From: www.cio.com

Advice for Evaluating CRM Cloud Platforms

– David Taber, CIO

April 30, 2010

A day doesn't go by without headlines about cloud computing, virtualization, and the next computing platform. No doubt these computing models are important, but when it comes to CRM — what's important about cloud computing? And how should platforms be evaluated for CRM applications?

As I've written previously, in CRM applications the ability to easily modify, extend, and integrate the application is more important than any particular feature. Because the ultimate CRM system is built, not bought, platform strength trumps feature-list. So anything that makes the platform better makes the CRM more likely to be robust and durable. (That said, if you have a perfect platform and the users hate the UI, the CRM system will be stillborn.)

VMForce, Explained: A Faster Path to Apps in the Cloud

When evaluating platforms, however, it's important to avoid being stuck in the muck of acronyms and jargon. What really matters about a platform in the context of CRM?

- **The popularity of its APIs, languages, and libraries.** A platform with an implied developer population of a million is way more important than one with a small band of developer zealots. Remember ABAP? It doesn't matter if Ruby on Rails is beautiful if you can't assemble a productive team in where it's needed. A pragmatic focus on developer populations, rather than underlying technology, favors VB.net, Java, PHP, PERL, and Python.
- **The number and usage of add-ins, tools, and development aids.** It really matters whether the libraries have been vetted by a large group of (probably open-source) users. Will they properly handle UTF-32 characters? Too often, not so much. You're also looking for IDE plug-ins and extensions (Eclipse or maybe NetBeans), test harnesses, and build environments.
- **Is the platform well thought-out and coherent way to integrate, or is it a bunch of marketing hooey?** I spent too many years of my life marketing baloney to developers: Don't get taken. There's two reasons for a cloud platform: ease of development and reach/scope of integration with other applications. Look at the APIs for consistency and scope. Can you get at all the important objects in the CRM system? Are the APIs strewn across 17 dlls, or are they a logical set of Web services? Are all APIs available from any subsystem, or is the platform partitioned? Can an application start a CRM transaction or workflow from outside the platform, and can an outside system fully participate in the CRM application's triggers and workflows?
- **Real-world scalability.** How many hours of downtime happen per month? What's the response time look like in busy hours? Does the platform have governor limits or force Byzantine code structures, just for the convenience of the vendors? Or are there straightforward ways to handle 10,000 or 100,000 records in one shot? You'd be surprised how many scalability limitations are embedded in CRM APIs.
- **A fine-grained security model that is enforced for all API actions.** There are three levels of security models to think about: the underlying database C/R/U/D privileges; the application level roles, objects, and actions; and the web services' methods.

As this list shows, the devil is in the details. Because it's possible, for instance, to have APIs based on a very popular language but have a CRM vendor add proprietary extensions that dwarf the effective size of the developer community. Or to have APIs that are perfect, but only usable on the data stored in the CRM application's database.

Unfortunately, I know of no industry analyst with the hands-on knowledge of any CRM application or cloud-computing platform to provide meaningful advice. So the only way to really know if your CRM platform is solid and flexible enough for you is to do a pilot project with a realistic application. Sure, this can get pricey, but it's cheap in comparison to choosing the wrong platform.

Does the VMware-Salesforce (VMW) platform make each of their offerings better? It certainly has the potential to. VMware gets a boost to its developer community and very popular set of business objects and application functionality. Salesforce gets more of the Java developer community and the benchmark for virtualized computing, extending its platform with a scalable general purpose cloud stack. While not directly targeted at Microsoft's (MSFT) Azure platform, VMforce gives CIOs a serious contender for creating a range of business apps entirely within the cloud.

So what's the dark side of these clouds? Some types of app may really be better on your own machines, and you'll have to think hard about how to move from a public cloud (nearly always the right answer for prototyping and pilots) to a private one (which may make sense for your official corporate deployments).

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.