

# Software

Strategy & Technology  
For Software Executives

# BUSINESS

a webcom publication

Software Business Executive Report

June 8<sup>th</sup>, 2009

## The Enterprise Software Illusion

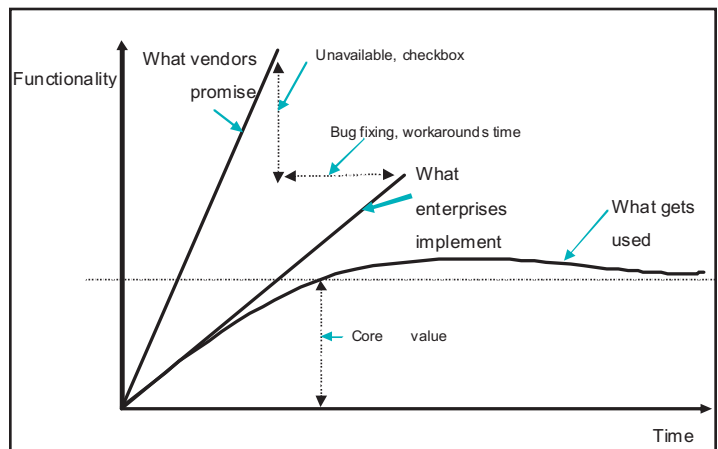
**Joe Ruck, CEO**  
*BoardVantage*

It is no secret that the traditional enterprise software business is undergoing fundamental changes. In the past couple of years, we have witnessed a wave of acquisitions; on the high end of the food chain we've seen industry giants gobbling up some very large rivals, including the hostile take-over of PeopleSoft and Siebel by Oracle and the friendlier buy-outs of Cognos and Business Objects by IBM and SAP respectively. In the midrange, there have also been numerous acquisitions including Web Methods and Hyperion. Many remaining independent players have seen their stock prices stagnate and virtually none have fully recovered from the bursting of the tech bubble. In the start-up scene there is a dearth of promising enterprise software as most VCs avoid the on-premise software model and are throwing their weight behind SaaS. What's going on here? Although every industry has its ups and downs, customers should be asking if enterprise software vendors will give them a good return on their investment.

Take CRM as an example. Gartner reports that 55 percent of projects fail to meet expectations and Butler Group reported a whopping 70 percent of projects failed. With those industry statistics well-understood, no rational person should even start a CRM project unless they believe they have some 'secret sauce' that sets them apart from the herd. While there are undoubtedly some exceptional implementations, the only consistent message from the market is that there is no 'secret sauce'. Some commentators have taken the view that this is primarily a failure of IT project management. There is a belief that just one more round of weekly progress meetings, hiring a big-shot project manager, outsourcing or some consultant's new-fangled methodology will fix the problem. But in the light of the breadth and depth of the challenges, is it fair or even reasonable to pin the blame on IT?

Having been involved in countless implementations, I have found that on typical projects problems arise the moment implementation starts. It is not unusual to discover that presumed

capabilities are non-existent or practically unusable, perhaps functionality which was added solely for the purpose of being able to "check the box". Two or three of those are enough to badly throw a project off-track. Also, a project will inevitably take longer as bugs are uncovered. Bugs are a fact of life and some of these will be fixed, while others will have to be worked around, but all will add to the schedule. Finally, after roll-out is complete, many organizations discover that the user experience is so different than presumed that usage itself is much less than anticipated, or that the system goes entirely unused. Its factors like these that lead to project failure rates of 50 percent or more.



**Figure 1: Implementation and Use of Functionality**

In this diagram, we illustrate the relationship between implemented and used functionality over time. Starting from the top left, vendors have no shortage of promises and a rapid flurry of new functionality announced with every press release. What actually gets implemented is always somewhat less than this for a couple of reasons. First, what is announced is often unavailable or what might be termed 'checkbox' functionality, designed to look good on paper, but not expected to be used in anger. Secondly, it always takes longer than expected to implement, due to the inevitable range of bugs. In some cases a fix may be found, but in most cases, it is more expeditious to simply work

around the bug. However, both situations end up taking both time and resource to delay the implementation. Finally, what actually gets used is usually only a fraction of what is available. Many CRM implementations are in fact used mainly only as simple shared contact databases, since that is what addresses the principal pain point of the user. More sophisticated functionality that in principle sounds attractive, can in practice simply be too complex to warrant the investment in training to master.

## **Who is to Blame for this Mess?**

Clearly the vendor makes an inviting target, but most vendors work hard to gather requirements, only to be disappointed that the feature set they built on the basis of customer input does not meet the need.

There are certain unspoken assumptions around requirement gathering: that end-users will adequately articulate what they want or need in advance of actually seeing it. Experience shows that in fact they struggle to do so and in many cases plainly can't. Management has a rosy but inaccurate view of their business processes, employees are reluctant to explain the ugly workarounds that are in fact central to the process, and both lack the imagination to see how things could be done differently. Any projects that depend on any significant amount of upfront analysis will therefore fail. The only approach that consistently works is to show them a working system – 'like this'. Users simply don't know what they want until they see it.

Perhaps the most peculiar feature of enterprise software is that this state of affairs has continued for so many years. Maybe it has to do with the fact that few people want to admit to spending six or seven figures on a failed project, so not surprisingly, few projects are ever publicly declared as failures. Failures may be blamed on individual vendors or individual team members, but never the basic premise of the approach taken or the industry as a whole.

This is the enterprise software fiction – the promise of business transformation, the reality of multiple project failures, and until recently at least, both vendors and customers participating in the fiction, with an every increasing amount of software ending up as shelfware.

SaaS neatly side-steps this, as "what you see, is what you get". In the event this proves inadequate, then the subscription can be simply cancelled. There is no huge sunk cost to try and recover or cover up. Conversely, the fact that the SaaS vendor needs to sell his or her product every day to the user means that service now becomes a differentiator. The vendor's success is now perfectly aligned with the project success, there can be no shelf-ware. Going back to our example of CRM, Yankee Group have specifically cited SaaS as a success factor in implementation. My own previous experience of zero success with enterprise CRM in three separate companies and one success out of one attempt with SaaS CRM, is an experience shared by many.

SaaS isn't the perfect solution to every software project, but for many it is, and remains the one type of IT project you can walk away from with your budget intact if it all goes wrong.

*Joe Ruck is president and CEO of BoardVantage. He has led many high-technology companies through successful growth to IPO or acquisition. Prior to joining BoardVantage, Joe was senior vice president of marketing at Interwoven and part of the team that drove the company through one of the most successful IPOs of 1999. Previously, he held sales, marketing, and executive positions at Sun Microsystems, Network Appliance, and Genesys Telecommunications, subsequently acquired by Alcatel. Joe holds a BS in engineering from Oregon State University and an MBA from Santa Clara University.*