Copyright Notice

This paper was presented at EDUCAUSE 2000 in Nashville, October 10-13, 2000. It is the intellectual property of the author(s). Permission to print or disseminate all or part of this material is granted provided that the copies are not made or distributed for commercial advantage and the title and author(s) of the paper appear. To copy or disseminate otherwise, or to republish in any form, requires written permission from the author(s).
A User-Centered Approach to Student Information Systems Design

Calvin Chan
Instrumentation Technologist
The Graduate School
University of Wisconsin-Madison
Madison
Wisconsin

Osmond Chen
Project Leader Specialist
Division of Information Technology
University of Wisconsin-Madison
Madison
Wisconsin

Domain 4
Friday, October 13
8:15 a.m. - 9:00 a.m.
Canal A&B

Abstract

In writing applications to supplement PeopleSoft functionality at the University of Wisconsin-Madison, we were reminded that a user-centered iterative approach was the only way to ensure success. But what strategies can system designers employ to truly work from a user perspective? This paper will discuss our approach and its results.

Electronic collaboration between the Graduate School and 120 academic departments virtually stopped with the new student information system. We needed solutions that suited over 200 users and a new approach that centered on the user and encouraged change as needs changed and as our understanding of processes improved.
A User-Centered Approach to Student Information Systems Design

In writing applications to supplement PeopleSoft functionality on our campus, we were reminded that a user-centered iterative approach was the only way to ensure success. But what strategies can system designers employ to truly work from a user perspective? This paper will discuss our approach and its results.

Electronic collaboration between the Graduate School and 120 academic departments virtually stopped with the introduction of a new, vendor-provided student information system. Data entry was slow and the audit trail was not functional. Even for data viewing, training was long and expensive, and screens customized to meet user needs would require significant re-programming at considerable cost at each upgrade. We needed solutions that suited over 200 users in the central unit and throughout campus. We needed a new approach that centered on the user and encouraged change as needs changed and as our understanding of processes improved.

Admissions Process

The admissions process at the University of Wisconsin-Madison Graduate School is a distributed cooperative process. Considerable information moves back and forth between the Graduate School Admissions Office and the academic departments. The Graduate School Admissions Office is responsible for initial data entry, credential evaluation, confirming minimum Graduate School requirements, checking visa requirements for international applicants, and the final admit. For their part, the academic departments are responsible for the rest of the admissions process, such as, checking their own minimum requirements for admission, faculty review, and recommendation of admission.

After identifying critical processes that the new student information system could not handle, users and software engineers collaborated on a supplementary system. We identified users, analyzed our processes, identified gaps, and developed solutions. During the process, we identified and focused on two types of users: internal and external users. Internal users consisted of the Graduate School Admissions staff and external users included admissions staff from the academic departments around campus. We made some distinctions on the characteristics of each group. Then we put external and internal users together with the software engineers to create a prototype of the supplementary system. We implemented the system as soon as it had some useful functions. We worked closely with both internal and external users, consulting them at the earliest stage and at sequent iterations of the design and development process.
Internal and External Users

Internal users, consisting of the Graduate School Admissions staff, work in close proximity (one floor of one wing of one building). Since they work together on similar tasks and know each other well, it is feasible for them to check paper documents quickly if the need arises. They need to see what others in the Graduate School Admissions Office and what departments are doing with an applicant. Furthermore, they have a sizeable technical staff to help out with difficulties.

External users, include staff in 120 academic departments, spread out all over campus. For them, going back to a paper-based system would significantly slow processes. From faculty members with Ph.D.s to entry-level administrative staff with GEDs, different staff members have varying duties and responsibilities besides graduate admissions -- everything from timetable to payroll to undergraduate record keeping to tracking of satisfactory progress. Many departments have limited or no technical support. They are comfortable being on the Web, and they need to see what the Graduate School Admissions Office is doing and what other departments are doing with an applicant.

User-Centered Approach

Fortunately Graduate School staff and academic departments had worked together in previous years to develop earlier systems. The Graduate School assigned an administrative staff member as the team leader to assemble a cross-functional team that included three Graduate School admissions personnel, four people from academic departments, and three software engineers (two from the Graduate School and one from the central computing unit).

During brainstorming sessions with the internal and external users, we identified several significant problems with the vendor-supplied system that would cause considerable delay in our admissions process. First, data entry and data viewing were spread over many menus and screens: Address and e-mail were on one screen, previous institutions on another, GRE and TOEFL scores on another, and so on. Therefore users had to execute a series of time-consuming steps of navigation to enter or retrieve information through various menus and screens. Second, there was no way to upload data from our existing Web-based online admissions application to the vendor-supplied system. We needed a solution that would let us continue to upload the data directly to the central system without re-keying. Third, the vendor-supplied system did not have a good way to track who entered which decision codes on which applicants. In addition, there was no way to add links to other admissions-related information on the Web. The e-mail address on the panel was not hyper-
linked. A user who wanted to e-mail an applicant had to cut and paste the e-mail address from the panel to the user’s e-mail program. Lastly, there was no easy way to generate for each academic program a sortable list of applicants and relevant information about them, so users could enter evaluation requests and decision codes one after another. In large academic departments with several hundred of applicants, users could not afford to go through a half of dozen menus and screens for each applicant to get the information they need and to enter evaluation requests and decision codes.

Although it was possible to customized screens to fill some of these gaps, such screens would have to be analyzed each time the vendor upgraded or even patched the system. In most cases, each screen would have to be reprogrammed entirely or partially. We needed a solution that allowed us to minimize the impact and work we needed to do when an upgrade or a patch occurred.

As a result of the brainstorming sessions, internal users, external users, and software engineers listed all gaps and decided which gaps must be filled in order to carry out the University’s research and educational missions. We decided three modules were critical to the admissions success: 1) Data-Upload module, 2) Scanner-Aided Data-Entry module, and 3) Graduate Web Information System (GWIS) module.

**Data-Upload Module**

The first module is the Data-Upload module. Since there was no way to upload data from the Web-based online admissions application to the vendor-supplied system, we developed a solution. Even after an upgrade or a patch, the upload requires minimal or no modification. The upload module works not only for applications received on the Web, but also for paper applications entered via the scanner-aided data-entry module. Because it uses the same vendor-supplied programming functions that the vendor’s data entry screens use, this upload module fills all fields in the central system properly.

**Scanner-Aided Data-Entry Module**

The second module is scanner-aided data-entry system to speed up the data-entry process of paper application forms. The scanner-aided system uses a high-speed scanner to capture the images of the application forms and displays them on the screen. Data-entry clerks do not have to look back and forth between paper and screen. The objectives are to increase performance with minimal physical and cognitive workload and to increase accuracy with built-in lookup and error checking functions. Several features were determined during the initial brainstorming sessions and sequent iterations in the design phase to speed up the data entry process. Data-
entry clerks enter only the necessary information assisted with intelligent defaults, auto fill, and auto completion. Additional features included the following:

- Auto flow and Auto track
- Elimination of code use
- Partial search from anywhere
- No mouse needed

The result was improved data entry for our internal users (Graduate School Admissions staff). Empirical comparisons showed that the time to enter an application was reduced from 6 minutes 40 seconds (vendor-supplied system) to 2 minute 20 second (scanner-aided module). This saved more than 900 labor hours per year, so that the Graduate School no longer needs to hire seasonal staff for data entry.

Graduate Web Information System (GWIS) Module

The third module is a Web-based interface for the Graduate School and the academic departments to retrieve applicant information and to enter evaluation requests and decision codes quickly and easily.

Because of the political climate, which discouraged customizations to the vendor-supplied system, and because the Web would allow us to present all necessary functions on screens everyone could learn to use in 15 minutes, we decided to build a Web application to allow the Graduate School Admissions staff and academic departmental staff to track and update the progress of an applicant. To develop this module, we worked with a cross section of external users. The group included new employees with a fresh viewpoint, users who were familiar with the old mainframe transactions, and departments with special needs. We had four representatives on the team that met throughout the process. Later we convened a focus group with an additional 12 representatives. Still later we invited all academic departmental staff to open houses, and we used feedback from these to refine the module. We let them know how much we valued their input by showing them quickly how what they wanted would work. We asked them to think about an ideal system, no matter how they were used to doing things in the past. As we refined the system we brought new functions to their desks, as they responded we refined the system further.

We first developed a single screen that listed each department’s applicants with relevant data about each applicant. Departments could immediately sort these lists by name, ID, domestic or international, or latest decision code. In addition, we developed the data sheet. Clicking an applicant’s name on the list brought up a data sheet suitable for printing for the faculty admissions committee. Relevant
information about an applicant was displayed on one screen, compared with half of
dozens screens in the vendor-supplied system. The e-mail link on the data sheet was
hyperlinked. Departments could immediately use this system from standard Web
browsers with minimal training. They could print the page for review by faculty
committees.

At first, academic departmental staff had to print the list, mark evaluation
requests, and decision codes on it, and send this paper list to the Graduate School.
Some months later, they could enter evaluation requests and decision codes on the
Web. When this was done, communication between the Graduate School
Admissions staff and departmental staff took under one day instead of more than
two days (time it took for paper to be sent via campus mail). The system totally
eliminated the need for paper. The system was simple enough to use that academic
departmental staff now teach each other. Printing, data entry, linking, and so on
work like everything on the Web.

We treat the system as a ‘work in progress’ to encourage the academic
departmental staff to think about how it works, and how to improve it. In this way
we improve the product continuously and the internal users, external users, and
software engineers have a sense of ownership.

Conclusions

In the process of writing these independent applications to supplement
PeopleSoft functionality on our campus, we were reminded that an iterative, user-
centered approach is essential for its success. All too often programmers and those
who use their programs end up in adversarial relationships. With GWIS, we develop
a healthy working relationship, which continue to produce far better products. Also
it makes work fun for internal and external users, and software engineers. The
presentation will be relevant to all who are involved in implementing vendor-
supplied information systems on their campuses and who want to ensure that their
customers have the tools to make these systems truly useful.