Print Article Close Window

From: www.cio.com

Cloud App Integration: What's the Best Path?

– David Taber, CIO

January 14, 2011

There's a lot of noise from vendors of every stripe about the cloud. Unfortunately, in the vendors' efforts to show how all their products are cloud-based, there's a lot of blurring about the specifics of what it means to be a cloud application. Consequently, this article will apply differently to every cloud vendor. (And for the purposes of this article, let's keep the discussion to SaaS and cloud-based apps from a vendor or integrator, not ones you build yourself, although some of the same principles apply.)

Cloud Computing: 2011 Predictions

With that disclaimer behind us, one of the distinguishing characteristics of cloud software is the variety of ways it can be integrated. As most cloud applications present themselves as a series of Web services, they lend themselves to a service-oriented architecture, even if they don't follow all the SOA protocols. With the right toolkits and development attitude, you can integrate cloud applications with a variety of techniques...and use as many of them concurrently as you like, even in the same application. Of course, you have to understand the limitations of each approach—but there's nothing wrong with getting things done quickly. Let's look at this as layers of an onion.

Layer 1: On-Screen Integration

Otherwise known as mashups, this style of integration is the ultimate in quick and dirty. The coding exercise is the construction of iFrames for the screen layout and URLs with lots of parameters for grabbing the goodies from the other cloud. This is the baseline method for pulling images, maps, news items, and data feeds from publicly available services like Google (GOOG) or Yahoo (YHOO). This method will become increasingly powerful (particularly for demos) as graphing packages and other document services become commonplace as cloud services. AJAX can give the pages a modern, intuitive, and responsive UI. Unfortunately, mashups don't inherently offer much in the way of security, so you'll have to look at tricky coding practices and server-side validation (for example, here) for sensitive data, and you'll probably want single sign-on or other authorization infrastructure to control access without irritating users. So the tradeoff at this layer is: simple code and read-only, or secured with complex code.

Layer 2: Presentation Layer Integration

Depending on the way your cloud application generates Web pages, you may have a programming layer on the server side which provides fertile ground for cloud integration. (In contrast, the mashup strategy works almost entirely in the browser.) While the mashup strategy is great for stitching together entire segments of a page (e.g., adding a map or graphic to a layout), integrating at the presentation layer shines in its ability to add individual fields within a section of a page. For example, it would be nice to add an indication of "how many days overdue is a customer payment" to the summary area of the CRM account page, but this field might only be available in your accounting system. Pulling this in at the presentation layer gives the users what they need to see, and is faster than doing a full-blown integration.

Of course, the strength of this approach is also its weakness: that payment overdue indicator would not be stored anywhere in the CRM system, so it wouldn't be available to support reports, alerts, or other functions. This approach is usually used for read-only data, as the presentation layer may not have the kind of security infrastructure available in the rest of the system. It all depends on the language you're using and the Web service security libraries available — but it usually doesn't make sense to attempt complex security mechanisms when integrating at the presentation layer.

Layer 3: Business Logic Integration

This is where the heavy lifting of integration gets done, because this is where the application context is kept and where the best security and Web services infrastructure is available. What really sets cloud applications apart is the richness and ease of their APIs: do they support call out and call in, WSDL/SOAP, RESTful APIs, or only simpler conversations with XML, JSON, or similar vocabularies? For productivity, there's no substitute for accurate documentation and code samples...so evaluate cloud vendors on this basis.

Most cloud applications' integration architecture is quite loosely coupled and based on a request/response model. Frequent polling is rarely a good idea, and tight integration loops (like two-phase commit) are tough. In situations where a cloud must push a message, your developers will have to create logic within that application to trigger sending the message. Your developers will also need to develop a strategy (perhaps using a dedicated integration server) to handle network timeouts, application downtime, and guaranteed message delivery.

At this layer, integration code will have access to all system objects and functions, so security will be essential. But it's such a big topic I'll be dealing with that in a separate article.

[For complete coverage of the Cloud Apps Wars -- including a complete guide to the business war, the competing products including Google Docs and Office 2010, the implications for users and IT, and more -- see CIO.com's Cloud Apps Wars Bible.]

Layer 4: Data Integration

This is dealing directly with the cloud application's database. In many cloud systems, there is no real way to directly access this level because it's really not safe for writing. Even for read integration direct database access can be problematic, as the table has no indication of application state or transaction coordination. That said, for bulk reading of data (for example, to replicate it for an on-prem data warehouse or a cloud-based analytics tool), nothing beats the speed of direct database access.

At this layer, security is an issue because the application's security model transcends what's visible in the tables' access controls. In most cases, data integration will be done with super-user privileges so the resulting data flow should not be directly accessible to standard users.

Next week, we'll look at another dimension of integrating across clouds.

David Taber is the author of the new Prentice Hall book, "Salesforce.com Secrets of Success" and is the CEO of SalesLogistix, a certified Salesforce.com consultancy focused on business process improvement through use of CRM systems. SalesLogistix clients are in North America, Europe, Israel, and India, and David has over 25 years experience in high tech, including 10 years at the VP level or above.

Follow everything from CIO.com on Twitter @CIOonline.