cover story

The software revolution

By Michael Vitale*

During the past decade, a revolution has occurred in the way large private-sector organisations acquire the software that underpins their daily operations and generates the data they need to make better long-term decisions.

Rather than building such software themselves, or paying someone else to build it according to their specifications, many — and in some sectors, most — organisations have elected to purchase large Enterprise Resource Planning (ERP) systems.

Well-known ERP vendors such as SAP, PeopleSoft, and Oracle have grown rapidly as a result: their annual revenue increased from US$11 billion in 1997 to exceed US$20 billion in 2000.

Although their growth slowed temporarily after the world survived the Y2K crisis, these ERP companies are riding the crest of a wave driven by economic and technical factors that make their continued success likely.

Some Australian organisations have purchased ERP systems, but other organisations continue to struggle with homegrown administrative software that is often more expensive, more risky and less functional that what is available on the market.

There has been a lot of coverage in the media recently of the struggles of several academic institutions such as the University of NSW, Monash University and RMIT to install their purchased systems.

However, I believe that most organisations would be much better off buying software, so long as they install the software thoughtfully.

Professor Michael Vitale is the former Dean of the Australian Graduate School of Management. He has held a senior IT management position at the Prudential Insurance Company of America and been an Associate Professor at Harvard Business School.

There is little reason for organisations to dilute their resources, or to distract their managers' attention, by focusing on software development rather than on their primary mission.

Software development as an inherited disease

More than fifty years after mankind began writing software, the statistics on software projects remain dismal.

The typical software development project is delivered late, well over budget, and without all of the promised functionality.

And that's the good news — many projects deliver nothing at all, having been cancelled midstream when the lack of progress became too obvious and too expensive to ignore.

Take, for example, the case of CASMAC (Core Australian Specification for Management and Administrative Computing), which was intended to devise a common set of management and administrative systems across the Australian university network.

In 1991 some thirty Australian universities agreed to develop general administrative computing systems that individual universities could then tailor to meet local needs.

In 1998 CASMAC was abandoned, having delivered none of the required software. Payments and in-kind contributions to the failed CASMAC project by just four of the universities involved were estimated by the Victorian Auditor-General to exceed A$9 million.

One of the Australian universities involved in the CASMAC fiasco later spent $30 million on its ERP system, and another spent almost $5 million.

This story is unusual only because it is better documented than most other software failures.

The lessons of the CASMAC experience are not just limited to the academic sector; all commercial organisations should understand that software development remains a difficult, expensive, and risky undertaking.

The objections

The primary objections to purchased software are that it is expensive and that it does not fit the unique operational processes of a given organisation.

Many organisations operate under tight spending constraints, and software can seem expensive.

Naturally, the cost of any software must be compared with the benefits it will provide, as well as with the cost of other options, including custom-built software.

It is important to bear in mind that, if properly chosen and implemented, purchased software can be considerably less risky than the homegrown variety.

Most software vendors and consultants, if pressed, are willing to sign fixed-price contracts with penalty clauses. Most in-house software development organisations are neither able nor willing to enter into such agreements, which in any case would simply result in moving money around within an institution.

Many organisations operate day to day on the basis of processes that are undocumented and more the result of historical accident than of deliberate design.

Yet these same processes are sometimes clung to tenaciously by staff and then embedded into software code, often at significant costs, by software builders chanting the mantra of 'meeting user needs'.

Outside the front door of the Fat Ladies Arms, a pub in Wellington, is a sign titled 'Cowboy Philosophy'. The sign reads: 'About half our problems in life come from wanting our own way, and the other half from getting it.'[1]

This cowboy philosophy sums up precisely the difficulty of incorporating 'user needs' into software.

In fact, users do not have 'needs' — for if they did, the typical software development process of prioritising those 'needs' and selecting the ones to be included in a new system would not make any sense.

Users have ideas, desires, prejudices, habits, and so on — but rarely 'needs'. And no administrative process, no matter how old or well known, should be allowed to drive software development unless it is truly central to an organisation's strategy.

Accounting, financial, human resource, and other administrative processes are highly unlikely to be in this category for most organisations.

In the absence of hard evidence, it is equally unlikely that a given organisation's processes represent best practice.

Many organisations would be better off buying a package and changing their processes to match.

This of course requires a high degree of top management involvement and support, without which no organisation should embark on a major software project in any case.

The challenges

Given the size of many software implementations, installing the software clearly creates a set of challenges that must be overcome in order to achieve success — or in some cases even to keep the organisation running.

Take, for example, the highly-publicised struggles of RMIT University in Melbourne, which has run into considerable difficulties with the installation of the PeopleSoft ERP system.

Above: Professor Michael Vitale

The university says it has already spent more than $30 million buying, customising and installing the software, with at least another $15 million to go before the project is complete.

Along the way the project angered students who were wrongly billed, corrupted the university's databases, and brought about a review by the Victorian Auditor-General, the same office that investigated problems with CASMAC five years earlier.

Clearly the RMIT experience has not been a happy one.

RMIT Vice-Chancellor Ruth Duncan has attributed the problems to poor project management by the university and the high degree of modification of the purchased software.

Project management is an issue for custom-built software as well as for purchased software; an organisation lacking strong skills in this area should certainly be extremely cautious about undertaking in-house development.

The question of modification comes back to the issue of 'needs', which is certainly no easier to manage when software is being developed than when purchased software is being modified.

In short, while the experiences of RMIT and other organisations with large purchased systems have been less than ideal, there is little reason to believe that they would have been even as well off had they chosen to develop the software themselves.

Their inability to successfully manage projects and make decisions about software features are likely to have caused at least as much trouble had they chosen to build software rather than buy it.

Moving forward

The importance of administrative computing to the smooth, economical operation of an organisation cannot be denied.

Organisations will need to become very good at selecting and implementing such software, including managing the organisational and personnel changes required for implementation to succeed.

But there is no reason whatsoever for organisations to become good at writing administrative software, and there is no reason for them to tolerate badly written systems.

The software market is richly supplied with eager vendors of customisable packages who in the current market will compete vigorously for an organisation's business.

Taking advantage of this market can be a significant step toward overcoming an often-unexamined habit of in-house software development.

The better option for most organisations is to find the pre-written software package that best fits their information technology architecture, their strategy, and their way of doing business.

1 *A second sign outside the same door reads, 'This is the best bar in the world'. The author is considerably less certain of the validity of this second sign.*

Reprinted with permission from the *Australian Law Management Journal*, Law Council of Australia, Legal Practice Section. http://www.lawcouncil.asn.au/lps/publications.html.