integrations) and the operating windows for scheduled batch processes (for larger batch data integrations). This is very important, as setting those parameters will shape exactly how you design and build your integrations.

Once you've taken those steps, then you're at a point where you can properly seek out and evaluate the integration tools that may be at your disposal. Look first to buy vs. build, researching any available connectors from your SaaS provider (like Force.com Connect for Oracle), from your on-premise/ERP provider (like Knowledge Modules for Oracle Data Integrator), or from your preferred on-site integration vendor (Informatica, TIBCO, etc.).

For firms that don't have an invested and established relationship with an on-site integration vendor, you can consider leveraging your initial on-premise/cloud integration as a move to IaaS (Integration as a Service). IaaS is a market that seeks to supplant on-premise integration tools with cloud-based integration platforms, offering many of same characteristics and benefits discussed previously for general SaaS applications. Small and medium sized firms with limited integration capabilities could find the value proposition of IaaS compelling, particularly if they're already considering SaaS delivery for some other business functionality. Dell Boomi, Informatica Cloud, and IBM Cast Iron are some examples of players in this growing market space.

When designing your integration, be sure to understand and follow SOA (Service Oriented Architecture) principles. Resist the temptation to code point-to-point interfaces and keep the coupling between individual systems loose. Basic SOA principles include techniques like coding to abstraction layers, to hide implementation details for services and protect your calling application from changes in those implementation details. It also means aggregating services where it makes sense. For instance, you can take several very granular API calls that may be

offered by your SaaS provider and wrap them in a single BPEL process. This can turn a very "chatty" conversation between applications into a single coarse web service that will be easier for your application developers to invoke. All of this increases your integration's ability to withstand (or quickly react to) inevitable changes.

And finally, be sure that your organization has some form of an integration team or competency center. This should involve a small group of employees – IT, functional business users, and management. The IT representatives should have the requisite technical skills to evaluate the data tools and services you may use, and then to understand the workings of the tools once they're acquired. The business users should have the knowledge to specify the initial requirements, all the way through to reviewing the integrated data for correctness. And management, in addition being able to drive decisions and keep the integration project(s) on track, is very often the ultimate consumer of the integrated data. You want to make sure all those skills are adequately represented today, as well as down the line as those employees change jobs or positions.

About the Author

Glen T. Ryen is the Vice President of Technology at Prisio Technologies. He has over 16 years of experience in integrating, implementing, and customizing the Oracle E-Business Suite. He has experience with ETL, middleware, and SOA-based integrations. He has helped numerous Fortune 500 customers in various industries with their Oracle Applications needs, in technical, functional, and management capacities. He can be reached at GlenR@prisiotechnologies.com.

Abbreviations Used

API	Application Programming Interface
BPEL	Business Process Execution Language
CDC	Change Data Capture
CRM	Customer Relationship Management
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
ETL	Extract, Transform, and Load
HCM	Human Capital Management
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JMS	Java Messaging Service
IaaS	Integration as a Service
REST	REpresentational State Transfer
SaaS	Software as a Service
SLA	Service Level Agreement
SOA	Service Oriented Architecture
XML	Extensible Markup Language

