

**To:** Members of the Faculty Senate  
**From:** Eric Roberts, Professor of Computer Science  
**Subject:** Here be dragons: The economics of enterprise software systems  
**Date:** May 27, 2004

---

In seeking to gain some insight into why the installation of the new administrative software systems has been so painful, one of the most difficult things to understand is why the use of commercial software does not produce the benefits one expects. As Dean Joss observed at the last Senate meeting, enterprise software systems have often had significant value for companies, particularly those that are able to optimize their business processes to fit the requirements of the new tools. Moreover, there are strong theoretical arguments to suggest that the economics of software production should provide substantial benefits to business models that allow the costs of software development to be shared over a wide range of users. For some reason, however, these theoretical advantages seem to have less effect in the context of a large, decentralized university like Stanford. The purpose of this memo is to offer some reasons that explain this observed phenomenon.

### **Experience of other institutions**

The first point to make is that the empirical evidence shows that Stanford is by no means alone in experiencing problems with its adoption of data processing systems. Over the last few years, *The Chronicle of Higher Education* has included reports from many institutions about horror stories they've encountered when putting into place new data processing systems. I have seen no reports of schools in which such installations went smoothly, although those stories may not be considered news under the "if it bleeds, it leads" philosophy of reporting. However, I spoke at some length after the last Senate meeting with Chief Financial Officer Randy Livingston, who admits that he hears similar stories whenever he goes to meetings with his counterparts at other institutions. Life in the world of educational data processing is not going well.

### **The "buy not build" philosophy**

In 1996, Stanford, under the direction of then-Provost Condoleezza Rice, made an explicit strategic decision to abandon its locally developed data processing software in favor of adapting a commercial system to the needs of the university. The justification for such a strategy is primarily economic. Software systems are expensive and complex. What's more, the expense of a software system lies almost entirely in its development; once a system is built and tested, the marginal cost of delivering that same system to other users is typically quite small. The concentration of cost in the development phase creates a strong incentive to share development expenses over a large user base. If it costs \$10,000,000 to develop a system, it seems foolish for a single institutional user to bear that cost alone. Given that the bulk of that \$10,000,000 represents development, it makes far more sense—at least in theory—for a consortium of institutions to purchase software from a vendor that can then distribute those costs over the community of users. If there are, for example, 100 users of a commercial system with a development cost of \$10,000,000, a vendor should be able to charge each institution \$100,000 and still break even on the deal. Even if the price is raised to generate a modest profit for the

vendor, the enormous savings attached to shared development should reduce the cost to each individual institution.

This economic argument provides the underlying rationale for the “buy not build” strategy, which gained considerable momentum among educational institutions during the 1990s. While the theory sounds wonderful on paper, anyone looking at the experience of educational institutions in recent years must find some way to explain the empirical observation that most of those institutions report an increase in costs, rather than the anticipated reduction.

In part, the failure to achieve success lies in a failure to implement the “buy not build” strategy effectively. As Dean Joss noted at the last Senate meeting, the success of any enterprise system depends on refashioning the business practices of the institution to match the software rather than trying to change the software to accommodate the idiosyncrasies of the institution. Changing the software violates the underlying economic assumption that allows for the reduction in cost. If each institution tailors the system to suit its needs, the cost advantage vanishes.

In a 2002 memo in response to my report on the problems in PeopleSoft, Chris Handley cited Stanford’s failure to implement the “buy not build” philosophy successfully as the reason for the disastrous decision, later abandoned, to implement Oracle’s Core Financials (COREFIN) system at Stanford over the years from 1996 to 1998:

In the implementation of COREFIN during the mid 1990s, Stanford violated the “Buy not Build” approach repeatedly. The result was that the replacement of the General Ledger that should have taken less than a year and cost approximately \$7 or 8 million actually took 4 years and cost \$35 million. Additionally the software product of all of this effort was so highly customized that it was, in practice, not able to be upgraded to later versions.

We have been here before. And yet, in the published minutes of the May 13<sup>th</sup> meeting, Mr. Handley admits that “It is true that we did not re-engineer our business processes before rolling out the systems. So part of the complexity of this is we’re using old processes and new systems. We have to change that.”

I think it is important to question whether we should in fact be seeking to change institutional behavior in accordance to the dictates of our software systems. In many ways, much of the relief that people expressed at the last Senate meeting comes from the fact that the university is in certain ways moving in the opposite direction. One of the central conclusions from the February Senate meeting was a decision to allocate more resources to ensuring that the Oracle software would meet the needs of university business practices, and not the other way around. The establishment of a group of acceptance partners and the inclusion of more end users in the requirements-definition process has refocused the development effort to the needs of the business units and away from the baseline functionality of the Oracle systems as they were supplied. In terms of getting our administrative work done and ensure that Stanford remains in compliance with federal regulations on grants and contracts, that strategic decision was absolutely correct. At the same time, it moves us even further away from the idealized assumptions that underlie the cost advantage of the “buy not build” philosophy.

## Why “buy not build” fails in the university context

In this final section, I intend to take this argument farther and assert that the “buy not build” philosophy is ill-suited to meeting the data-processing needs of a university. In the fall, I hope to develop these ideas more fully in the context of a comprehensive review. For the moment, all I can do is outline my reservations.

- *The decentralized structure of a university makes it impossible to achieve the theoretical cost advantage of the “buy not build” strategy.* Companies that have been successful in achieving cost reduction through enterprise software have done so by forcing a change in the culture and processes with which those organizations work. Such change is possible in the typical top-down management culture of a corporation but is difficult to replicate in the highly balkanized structure of a university in which power is decentralized to the point that the central authority has very little control over the operation of the many research and academic units that make up the university as a whole. Even if one wanted to change that structure for the economic benefit of the university as a whole, the existing power dynamics within the institution militate against such change. Centralization is antithetical to the freedom and flexibility an academic institution needs to maintain its intellectually focused character.
- *Universities operate under exogenous constraints quite different from those in the corporate community for which most enterprise systems are designed.* Most of the complaints that I have heard within the School of Engineering center on the difficulty administrators have obtaining the data necessary to maintain compliance for grants and contracts. It is impossible to adopt the position that the university must adjust its practice to fit what is provided by the software if external agencies require us to maintain information that the software does not provide.
- *Enterprise system vendors have an economic incentive to fit universities into a broader base of clients for which a different set of assumptions applies.* From the vendor’s point of view, profits are maximized by sharing development costs over the widest possible set of customers. As a result, many vendors seek to shoehorn university requirements into systems designed for the corporate world. This problem is illustrated quite clearly in the fact that the current Oracle systems assume that procurement orders carry sales tax, which no nonprofit university pays. Such attempts to make the same software work for too wide an audience end up reducing its effectiveness for those clients that fall outside the community for which the features were designed.
- *Most enterprise software companies have difficulty attracting the top talent necessary to develop high-quality software at a reasonable cost.* The economic advantage of the “build not buy” philosophy is predicated on the assumption that the development cost of building an administrative data processing system is roughly comparable no matter who builds that system. In the complex economics of software development, that assumption may not apply. Software developers vary enormously in individual productivity, sometimes by factors that approach two orders of magnitude or more. Database companies like Oracle and PeopleSoft have not been the companies that our most talented computer science graduates have chosen, who seem to regard the work of those companies as more mundane and less intellectually challenging than that of a startup like Google. As a result, the dollar that Oracle spends on software development may not go nearly as far as the same dollar spent by an entity that can attract highly productive programmers. Interestingly, universities are one of the few places

that *can* attract such talent. That fact changes the cost equation. If a university could build a system for \$100,000, there is no longer a cost advantage to sharing the costs of a \$10,000,000 implementation effort undertaken by programmers who are 100 times less productive.

- *Even if commercial enterprise systems can be developed at lower cost, that fact alone does not guarantee that those systems can be purchased at a lower price.* The “invisible hand” of the marketplace is supposed to ensure that prices gravitate toward an equilibrium that represents the underlying cost. Classical price theory, however, depends on a variety of assumptions about the characteristics of the market. In particular, price theory assumes that buyers and sellers in that marketplace have the flexibility to exercise individual choice to move the market toward its equilibrium position. For a market to operate efficiently, new sellers have to be able to enter that market to undercut prices that are too high. Conversely, buyers must maintain the flexibility to choose the lower-priced commodity or service. In the world of enterprise software, each of these assumptions is violated. The cost of establishing a new company in the enterprise software business creates an enormous barrier to entry, particularly when coupled with enormous network advantage enjoyed by industry leaders. On the other side of the equation, buyers face an enormous barrier to exit given the scale of the investment required to change from one vendor to another. Universities that adopt software systems from a particular vendor become locked into that relationship. As a result, the vendor faces no market constraint on price.

Of all the problems with enterprise software systems identified in this memo, I am most concerned about the last. At the same February meeting at which Chris Handley made his first report of the year on the administrative systems, Faculty Senate approved a resolution putting Stanford on record as opposing the outrageous pricing structure adopted by Elsevier and other for-profit journals. In that context, the university sought to develop alternative distribution channels to avoid being held hostage to a pricing structure that had lost its market moorings. In HighWire Press, we have an in-house enterprise that allows the university to “support affordable scholarly journals” while protecting us against the vicissitudes of market failure. If you replace the sentence

Special attention should be paid to for-profit journals in general and to those published by Elsevier in particular.

with the parallel

Special attention should be paid to for-profit software vendors in general and to those produced by Oracle in particular.

very little of the underlying arguments would need to change.