

## Changing Assumptions in Best Practice

**M**ore than a decade ago, the predominant means by which colleges and universities deployed major information systems shifted from in-house development to the acquisition of packaged software. Today, most of us prefer to buy whenever we can and build only when we must. There are reasons for this preference to buy rather than build. Packaged systems can offer some important advantages over local development. Commercial firms have far more resources than individual college or university IT shops. They make huge investments in functional design, software development, testing, quality assurance, ongoing improvements, and problem resolution.

One of the hallmarks of current-generation system products, notably the integrated ERP (enterprise resource planning) suites that dominate campus business functions, is the encoding of “best practice” in the software. As an integral part of system design and evolution, software firms work with functional groups or product advisory boards to refine functional specifications. When an institution purchases the package, it receives the collected wisdom of the teams who defined best practice. By contrast, most of the business processes at our own institutions are not the result of a well-planned and thoughtfully designed model. They just, well, sort of happened. They evolved over time, invariably from paper-based, department-centric procedures. The evolution is usually *ad hoc*, in the form of additional steps (for instance, including further signatures of authorization), which add time and complexity to the

business process. Steps are often added to business processes; they are seldom eliminated.

The less successful in-house systems of the past encoded these *ad hoc* business processes, perpetuating the status quo rather than being drivers for improvement. Commercial packages based on best practice altered that situation. Uniformly, the advice given to ERP implementers was—and is—to “go vanilla.” Do not make changes or customize the software. Doing so will result in long-term maintenance and support problems, increased costs, and the opportunity cost of not deploying best practice as encoded in the ERP software.

But what if some of our assumptions changed? What if, instead of encoding a mess of *ad hoc* business processes, we in higher education thoughtfully identified those processes that could potentially differentiate us from our peers and competitors? What if we planned and designed our practices, at least the ones that have the potential for differentiation? What if the perspective we took in our administrative processes were that of the end-user rather than that of the department—the latter viewpoint dominating best-practice software packages?

In my view, there are several factors that, particularly when taken together, can broaden the available alternatives for planning next-generation campus systems. Some of these factors are already exerting increasing influence on the commercial systems marketplace:

- Institutional commitment to designing end-to-end business and academic processes so that they are

end-user centric, plus identifying those processes that have the potential to strategically differentiate individual institutions

- Modular systems made up of well-defined business processes that themselves are composed of software services built on a foundation of rigorous open standards
- Community source (open source) software development and the rapid evolution of multiple organizational models that can sustain product development and support over time
- Next-generation tools that are designed on a network model (rather than a computer model) so that they can leverage the Internet’s critical success factors as identified by Vint Cerf: end-to-end design, layered architecture, and open standards
- A growing knowledge base of how to make these new approaches work effectively in higher education

Some institutions, my own included, have used a structured methodology to streamline or radically redesign key institutional processes. At the University of British Columbia, we use an extremely effective methodology adapted from the Hammer-Champy method, introduced to us by JM Associates. The methodology scales well from small improvements to radical changes where the old process is “blown up” and an entirely new one is created. It requires an investment of people’s time, commitment, and money, but people who participate in the process-redesign projects emerge from the experience with energy and enthusiasm to propel more positive change in how

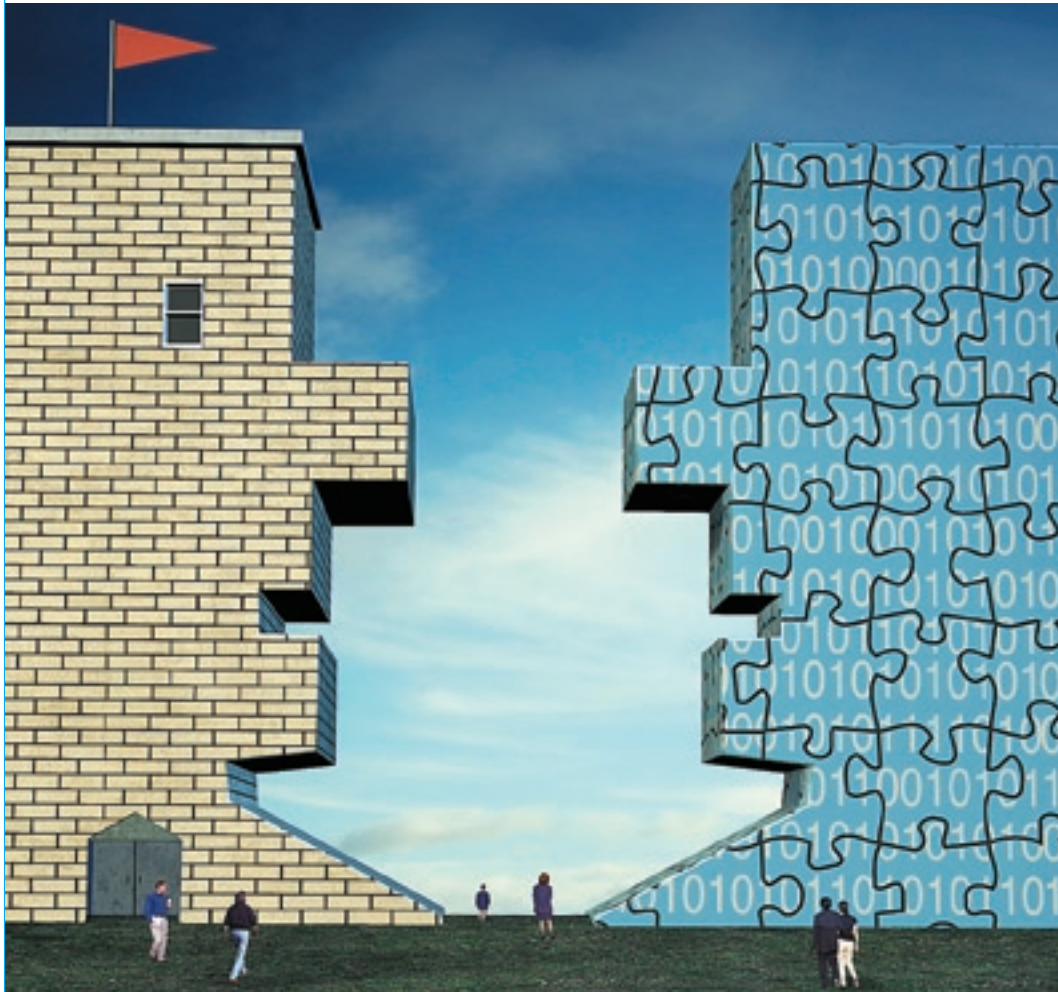


Illustration by Randy Lyhus, © 2006

things are done within the university.

Institutionally defined business or academic processes may or may not fit the embedded definition of “best practice” in current-generation systems. This is where the promise of “software as a service” based on a service-oriented architecture (SOA) becomes important. Next-generation systems will very likely be based on standards-based reusable components that can be assembled and configured to implement a business process. There is reason for being careful not to oversell the benefits of SOA, but there is even more reason for being optimistic. Major systems vendors are investing heavily in Web-services-based products. They are working with standards bodies like the World Wide Web Consortium (W3C), which *should* be a good thing for the industry and for higher education institutions. In time, we should see next-generation suites of commercial systems based on SOA principles.

Some people worry that the big players in the IT industry could have undue in-

fluence on de facto standards by tweaking those standards toward proprietary protocols. There’s nothing new here. But the vitality of the community source movement provides an interesting alternative. Higher education has a long tradition of collaborative software development, possibly predating the term *open source*. Colleges and universities have been a breeding ground for the standards-based thinking that drove the development of TCP/IP and other bedrock ideas on which our most successful research and learning collaborations are based.

More recently, Sakai, Quali, and JA-SIG’s uPortal framework—to name but a few—have demonstrated the ability of those in our community to work together on common projects. Today, uPortal is in production at more than 800 colleges and universities around the world, fully half of which carry a commercial rebranding through a vendor partnership. Clearly, the community source movement is not anticommercial. Rather, it is influencing, if not redefining, the prevailing business

model and alternatives for software choice. The organizational structures of the Sakai Foundation and JA-SIG illustrate different ways of building communities. Other models—Wikipedia and the Java Community Process come to mind—suggest that there is no one best way to organize and sustain community source processes.

The emergence of new tools, particularly new programming languages built on an Internet model, will influence campus systems only if they are applied within the context of user-centric business processes and modular services. Still, they are important. It would be difficult to imagine today’s cellphones being powered by Assembler rather than Java. Yet Java alone did not lead to the innovative power of mobile phones.

Whatever their contributing role may become, there is a lot of interest in tools, such as Ruby on Rails, that mirror the successful attributes of the Internet.

To put all of this together requires people, knowledge, and commitment. One of the wonderful aspects of working in the higher education environment is our predisposition to collaborate and to share what we learn. Growth is accelerating in our collective knowledge base around business-process redesign, service-oriented architecture, community source processes, and next-generation technologies. Perhaps we are not so far from the day when we can lay claim to college and university systems that are based not on best practice as defined by someone else but, rather, on *our* practice.



**Ted Dodds is Associate Vice President for Information Technology and CIO at the University of British Columbia.**