---

**References**

Hannon, C. (2004, October 22). Just Say No to Poorly Designed Software. *Chronicle of Higher Education*, p. B16. Retrieved from http://ezp.lib.cwu.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=14905357&site=ehost-live



---

**Just Say No to Poorly Designed Software**

IN THE PAST 10 YEARS I have taught at six colleges and universities. During that time I have been a user of countless information systems, including those that manage courses, control "smart" classrooms, and organize student registration and advising. The libraries alone at those institutions have more than a dozen different systems. To put it mildly, none of the systems was as good as it could -- or should -- have been.

Not once in those 10 years have I been interviewed as a potential user of a new information system, or consulted by its designers (those who create the system's user interface) or developers (those who write the system's code) to ensure that it would help me accomplish my goals or improve my work. Rather, I have been tantalized by the promises of one system after another only to be disappointed by the final product, which all too often has an interface that is difficult to navigate, includes dozens of features that are only marginally useful, and -- most aggravating of all -- lacks features that I had assumed would be included.

Last spring, for example, my college adopted an online registration-and-advising system. It is a mature product, used by hundreds of other colleges and universities. Yet its interface is crude and inefficient, requiring me to click at least 11 times to reach an advisee's transcript.

If the system's designers had interviewed faculty members about how we used paper records in advising, they would have found that we spent a lot of time looking through files for transcript information. They would have identified reading transcripts as an essential goal of users of the system, and designed the software to make finding that information as quick and painless as possible.

The new system has other flaws. Incredibly, it does not allow me to send an e-mail message to all my advisees, or to schedule appointments to meet with students. Those are some of the most common faculty tasks during registration, yet they were apparently overlooked in the design of the system.

Like many colleges across the country, mine is in the midst of a multiyear adoption of an information system that will handle everything from human resources to admissions, registration, and billing for the entire institution. One of the many changes that have resulted from that adoption is a new system for reporting departmental budgets. Our previous budget report was a study in simplicity: a single column of budget items, a single row reporting activity for each item. The new report is incomprehensible, including a great deal of information about such matters as faculty salaries, federal work-study dollars, and administrative and secretarial wages. I can certainly imagine uses for that information. But the new form does not meet my most common goals when I look at such documents: to see at a glance how much has been spent from, and how much remains in, each of my budget lines.

As an interface to the new information system, the budget report fails to help me accomplish my goals as a department chairman. The same can be said of the interfaces of most of the information systems on the campuses where I have worked. Why do we put up with such bad software design?

Alan Cooper offers two reasons in The Inmates Are Running the Asylum: Why High-Tech Products Drive Us Crazy and How to Restore the Sanity (Sams, 1999): Either we blame ourselves for the frustrations caused by technology, assuming it's impossible to create better software, or we have become apologists for bad design because we are so grateful for any system that offers a respite from the ordeals of dealing with paper-based information systems. In Cooper's metaphor, we have become so impressed with the dancing bears of technology that we ignore the fact that they don't dance very well.

TO SOLVE the problem of inadequate computer systems, administrators at the highest levels on our campuses must refuse to do business with companies that do not begin their systems-analysis process by interviewing potential users of their systems. Designers need to understand how users accomplish their goals with current systems (whether electronic or paper-based), and to develop systems based on meeting users' goals easily.

Systems integrators, who work in teams to install new systems and make them "speak" to other systems, often do invite users to provide feedback. But unless you were interviewed during the design of the system that is to be integrated into others on your campus, you should refuse to participate in those sessions. They are intended either to solicit validation of an already determined interface design (so that designers can claim that the design has been tested by users), or to make sure the systems work when a lot of people try to use them at the same time. Any comments that you offer at that stage about making the system more useful to you and your colleagues will be received politely as suggestions for future versions of the product, and then forgotten.

One of the reasons that few designers focus on users' goals is that in most cases, the customer for campus information systems has different motives from the users. If your institution has adopted a campuswide calendar system, for instance, the customer -- the person who evaluated the available systems and chose one -- was probably your network administrator or another employee in your college's administrative-computing office. Although in some cases the customer will also be a user of the system, his or her goals in determining which system to purchase are wholly unconnected to the goals of most users. The customer needs systems that are affordable, are easy to install and support, and integrate well with other systems on the campus.

The customer's list of system requirements will generally include a reference to "ease of use," but because that is not one of the highest priorities, it rarely leads to the designers' interviewing users about their needs. Moreover, chief information officers traditionally evaluate their integration teams based on whether a system was deployed on time, within budget, and with minimal loss of data and interruption of service. Rarely is user satisfaction given the same weight as those criteria.

If a new system is developed in house, there is no excuse for neglecting to include user interviews at the earliest stages of the design process. With commercial products, colleges and universities are often at the mercy of vendors. Ironically, the very maturity of a commercial system can become a liability: It may have been designed before something as obvious as integration with e-mail software was considered an essential feature. Yet maturity implies stability, and campus information officers are understandably wary of awarding contracts to new companies -- those most likely to follow sound design procedures -- for fear that they will be out of business, and unable to support their products, a few years later.

Software companies argue that introducing new design methods would delay their products' getting to market, and that their competitors would take advantage of the delay to steal their market share. But the industry is replete with examples that contradict that argument. Google was not the first Internet search engine; it was the first designed with a focus on the typical user's goal of searching for Web sites rather than on companies' goal of advertising their services. Similarly, no smart phone or personal digital assistant will dominate the market unless it helps consumers accomplish their most basic goals easily and efficiently. Even the early battles over course-management systems have been fought on the grounds of design: WebCT has always had more features than its competitors, but Blackboard -- now the most widely adopted system -- made an early commitment to an interaction design that helps faculty members and students accomplish their most common teaching and learning goals.

My colleagues and I try to teach our students to be informed and critical users of information technology. We can reinforce that lesson by demanding better information technology ourselves, and urging our institutions not to buy poorly designed software. Computer systems can be very flexible; it's time to insist that their designers be accommodating, too.

~~~~~~~~

By Charles Hannon

Charles Hannon is an associate professor and the chair of the program in information-technology leadership at Washington & Jefferson College. He is also the author of Faulkner and the Discourses of Culture, forthcoming from Louisiana State University Press.

---
~~~~~~~~