



From: [www.cio.com](http://www.cio.com)

## CRM Success Sealed with a KISSS

– David Taber, CIO

May 28, 2010

When Bill Raduchel was CIO of Sun Microsystems ([JAVA](#)), he espoused KISSS (Keep It Small, Simple, and Separable), the extended version of the KISS principle. IT success wasn't just a matter of keeping things simple, it was making sure that projects were as small and separable as possible. This was years before the Agile Manifesto, but the logic — both technical and user-centered — was built on the same foundation. The [Mythical Man-Month](#) is still in print for a reason.

As I've [written before](#), it is essential to avoid the big-bang style of project deliveries for CRM projects. You'll find that nearly every CRM consultancy will claim that they're an Agile shop. This is natural, as nearly all the authors of the Agile manifesto were themselves consultants. But it's one thing to blurt Agile all over one's Web site, and another to actually run projects that way.

Why do you care? Because CRM projects need to be much more flexible and adaptive than general IT applications. All too often, the users don't really know what they need, and the smart ones will admit it. Even if they did know, the business rules and your company's competitive environment will change before an 18-month "big bang" project ever gets deployed. CRM projects are not only less expensive when delivered incrementally, they are a better fit with the business needs. So it's important to get your project staff — and the CRM system's executive champions — comfortable with Agile.

When evaluating your CRM consultancy, look beyond the Web site and the sales pitch. Ask them for specific customer references where Agile methods positively impacted the project results. Make sure that the specific project lead who'll be assigned to your project has experience with Agile projects (particularly with customers of your size and IT profile). Ask them what will be important characteristics of a scrum master. Telltale signs of a poseur: if they give you a blank stare, or don't know whether the scrum master works for them or for you.

But the acid test of real Agile knowledge comes when you look at their proposal and project plan. If they are promising to deliver a fixed specification for a fixed price and/or a pre-determined schedule, watch out. Agile adherents cannot and will not make that commitment. As attractive as those promises might be, they're only possible if the project is a complete cookie-cutter or being bid with a ridiculously high margin. Either way, they're not being straight with you...and you're not going to get the optimal business value for your money.

### The Other Side of the Coin

If your implementation team should be Agile, does it make a difference if your CRM vendor delivers their product using the same principles?

Of the major CRM vendors, only salesforce.com and (at least aspirationally) SugarCRM use Agile teams to deliver their product. You really see it in the delivery schedule: new features show up at least three times a year. You also see it in the feature lists, which may look meager if you review a single release in isolation. Major functional enhancements are delivered incrementally, spanning two releases or more. In contrast, the traditional waterfall deliveries of Oracle ([ORCL](#)), Microsoft ([MSFT](#)), and SAP mean you'll see a larger feature set delivered in widely-spaced releases. While the traditional model provides lovely multi-year feature roadmaps, anybody who's been around for a while knows that the projected dates are almost always wishful thinking. Worse, the big-bang model does not have deep user-centered design baked in, so the user interfaces initially delivered in the waterfall model can be real impediments to end-user adoption.

Coming back to the KISSS principle, for CRM vendors it's more than just a matter of delivery schedule. The ability to deliver "separable" — highly modular — features depends upon the

architecture of the CRM system. Just because the product is written in Java or C# doesn't mean that it'll be easy to interface with or extend. If the foundation is 1M lines of code and 200 highly intertwined tables, it'll be much harder to write good extensions and flexible integrations. Instead, you want to see clean, consistent call-in and call-out exposure of all important system methods via Web services or RESTful APIs that don't require 10 levels of pointer-chasing to get to an object. Also, make sure that new APIs (and supporting documentation) are delivered more or less simultaneously with the CRM features they apply to. Check developer forums/discussion groups for indications of the scope and quality of the APIs and the documentation that comes with them. Even the best vendors have small gaps and errors in their docs, but you don't want your developers having to discover API behaviors through months of trial-and-error. This is a movie we've all seen too often.

The KISSS principle definitely applies to your vendors, your integrators...and your own project teams.

*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](#).**

© 2010 CXO Media Inc.