RE-COVER-ING FROM SOFTWARE DEVELOPMENT

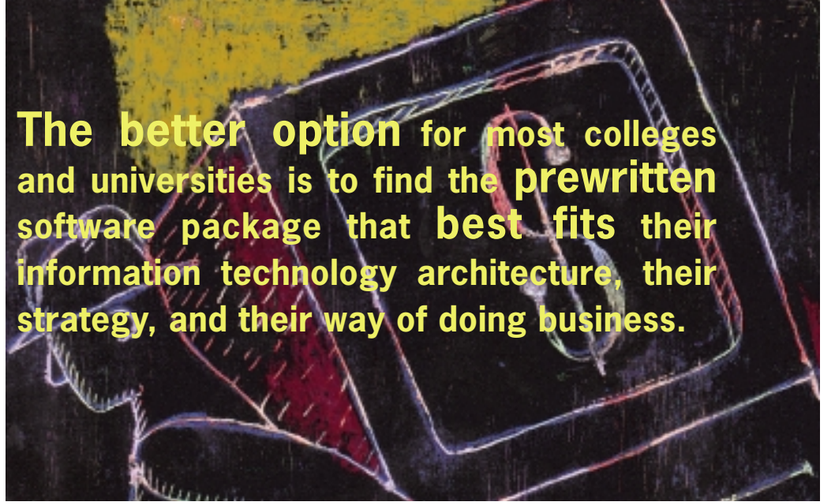
BY MICHAEL R. VITALE

The Software Revolution

During the past decade, a revolution has occurred in the way large private-sector organizations 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—organizations have elected to purchase large Enterprise Resource Planning (ERP) systems. Well-known ERP vendors, including SAP, Baan, PeopleSoft, and Oracle, have grown rapidly as a result. Although their growth has slowed temporarily as prospective customers batten down the technological hatches for the transition to Y2K, these ERP companies are riding the crest of a wave driven by economic and technical factors that make their continued success likely.

Michael R. Vitale is Professor in the Centre for Management of Information Technology, Melbourne Business School, University of Melbourne. Earlier in his career he was the Director of Administrative Computing at Dartmouth College and later Associate Professor at Harvard Business School.

Illustration by Brant Day



Some academic institutions, including MIT, have also purchased ERP systems, but other institutions continue to struggle with homegrown administrative software that is often more expensive, riskier, and less functional that what is available on the market. The usual excuse for undertaking the additional effort involved is that the institution's processes and procedures are so special that they must be supported by custom-built systems. The premise of this article is that most institutions would be much better off buying software and then adopting the built-in best-practice processes that come with the packages. There is simply very little reason for colleges and universities to dilute their resources, and distract their managers' attention, by focusing on software development rather than their primary educational mission.

Software Development as an Inherited Disease

More than forty 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. Higher education institutions are by no means exempt from these problems. Take for example CASMAC, the Core Australian Specification for Management and Administrative Computing.

CASMAC was intended to devise a common set of management and administrative systems across the Australian university network. Some thirty Australian universities agreed in 1991 to develop general administrative computing systems that individual universities could then tailor to meet local needs. By 1993, the group of universities had been unable to agree on a hardware platform and therefore broke into two subgroups, one using Oracle and the other using the "Powerhouse" development environment supplied by CHA Computer Solutions Pty Ltd. The latter group of nineteen universities, called the "UniPower group," agreed to pay costs estimated at approximately A$11 million for software development and licenses; the universities would also contribute staff time to the project.

In 1998 the UniPower group abandoned CASMAC, citing the inability of CHA to deliver the required software. The auditor-general of the Australian state of Victoria, who investigated the failure of CASMAC, noted that "certain deficiencies in the project management and development process" contributed to the failure.[1] The auditor-general's report further noted:

> *Certain of the Victorian participating universities had not:*
> - *quantified the expected benefits to be derived from the project, nor prepared a cost-benefit analysis to determine whether their participation in the project was in the best interests of the university;*
> - *prepared a detailed budget to determine the likely costs of developing, implementing and maintaining the UniPower CASMAC software on an ongoing basis;*
> - *developed suitable financial management systems to accurately record and monitor ongoing project expenditure; and*
> - *established procedures to calculate and record the cost of providing inhouse resources to UniPower.*[2]

Payments and in-kind contributions to the failed CASMAC project by just four of the nineteen universities were estimated by the auditor-general to exceed A$9 million.[3] By mid-1999, at least two of the UniPower universities had purchased ERP systems to carry out the tasks they had hoped to perform via CASMAC.

The story of CASMAC is unusual only because it is better documented than most other software failures. The overall lesson is clear: software development remains a difficult, expensive, and risky undertaking, even when a professional software development organization is involved. Most universities have no business doing it.

The Alternatives

The CAUSE side of EDUCAUSE began as the College and University Systems Exchange, founded in the United States in 1971 to facilitate the sharing of administrative software among institutions of higher education. By 1972 CAUSE had created a software library on magnetic tape and punched cards and had begun to receive complaints that the software it distributed did not match the documentation. Although EDUCAUSE has moved away from this original goal, the idea of using prewritten software remains valid. There is no particular reason, however, to imagine that using software custom-built by the technical staff of one university will produce good results at another—particularly if, like much homegrown software, the system isn't delivering according to specifications in the first place. The better option for most colleges and universities is to find the prewritten software package that best fits their information technology architecture, their strategy, and their way of doing business. In most cases, an institution may not even need to issue an elaborate or detailed request for proposal (RFP). The number of truly qualified vendors is likely to be small, and if the university follows the lessons learned, at significant cost, by private organizations, it will not attempt to make any modifications to the source code of the package.

This approach starts in the software market, rather than the university. It begins with the observation that the commercial software market has matured to the point that richly featured solutions are readily and widely available. The economics of commercial software development are well understood and heavily favor vendors who can sell multiple copies of a given package, rather than developing one-off solutions.

The Objections

The primary objections to purchased software are, first, that it is expensive and, second, that it does not fit the unique operational processes of a given institution. Many educational institutions operate under tight spending constraints, and software can seem expensive. One of the Australian universities involved in the CASMAC fiasco later spent A$30 million on its ERP system, and another spent almost A$5 million. 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 as well that, 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 organizations are neither able nor willing to enter into such agreements, which in any case would simply result in moving money around within an institution.

Like other organizations, institutions of higher education operate day to day on the basis of processes that are often undocumented and the result of historical accident more than deliberate design. The Victorian auditor-general's comments, quoted above, regarding the management processes of the UniPower universities are probably indicative of the overall level of process quality in many colleges and universities. 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, New Zealand, 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."[4] 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 prioritizing the "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 not "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 organization's strategy. Accounting, financial, human resource, and other administrative processes are highly unlikely to be in this category for universities.

In the absence of hard evidence, it is equally unlikely that a given university's processes represent best practice. Higher education institutions are not in the business of developing or honing administrative processes and rightfully do not often devote much attention to them. Most universities 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 institution should embark on a major software project in any case.

Moving Forward

The importance of administrative computing to the smooth, economical operation of a college or university cannot be denied; indeed, as university activities become steadily more diverse and more complicated, the importance of good administration, and good underlying administrative systems, will only grow. Institutions of higher education will need to become very good at selecting and implementing such software, including managing the organizational and personnel changes required for implementation to succeed. But there is no reason whatsoever for most colleges and universities 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 competent, eager vendors of customizable packages, vendors who will compete for an institution's business. Taking advantage of this market can be a significant step toward overcoming an often unexamined habit of in-house software development.

Notes

1. See the Web site of the Victorian Auditor-General's Office (Australia): <http://www.audit.vic.gov.au/mp99/mp99doc.htm>, May 26, 1999 (visited October 31, 1999), paragraph 3.1.10.
2. Ibid., paragraph 3.1.14.
3. Ibid., paragraph 3.1.15.
4. 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.