DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.033—Computer System Engineering   Handout 22—April 11, 1996

Design Project 2: Electronic Student Services

As you might have read in the Tech (V116.2, 2/9/96, "Re-engineering teams target student service processes"), MIT student services are next on the list for re-engineering. A recent management fad which is sweeping the nation's corporations as well as MIT, re-engineering aims to save money and improve services by simplifying administrative processes, like payroll, acquisitions, and purchase orders—or like adding and dropping classes, registering, and submitting grades. Re-engineering rethinks a process from the ground up, making it serve the community faster, easier, and better.

It sounds like a tall order, so who better to tackle it then 6.033 students? The MIT Financial and Academic Services Transition Team (the FAST team) is in charge of re-engineering student services, and they need your help. Specifically, they want you to work on the services listed below. Currently they're taken care of by a big mess of different forms, involving many different campus offices; the FAST team wants you to rethink them and integrate them into the MIT computing environment.

1. Class schedules. Students and their advisors should be able to look at students' schedules on line.
2. Add/drop.
3. Lotteries (for HASS-Ds and PEs).
4. Updating address and phone information.
5. Preregistration.
6. Registration.
7. Grades. Faculty should be able to submit grades; students should be able to get them.
8. Degree audits.
9. Requests for transcripts and certification-of-student-status.

Students and faculty currently find that they spend way too much time filling out these forms, especially on and around registration day. The FAST team wants to make them quicker and easier to fill out, but make them even more secure and reliable than they are now. Here are the goals they list for student service re-engineering:

1. Improve services to students by simplifying administrative processes, reducing unnecessary steps and allowing more time to be devoted to student-valued services;
2. Simplify processes and better meet faculty needs (e.g., improved access to information, reduced paper work) so that faculty can better assist, advise, and interact with students;
3. Reduce processing costs so that funds can be reinvested in key areas;
4. Become the leader, in terms of cost, quality, timeliness, and service, in providing administrative services to students.

The FAST team has finished a preliminary design, but the design isn't really fleshed out, and implementation work isn't scheduled to begin until after May. The team is very interested in your projects! We plan to forward the best ones to them. In the past, 6.033 design projects have made impact on the MIT community, and your design project could have direct influence on how these processes will be automated!

Your team's assignment

First, this is a team project, so form a team of 3 people where everyone has the same recitation instructor. Please email your recitation instructor before Thursday, April 18 with the names and email addresses of your team members. Then get to work! **Your report is due in recitation on Thursday, May 2.**

Your team's job is to write a 10 to 12-page report that describes the design of a system that supports electronic student services, specifically the 9 services listed above. You must analyze the services that exist now and decide what it means to implement them electronically, and how you can do it. All the services must be provided in a secure and reliable way while maintaining users' privacy.

Describe the protocols used by your system, the means of privacy and authentication, and the user interface(s) your system presents. Explain how your design addresses reliability issues (high availability of service, durability and integrity of data, etc.). Evaluate your design from a technical standpoint. You should also assess the social impact of your design.

Phrase the goals and design of your team's project in 6.033 terminology and use 6.033 design approaches and techniques that were discussed in lectures and recitations.

Your intended audience is Professor Marty Schlecht, the head of the FAST team. You may assume he has taken 6.033 in the past, but is not completely up to date on the developments of the last 5 years. In addition, he knows all the high-level goals of the project, but probably hasn't had the time to study all the technical and social aspects of reengineering student services. It is your team's job to give a him a good, coherent, self-contained, well-written proposal for a design, including an evaluation of the benefits and disadvantages.

It is crucial that you provide enough detail for Professor Schlecht to evaluate the real-world feasibility of your design. To this end, your report should include at least one specific example: Describe exactly what happens when a student adds a class (which currently requires signatures from the student, her advisor, and the instructor of the class she is adding) using your system.

Some issues

You might want to think about these issues while developing your design. This is by no means an exhaustive list; it will just get you started. Realize that some of these problems may be better addressed by non-electronic means: don't design only the computerized part of your system!

1. What exactly do the existing forms do? What problems do they solve?
2. How should the FAST team judge the success of your design?
3. Who are the system's users? Do they have conflicting needs?
4. How well does your system scale? Can it handle Reg Day?
5. How will you authenticate users and maintain privacy? Security and privacy must be absolute for this system, as grades, transcripts, and the like are very sensitive information.
6. What kind of records will be kept, and for how long? Academic transcripts, for example, are generally kept for many years.

7. How fault-tolerant is the system? How do you account for network failures? Client failures? Server failures? Campuswide power failures? Unexpected user inputs, or other human "failures"? Unexpected failures of your design?
8. How does the system recover from failures? Can a user initiate recovery procedures, or can it only be done by the system?
9. When can a form or form equivalent be lost in your system? How will a user discover this has happened? What will the user do then? (What will a student do when his submitted-at-the-last-minute drop card is lost by the system?)
10. What is the social impact of your system? To the advisor/student relationship? To student life in general?
11. Will there be different user interfaces for different users?
12. Who is ultimately responsible for the system? Does anyone have ultimate control over the system and its contents (like a superuser does in the workstation world)? If so, who?
13. How will you make sure that your system continues to work, with minimal changes, when major Institute policy shifts (like intermediate grades, class-specific drop dates, or advisor-specific add and drop policies) are implemented?

Some pointers and suggestions

The FAST team's design suggestions, and what they've learned about student services, are available online at http://web.mit.edu/studentserve/www. Most of the documents available there are in the form of very high-level overheads in JPEG format. We don't suggest that you spend too much time concentrating on the slides. They are very far from an implementation. Also, feel free to change their design however you see fit.

Several of these services are available on-line now. The Student Information System (or SIS; "add sis; sis" on Athena) allows access to students' class schedules and grades, lets students change their address and phone information, and provides a primitive degree audit. (SIS uses a second Kerberos password to further protect grade information, since people often share their normal passwords; the method SIS uses to ensure the initial security of this second password is worth looking at.) The Registrar's Office online (http://registrar.mit.edu/) has some other services, including another, friendlier class schedule interface (although it doesn't actually access your real schedule data, you have to provide it yourself). Use these systems to think about different ways to provide information to the user. You don't have to stay compatible with either of them.

Likewise, don't just implement the paper forms that exist now. Think through the processes *behind* the forms. Maybe you can simplify several forms into one, or get rid of the need for one of them—or all of them!—altogether.

Make sure you understand the real reason for each form. For instance, you might think that forcing your advisor to sign the Add/Drop form is a waste of time. But there's a good reason for this: it forces you to meet with your advisor before making big decisions. Maybe you can figure out a better way to do this, or argue why it's not necessary.

You may find it helpful to separate the design into two parts: the design of a lower-level, generic system which could support all the specific services we require, and designing the specific services using that generic system. Such a generic system would be more generally useful as well.

Some of the more important slides from the FAST team are included with this handout, along with the text of a recent Tech Talk article (V40.25, 4/3/96) on student service re-engineering.