DESIGN AND IMPLEMENTATION OF A DEPARTMENTAL INFORMATION MANAGEMENT SYSTEM

A Thesis

by

DAWEN XIE

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2007

Major Subject: Computer Engineering

DESIGN AND IMPLEMENTATION OF A DEPARTMENTAL INFORMATION MANAGEMENT SYSTEM

A Thesis

by

DAWEN XIE

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Nancy M. Amato
Committee Members,	Marvin L. Adams
	Jennifer L. Welch
Head of Department,	Valerie E. Taylor

August 2007

Major Subject: Computer Engineering

ABSTRACT

Design and Implementation of a Departmental Information Management System.

(August 2007)

Dawen Xie, B.S., Nankai University, P.R.China

Chair of Advisory Committee: Dr. Nancy M. Amato

A major challenge faced by an academic department is the effective management of the large amounts and different types of data that are encountered in its day-to-day operation, ranging from personal data to varied types of documents. This data has various access privileges and restrictions and will be provided by numerous sources: by the individuals themselves (e.g., a current student or an applicant who submits a resume), by a member of the faculty or departmental administration (e.g., an advisor who provides a meeting note or a faculty member who provides a review of a faculty candidate), or by an external party (e.g., transcripts or test scores for an applicant to the graduate program). Statistics regarding this information are needed by many different constituents, both inside and outside the department. Traditional paper-based operation is not only costly but also ineffective.

In this work, we design and implement an information management system called *One Stop Information Source (OSIS)* to effectively manage departmental information. We also provide a detailed case study of the *Financial Support Monitor (FSM)*, which is a tool to monitor student financial support history and to support different departmental procedures.

OSIS has made a big impact on the department's day-to-day operation and receives recognition from different types of users. It improves the efficiency of many operations. The difference in processing time is significant.

To my wife and my son.

ACKNOWLEDGMENTS

I want to thank my advisor, Dr. Nancy Amato, for her guidance and support in the last few years. She was gracious to provide me with the opportunity to join her research group when I was preparing for GRE and TOEFL. Her passion on research inspired me to pursue a PhD program. I benefited tremendously when I was in the PhD program. In November 2005, Dr. Amato asked me to work on a departmental information system. It turns out to be a win-win solution: We achieved big success in OSIS, and I finally discovered that I have more passion for system development than research. Dr. Amato was very supportive of my transition from the PhD program to the Master program. I appreciate her support for me to become who I am and in what I am doing now.

I also want to thank Dr. Marvin Adams and Dr. Jennifer Welch, two great professors I had the privilege to know, for the comments and feedback that enriched this thesis.

Members in Dr. Amato's research group helped me with the research and different projects on which I worked. Members in the OSIS development team share the same vision I do, and they put into great effort into the project.

I would like to thank my parents who support me with all their hearts. Finally, I thank my wife for being my cheer leader and always being there for me.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I
INTRODUCTION

A major challenge faced by an academic department is the effective management of large amounts and various types of data that are encountered in its day-to-day operation, ranging from personal data to varied types of documents. The data have various access privileges and restrictions and will be input by a variety of sources, ranging from individuals themselves (e.g., a current student or applicant submits a resume), to a member of the faculty to a member of the departmental administration (e.g., an advisor provides notes about a meeting or a faculty member provides a review of a faculty candidate), to automatic uploading of transcripts or test scores for an applicant. Automatically generated statistics regarding this information will be used by many different constituents, both inside and outside the department. Traditional paper-based operation is not only costly but also ineffective.

We designed and implemented an information management system, called *One Stop Information Source (OSIS)*, to effectively manage departmental information. The design of the overall system follows the guidelines from the design science for information system [11] and benefits from the study in Computer-Supported Co-operative Work (CSCW) [9]. To increase the usability of our system, we followed Computer-Human Interaction [2] and Iterative Design guidelines [8].

OSIS includes four main components. Figure 1 shows the functional diagram of the system. The following is a brief description of each component:

- OSIS-grad – a component used to manage information and procedures related to graduate programs and students.

This dissertation follows the style of *IEEE Transactions on Automatic Conrol*.

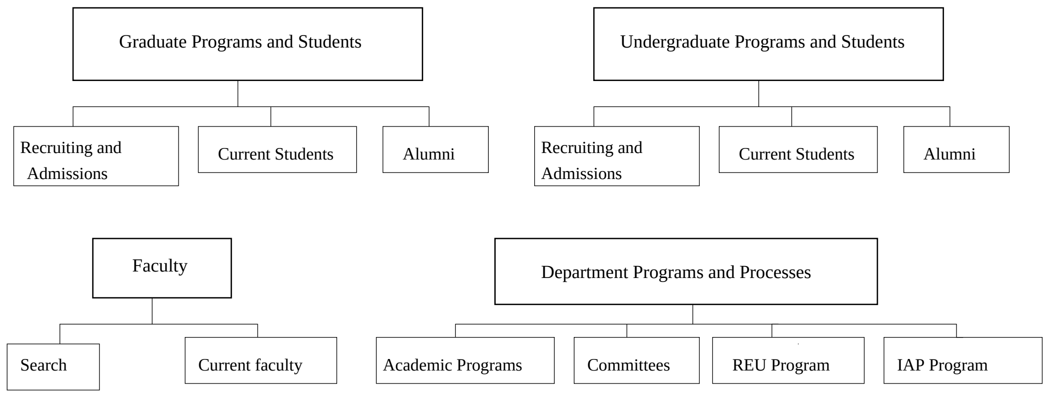
Graduate Programs and Students
Recruiting and Admissions
Current Students
Alumni
Undergraduate Programs and Students
Recruiting and Admissions
Current Students
Alumni
Faculty
Search
Current faculty
Department Programs and Processes
Academic Programs
Committees
REU Program
IAP Program

Fig. 1. Function diagram for OSIS.

- OSIS-ugrad – a parallel component to OSIS-grad for undergraduate programs and students.
- OSIS-faculty – a component used to manage information related to faculty.
- OSIS-dept – a component used to manage different departmental programs and procedures.

While we can think of OSIS having OSIS-grad, OSIS-ugrad (collectively called OSIS-student), OSIS-faculty and OSIS-dept components, most applications involve multiple components often with different interfaces.

In this work, we also provide a detailed case study of the *Financial Support Monitor (FSM)*, which is a tool to monitor student financial support history and to support different departmental procedures. The FSM interacts with all OSIS components: students apply for financial support (OSIS-student), faculties evaluate and fund some students(OSIS-faculty), and departmental accounting and advising deal with administration (OSIS-dept). One can view the financial support history for a particular student (OSIS-student), manage which students were supported on a faculty member's grant for a particular semester (OSIS-faculty), determine the number of PhD students in the department that were supported by research assistantships in a particular academic year (OSIS-dept), or assist the accounting staff in managing student payroll (OSIS-dept).

The following applications are currently in production use, and they were actively used by over 9,000 users in the first year.

- OSIS-grad – much of the planned functionality for this component is already in production use. It includes graduate admissions (on-line submission of application materials by applicants and references, on-line review by the faculty, and on-line decision by the admission committee); current graduate student information management (degree progress monitor, test scores storing (GRE, TOEFL, ELPE), annual PhD review and improvement and dismissal process, GAT/GANT/Scholarship application, review and assignment, enrollment history); and former graduate student information management (listing of degree, graduate year, advisor).
- OSIS-faculty – on-line submission of application materials by applicants and references, on-line review by the faculty, some support for the interviewing process.
- OSIS-dept – Research Experience for Undergraduate (REU) program (on-line submission of application materials by applicants and references and review by the faculty) and academic programs administration (semester course planning and scheduling support, instructor and TA assignments, room assignments, etc.)

A. Our Contribution

The main contribution of this thesis work is the design and implementation of a departmental information management system that significantly improved the efficiency of the department's daily operation. In the first year of production use, over 9,000 users have used our system. They include faculty members, students, staff members, various applicants and their references.

OSIS significantly improves the efficiency for many daily departmental operations. For example, the traditional paper-based graduate program application involves a sequence of procedures: paper submission of all materials to be sorted and collated into paper-based folders, which are passed sequentially from admissions chair to faculty members to review. The average processing time is several months. Identifying high priority applicants is difficult and often results in losing good ones. With the help of OSIS, we are able to electronically submit almost all application materials from applicants, references, admission related documents from Office of Admissions and Records (OAR), and perform automatic importing of test scores, undergraduate institution, etc. from Student Information System (SIMS). As a result, an application can be reviewed by all faculty members immediately and decisions can be made in days, making it easy to identify excellent applicants.

We can now do many things with OSIS that we could not do before. First, OSIS makes information sharing possible and convenient, e.g., in graduate admission, all faculty members are able to view any application and make comments; all comments are accessible by other faculty members. Second, generating statistics is very easy in OSIS while it required much bookkeeping and manual processing before.

In OSIS, we design and implement a set of web utility classes (See Chapter IV Section D.2 for details). These classes are general, self-contained, and greatly increase the re-usability and maintainability of our code. By building on top of the web utility classes, the development of OSIS application becomes relatively easy. Furthermore, the web utility classes can be used for other web applications. We plan to make them open source software in the future.

Although OSIS is a customized information management system for the department of computer science at Texas A&M University (TAMU), it can easily be configured to be used by other academic departments. We can extend OSIS to support much larger user group – an academic college or even the whole university. In fact, the OAR at TAMU is interested in our graduate admission system, and we are looking into options to share parts of the OSIS functionality.

B. Outline of Thesis

Chapter II describes related work from different fields. Chapter III presents the definition of requirements and functionality of our system. Chapter IV presents the design of our system. Chapter V provides a detailed case study of the FSM. Chapter VI describes the results and implementation status for our system. Chapter VII presents the conclusions.

CHAPTER II
PRELIMINARIES AND RELATED WORK

The main objective of OSIS is to improve the effectiveness and efficiency of the departmental administration's work through a paperless interface to facilitate information transition, processing, and storage. No current literature discusses this exact implementation; however, a growing amount of interdisciplinary work explores the similar goal: answering how the activities of complex systems can be coordinated [12]. Some work focused on coordination in parallel and distributed computer systems while others on coordination in human systems [12]. However, the common theme is that the targeted complex systems usually include both people and computers, our project shares the same characteristic.

This chapter will present some related work in four different disciplines and piece together the theory background of our work.

A. Information System

According to Hevner in [11], Information Systems (IS) are "implemented within an organization for the purpose of improving the effectiveness and efficiency of that organization" [11]. Two paradigms that characterize much of the research in the IS discipline are behavioral science and design science. The behavioral science paradigm seeks to develop and verify theories that explain or predict human or organizational behavior. The design science paradigm seeks to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts [11].

These two paradigms are complementary but distinct [13]. The behavioral science paradigm has its root in natural science research methods. It seeks to develop and justify theories that explain or predict organizational and human phenomena [11]. The design science paradigm has its roots in engineering and the science of the artificial [19]. It is fundamentally a problem solving paradigm and seeks to create innovations that define the ideas, practices, technical capabilities, and products through which the analysis, design, implementation, management, and use of information systems can be effectively and efficiently accomplished [6].

Our work obviously falls in the realm of the design science due to the problem solving nature. As the IS literature recognizes, while the importance of design is well recognized, designing a useful system is complex. We built on the work of the design science paradigm and followed the literature suggested guidelines in [11].

B. Computer-Supported Cooperative Work

Recently, a great deal of interest has been placed on designing computer tools to help people work together more effectively. This new field designs systems called Computer-Supported Cooperative Work (CSCW) [9, 10] or groupware [16]. According to Ellis et al. in [7], "the work in CSCW usually looks at how groups work and seeks to discover how technology (especially computers) can assist them." Groupware is the technology used to support collaborative work [16]. According to Ellis et al. in [7], groupware consists of "computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment." CSCW stresses the outcome or product while groupware is the enabling technology.

In our work, we studied closely how different components of the department administration function and designed a system that helps the complex coordination. Therefore, the work in CSCW can provide a lot of insights for our work.

C. Usability

Usability is one of the focuses of the field of Computer-Human Interaction [2]. As the name suggests, usability has to do with bridging the gap between people and machines. For a tool to be effective, it must allow intended users to accomplish their tasks in the best way possible. Usability depends on a number of factors including how well the functionality fits users' needs, how well the flow through the application fits users' tasks, and how well the response of the application fits users' expectations. According to Nielsen in [15], usability has multiple components and is traditionally associated with five attributes. First, Learnability: the system should be easy to learn so that the user can rapidly start getting some work done with the system. Second, Efficiency: the system should be efficient to use, so that once the user has learned the system, a high level of productivity is possible. Third, Memorability: the system should be easy to remember. Fourth, Errors: the system should have a low error rate, so that users make few errors during the use of the system. Fifth, Satisfaction: the system should be pleasant to use [15].

The key principle for maximizing usability is to employ an iterative design, which progressively refines the design through evaluation from the early stages of design [8]. The evaluation steps enable the designers and developers to incorporate users' and clients' feedback until the system reaches an acceptable level of usability.

The preferred method for ensuring usability is to test actual users on a working system. Achieving a high level of usability requires focusing design efforts on the intended end-user of the system.

Our work provides various types of web interface for users and the usability of our web interface results in the effectiveness of our system. We included usability early in the design/development process and included users in the design process to improve the usability attributes described in [15]. We also allowed usability and users' needs to drive our design and test our system frequently throughout the design process.

D. Database Design

Our work involves massive data management which requires a sophisticated database design. Database design is defined as: "the logical and physical structure of one or more databases to accommodate the information needs of the users in an organization for a defined set of applications" [5].

According to Date in [5], the design process follows several steps: 1. planning and analysis; 2. conceptual design; 3. choice of Database Management System (DBMS); 4. logical design; 5. physical design and 6. implementation. The following section briefly walks through each step [5].

The goals of the requirements analysis (Step 1) are to determine the data requirements of the database in terms of primitive objects; to classify and describe the information about these objects; and to identify and classify the relationships among the objects [5].

A data model is a conceptual representation of the data structures that are required by a database [4]. The data model is one part of the conceptual design process (Step 2). The Entity-Relationship (ER) [3] model is widely used for the data modeling. A basic component of the ER model is the Entity-Relationship (ER) diagram which is used to visually represent data objects. In the ER model, "Entities" are the principal data objects about which information is to be collected [3]. Entities are usually recognizable concepts, either concrete or abstract, such as people, places, things, or events which have relevance to the database. Some specific examples of entities are student, degree program, document. A "Relationship" represents an association between two or more entities [3]. An example of a relationship would be: students are enrolled in a degree program, a degree program involves different documents.

A Database Management System (DBMS) is computer software designed for the purpose of managing databases. Typical examples of DBMSs include Oracle, Microsoft SQL Server, and MySQL. Step 3 involves making choices of DBMS.

In logical design step (Step 4), the ER diagram is converted into tables in the database. Physical database design (Step 5) consists of executing tables in an actual DBMS software file. Database system implementation (Step 6) consists of entering data and maintaining the DBMS.

E. Summary

To build a user friendly, reliable, flexible, and effective paperless departmental information management system, we need insights from different disciplines. This chapter presents the key concepts and review of the literature. Although it is only a snap shot, it serves the purpose of laying out the theory background of our work.

CHAPTER III
REQUIREMENTS

Different components of OSIS share common functionality in different ways. Some components are very similar to each other. For example, OSIS-grad and OSIS-ugrad are parallel components that share significant functionality. Some common functionality is required in all components. For example, all components need functionality to upload and manage document. Some similar functionality is used in all applications, but they differ from each other slightly. For example, all on-line applications involve some kind of review process, although they may not be exactly the same. In the following, we provide an overview of the requirements and functionality for each component.

A. Students (OSIS-grad and OSIS-ugrad)

OSIS-grad and OSIS-ugrad are parallel components that share a large amount of functionality. In this section, we first introduce the common functionality for both components and then discuss the component specific functionality.

Academic program information is essential for an academic department. Providing tools for students, faculty, and staff to easily access/manage the various types of academic information is important. OSIS-grad and OSIS-ugrad provide tracking for academic information throughout students' stay in the program, e.g., students' basic information, test scores and various types of degree program related information. Data are mainly input and maintained by the advising office, but students are allowed to input and/or edit certain data. Most data are available (accessible) to student and all data are available to all faculty members. In this application, we plan to support the following functionality for current students:

- Student basic information: Name, contact information, gender, citizenship, ethnicity, U.S. residency, photo, etc.
- Resume: Used in financial support application evaluation and other programs. The student will be able to upload/update the resume and to specify how it can be used (restricted to the department, viewable to the Industrial Affiliates Program (IAP), etc.)
- Degree progress monitor: Provides information about milestones (degree plan, prelim exam, final exam, etc.), enrollment, academic performance (GPR).
- Financial support application, history, selection, and evaluation (GAT/GANT, student worker, scholarship)
- Email support: Listserv management and other mailer tools
- Statistics and report generation (admission, graduation, enrollment).
- Letter request and generation (good standing, travel, etc.)

Maintaining contact information and tracking the accomplishments of former students are important aspects of an academic department. We plan to support the following functionality for alumni:

- List of alumni to be used to generate list on public web pages and statistics accessible on internal web pages.
- Tracking of alumni (where they went and how they are doing) to build a network of former students to assist current students and for development purposes.
- Alumni events, newsletters, and other services to be provided for the department's alumni.

1. OSIS-grad Specific Functionality

OSIS-grad supports or will support three major components that are not needed for OSIS-ugrad: graduate admissions, the annual PhD student review, and petitions for degree level change (downgrade or upgrade).

a. Graduate Admissions

Graduate admissions is a very important factor that affects the quality of a graduate program. Thus, having an effective way to process graduate admissions is crucial. Traditionally, applicants and their references send all application related documents in hard copy to the department. It takes a significant amount of time and resources to process these documents. From the department's point of view, various types of documents need to be processed manually and then sent to faculty members for review. Since it involves a sequence of steps, the entire procedure is very time consuming and inefficient. To address this problem, OSIS-grad provides an on-line system for graduate admissions. The system allows applicants and their references to be able to use electronic submission for all application related materials; faculty members in the department to view all applications and the supporting materials and provide reviews on-line; the admissions committee to make decisions that can be made available to both applicants and all faculty members (at different times if desired). In this application, we support the following functionality:

- on-line submission of application materials by applicant and references,
- on-line review by faculty members,
- on-line decisions by the admission committee,
- automatic data import from external sources such as the office of admissions and records, and/or the university information management system (SIMS),

and

- support for advising e.g., to send email to applicants regarding their decision and to manage other application related information.

b. Annual PhD Review

OSIS-grad supports the annual PhD review procedure. It is in many ways similar to graduate admissions in that the student submits materials, others can also provide materials (reports by the advisor), another set of people review the application (the whole faculty or the graduate advisory committee), and make a decision (a rating for the PhD review or a decision on the degree change petition). Indeed, annual PhD review and many other applications have this need for the evaluation and review of ‘applications’. We have chosen to support this need by providing a review module that can be utilized in all applications.

c. Petition for Degree Level Change

Continuing on for a PhD after the Master’s or changing degree programs requires a petition for current graduate students in the department. It also involves review by departmental committees followed by a decision notification. Having an easy way for petition and review is the goal of this component. Depending on what is desired, there are differences in the required supporting documentation and in how the petition is processed. In all cases, the petition and all the required supporting material should be submitted by the student. The departmental committee will review the petition. In this application, we plan to support the following functionality:

- on-line submission for petition and supporting documentation,
- on-line submission for corresponding parties (student’s references able to submit recommendation letter), and

- on-line review system for departmental committee.

2. OSIS-ugrad Specific Functionality

In addition to the common functionality, OSIS-ugrad supports or will support the following functionality:

- 285/485, 291/491 requests. Undergraduates that wish to take independent study or research courses with professors need to submit proposals that are reviewed by the undergraduate advisors and/or undergraduate curriculum committee. OSIS-ugrad will support this procedure by accepting student submissions, and supporting the review and decision process.
- Fast Track Program. In this program, qualified undergraduates are able to take some graduate courses and receive credit for a paired undergraduate course, that is they can apply the undergraduate course to their undergraduate degree and the graduate course to their graduate degree. OSIS-ugrad can be used to monitor students taking advantage of this program and to identify students that should be encouraged to do so.
- Invited Graduate Admissions. In this program, the department offers 'free' admission to the graduate program to a select few top undergraduates. OSIS-ugrad will support this program by providing for the nomination, review, and selection of students.
- Peer teacher program. This program provides for excellent undergraduates to be selected to serve as a peer teachers for courses in which they have excelled. They do not do any grading but can help with questions and provide one-on-one tutoring. OSIS-ugrad will support the application, selection, and evaluation of peer-teachers.

B. Faculty (OSIS-faculty)

OSIS-faculty supports or will support procedures and processes related primarily to faculty, such as faculty search, faculty progress report submission and archiving, payroll requests and tracking (part of the FSM), statistics generation and data archiving, and support for the promotion and tenure committee.

The faculty search application provides an on-line system for tenure-track faculty search. Much of the functionality in this application is similar to those in the graduate admissions application. Additionally, the faculty search application will provide for interview support (job talk abstract submission by the candidate, scheduling support for departmental staff and faculty, travel planning support for departmental staff, post-interview faculty survey, etc.)

OSIS-faculty will also support a number of departmental procedures where faculty need to support materials for evaluation by departmental committees. For example, the submission of annual progress reports by all faculty (to be evaluated by the review committee, consisting of the department head, the associate department head, and the Promotion and Tenure (P&T) committee chair), and promotion and tenure materials (to be evaluated by the P&T committee). In both cases, at least initially, OSIS-faculty will only handle material submission by faculty – the reviewing and evaluation will be handled outside OSIS.

C. Department Programs and Processes (OSIS-dept)

OSIS-dept supports or will support departmental procedures and processes, such as course planning and scheduling, course requests, committees, the IAP, the REU program, awards tracking, and technical report tracking, etc.

1. Academic Programs and Departmental Committees

- Course planning and scheduling. Semester by semester course planning and scheduling support (course offerings, instructor and TA assignments, room assignments, etc.).
- Course Requests. Submissions, reviewing, archiving of course requests, both for permanent courses and courses that must be approved every semester such as CPSC 489/689, 291/491, 285/485.
- Committees. Provide portal for committees to view and archive work related to their committee with respect to the ability to restrict access to committee members, etc. Some committees for which portals are planned for include: P&T, Undergraduate Curriculum Committee (UGCC), Graduate Admissions & Awards Committee (GAAC), Graduate Advisory Committee (GAC), Graduate Assistantship and Scholarship Selection Committee (GASSC), Awards Committee (faculty, staff, graduate, undergraduate).

2. Research Experiences for Undergraduates (REU) Program

The REU program provides a 10-week summer research experience during which participants work closely with faculty members, graduate students and undergraduate mentors on current research projects. The program is open to undergraduate students from all colleges and universities. In this application, we support on-line application materials by students and references and on-line review by CS@TAMU faculty. We also plan to provide support for program administration. For example, mentor assignments, on-line submission and archiving of materials (research plans, reports, etc.), dynamic web page generation for program participants.

3. Industrial Affiliates Program

The IAP was developed to improve the computer science department's relationships with industry. In this application, we plan to support a resume bank where IAP members will have access to current student resumes, member contact support (automated Listserv maintenance), scholarship awardee monitor (awardee history).

4. Awards Tracking

The department is a community of students, faculty, and staff in which different members receive various types of awards. It is essential that the department has a convenient way to track all awards received by its members. In this application, we plan to keep track of all awards associated with the department and provide various types of statistics.

5. Technical Report Tracking

Members within the department make technical reports for their research work. In each year, the department assigns sequential technical report numbers based on the report generation date. The process is inconvenient and sometime causes duplicate numbers. In this application, we will provide an on-line system for technical report number management. We also plan to support a centralized archive for technical report.

CHAPTER IV
DESIGN

Design is very important in the interactive systems and needs to be evaluated at each stage of development. We followed the Iterative Design approach [8] which states that once a component has been partially or completely implemented, we can receive feedback from users. Based on that feedback we can find out problems in the design of the interface or problems in the functionality of the system, and re-evaluate alternatives. Our design has to take into account the various type of data and users in different applications.

We split our design into the system architecture, the web interface, the database, and the code. For the design of the web interface, we applied User Centered Design [17]. Working closely with the final users allowed us to know their specific needs. For the design of the database, we used the Entity-Relationship (ER) [3] model for the data modeling. The object-oriented programming paradigm was used to design and implement the code. This paradigm will ease maintainability. Also, object-oriented programming offers a natural way of defining functions and data in terms of a class hierarchy.

In this section we first provide an overview of the system architecture. We then describe our design for the web interface, database and code.

A. System Architecture

A big challenge for OSIS is that it needs to support a large number of users whose use for the system varies greatly. This challenge is a key factor that affects our design of the system architecture and the authentication.

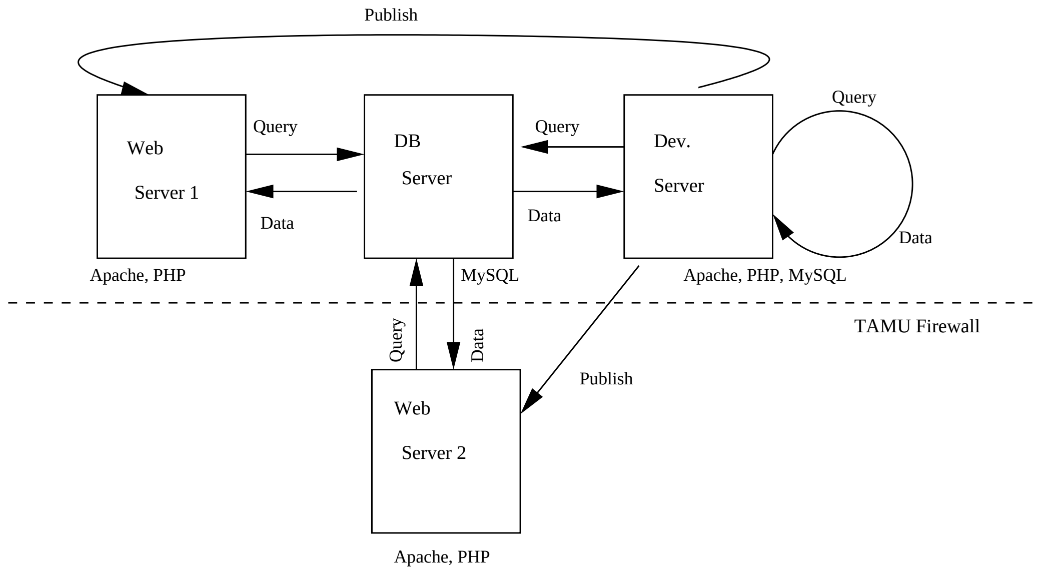

Fig. 2. System architecture.

The system architecture of OSIS is shown in Figure 2. User requests over the Internet are received by a web server, and the corresponding query is sent to a database server. The data are sent back to the web server and the resulting web page is generated and sent back to the user. Our system runs in Solaris with Apache [1] and PHP [18] for web serving, and MySQL [14] for database. All functionality is written in PHP with MySQL API.

In our current implementation, we use three production servers ("Web Server 1", "Web Server 2", and "DB server" in Figure 2) and one development server ("Dev. Server" in Figure 2). For the production servers, "Web Server 1" hosts applications for internal users to use different OSIS components; "Web Server 2" hosts applications for external users to apply to CS@TAMU programs (admissions, REU, faculty search, etc.); "DB Server" hosts the database for both web production servers. To increase the security of our system, we only put applications on the "Web Server 2", which stays outside the TAMU firewall when it is absolutely needed. The development server serves as the web server and the database server for development. It also provides tools to manage our “DB server” and to publish applications to different web servers.

B. Web Interface Design

Users are a central part in the design of web interface. In general, we tried to reduce the number of clicks the users have to make to get to the information they need. For all applications, we had different levels of user groups study before we released the applications. During this preliminary release, we got feedback from them regarding to the way the information was delivered, the ease of use, how intuitive the interface was, and what other options users would like OSIS to provide. We also discovered some user preferences which were easy to accommodate and greatly improves user’s efficiency in our system. We accept feedback from active users after the release the aim of which is to improve the user’s experience.

OSIS supported over 9,000 users in the first year. These users vary greatly in the way they interact with the system (See Chapter IV Section B.1 for more details) which affects the specification of how we deliver information through web interface to different users. For those people who are not members of our department but want to use a particular application, we try to make the access as easy as possible. For example, we provided a single page for references to provide recommendations and submit reference letters. For applicants, we provided navigation links to all information they need to access in the header of every page. When a user is a current member of our department, they normally have different reasons for using our system. Therefore, we provided a user friendly index page with all the options that are needed. We made the access to this index page easy to access by putting a link to it in the header of every page. For faculty and staff members, we also make it easy for them to search for specific information by providing user friendly searching interface.

As OSIS supports more and more applications, we also make interfaces for different applications as similar as possible. In this way, users do not have learning curve for each new application when we release it.

C. Database Design

We followed the outlines in Chapter II Section D for database design. In particular, we did a user group study to analyze system requirements before the design. We then use the ER model for the data modeling. In term of DBMS, we choose MySQL, which is one of the world's most popular open source databases. About 100 tables support the production use outlined in Chapter I (See Appendix B for database tables used in OSIS). Note that the web interface, which is implemented using PHP in our code, is the main way we provide for all users to manipulate the data in the database.

D. Code Design

To develop a system that can be easily maintained, we applied software engineering techniques. The Waterfall model [20] has been widely used to develop large systems. It starts with definition and analysis of requirements so that the system can be divided into different components. The next phase is the design of those components, followed by the implementation and evaluation of each component, and finally integrating them and testing the system as a whole.

Code in OSIS is written in PHP; we use the object-oriented programming paradigm. Code is the bridge between the web interface and the database. All web interfaces that users see and all underlying database operations are implemented in different classes. In our current production, we have implemented around 160 classes (around

86,000 lines of code).

1. Authentication

OSIS supports large numbers of users. These users vary greatly in the way they interact with the system. There are three main user groups who use OSIS.

- *Long Term Users* are the main user group for OSIS. Their usage is a few years. All of them have some association with the department, and they use OSIS on a regular basis. In particular, this group includes all faculty, staff and students within the department. The total number of users in this group is currently around 1,200.
- *Short Term Users:* are the most active user group for OSIS. Their usage is a few months. This group includes various types of applicants, graduate applicants, REU program applicants, or faculty candidates. In the first year, the total number of users in this group was approximate 2,800. Note that a portion of these users will transition to "long term user" once they affiliate with the department e.g., a graduate applicant becomes a graduate student.
- *One Time Users:* are the largest user group for OSIS, but their usage for OSIS is very limited – could be as little as one time or a few times. In particular, it includes references for different applications and non-TAMU persons who serve on a student's committee. In the first year, the total number of users in this group was approximately 5,500.

Two main requirements affect our design of the authentication for OSIS. First, we want it to be as secure as possible. Second, we want it to be as convenient as possible for the user. With a large number of users and a big difference between user groups, no single solution can meet all requirements. OSIS provides three types of authentication as described below.

- *Central Authentication System (CAS) authentication:* CAS authentication is provided by Texas A&M University and all users who have an association with the University are able to use this authentication. It meets both the security and the convenience requirements very well and is our first choice for authentication. In OSIS, all long term users and graduate applicants use CAS authentication.

  Code Example. The following explains how we use CAS authentication. The first step is to check to see if the user has logged in by calling the CAS server and passing the web service as a service parameter. The CAS server either redirects the user to the service page passing it a ticket or takes the user's browser to the login page and requests that the user log in using their NEO ID and password. In the second step, our web service page calls the validation service at CAS server and passes the web service and ticket returned from the login process. This second step is required so that the user does not get fooled by impostors granting invalid tickets and invalid authentications. If the service and ticket values are valid, the CAS server returns a string that contains user's unique information, that is the user's Universal Identification Number (UIN). We then use user's UIN as the key to identify who is logged in.

- *Email authentication:* Ideally, we would like to use CAS authentication as much as possible. Email authentication is used for those who do not have an association with the University and need to access our system on a regular basis. The user first needs to register using his/her email address and create his/her own password. The user is then able to use his/her email and password to access OSIS. An important requirement for this authentication method is the password resetting mechanism. OSIS handles password resetting in the following way: when password resetting request is made by the user, an email is sent to the user with a special URL. The user is able to access our system through the URL and modify the password. The URL expires when the password resetting operation is successful or the number of attempt exceeds the allowed number. In OSIS, all short term users except the graduate applicants use email authentication.

- *Secure ID Authentication*: Most one time users' usage of OSIS is very limited and their usage is requested by other OSIS users or some applications in OSIS. When these requests are made (a reference is specified by an applicant), a secure ID is created by the secure ID Authorization. In secure ID authentication, we follow the UUID [21] standard for creating universally unique random strings and uses them for secure ID. An email is sent to the applicant for verification, and an invitation email is sent to the applicant's reference with customizable instructions for how to provide the reference. Default instructions include a web-link to the reference page with the secure ID. Assuming the reference decides to participate, he/she will follow the web-link in the invitation email to the reference form. The secure ID provided in the web-link assures that authentication is preformed. A customizable reference form will be displayed. After the reference form is completed, a thank you email is sent to the reference for verification, and a notification email is sent to the applicant. The process is complete, but the reference is still allowed to login and correct any errors he/she may have made. Conformation emails will be sent if any modifications are made. In OSIS, all one time users use secure ID authentication.

2. Web Utility Classes

Different components of OSIS share common functionality in different ways. Some common functionality is required in all applications. For example, all applications need functionality to upload and manage documents. Some similar functionality is used in many applications but they differ from each other slightly. For example, all on-line applications (graduate admissions, faculty search, annual PhD review, and REU program application) involve some kind of review process although they may not be exactly the same.

To increase the re-usability and maintainability of our code, we introduced the idea of *web utility classes*. We designed and implemented around 40 utility classes (Figure 3). They are base classes for all components in OSIS. Building on top of the web utility classes, each component only needs to provide component specific functionality. As the base classes become more and more complete, building a new application become easier.

We partitioned these classes into four sets: authentication related classes, user information related classes, on-line application related classes, and tool related classes.

The first three sets of classes in Figure 3 share some common functionality.

- Web interface functionality includes the function to display various web interfaces for different group of users (an individual input form for data input, non-editable table for data display, summary information for a set of records, etc).
- Database management functionality includes all database operations (load data from the database, manipulate data in the database, search/sort data by different criteria, etc).

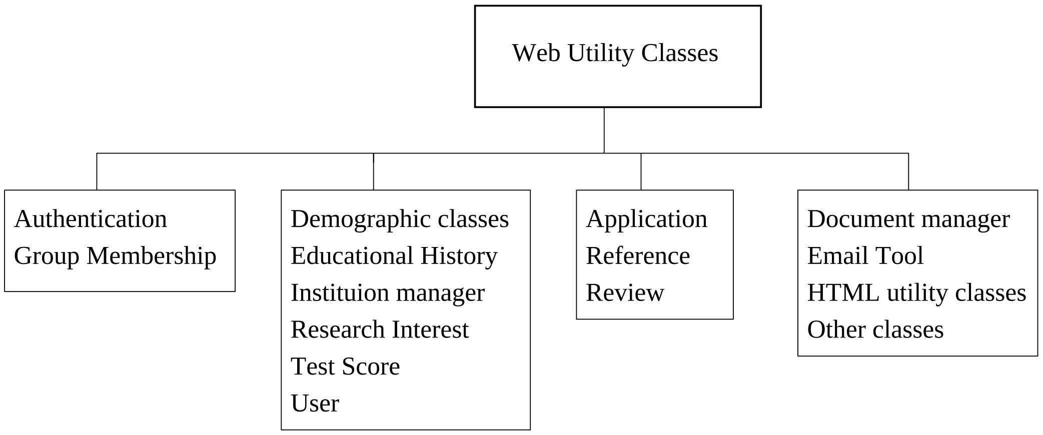

Fig. 3. Function diagram for web utility classes.

a. Authentication Related Classes

This set of classes provides support for authentication and management of user group membership information.

Authentication related classes provide support for email authentication and secure ID authentication. Both of them are derived from a base authentication class which provides log in and log out functionality.

A single user might have multiple roles at the same time for his/her usage of our system,: a faculty member may serve on different committees in different roles. The same user's role could change as time changes, – a graduate applicant becomes a student. We define *group* as a set of individuals identified by a common characteristic. We use *group membership* to store the user's group information over a period of time. The group membership class has functionality to check whether the user's group membership is valid or not.

Code Example. Figure 4 shows an example on how to provide customized page based on group membership. In OSIS, each application allows users with certain groups access. Once users are authenticated through our authentication system

```
 1 <?
 2 include("$utilsroot/groupsAdministrationClass.php");
 3 include("$gtsroot/includes/AuthSystem.php");
 4 $viewer = new AuthSystem($db,"neo",
             "$gtsroot/includes/WelcomePage.inc.php",
             "$gtsroot/includes/ErrorPage.inc.php",
             $neo_logout_redirect);
 5 if($viewer->GetGroupMembership()
       ->isActiveInGroup("Admin","OSIS")){
 6   include($gtsroot . '/ADMIN/includes/header.php');
 7 }else if($viewer->GetGroupMembership()
              ->isActiveInGroup("Faculty","OSIS")){
 8   include($gtsroot . '/includes/facultyHeader.php');
 9 }else if($viewer->GetGroupMembership()
             ->isActiveInGroup("Staff","OSIS")){
10   include($gtsroot . '/includes/staffHeader.php');
11 }else  if($viewer->IsCurrentStudent()) {
12   include($gtsroot ."/student/index.php");
13    exit();
14 }
15 ?>
```

Fig. 4. Pseudo-code description on how to provide customized page based on group membership.

(line 4 in Figure 4), the group membership class is called to determine users' group membership (line 5, 7, 9, 11 in Figure 4). Next, we provide users with a customized page with proper access level (line 6, 8, 10, 12 in Figure 4). One may have read access to all pages and have write access to a few pages. To prevent unauthorized access, we check the users' group membership in each individual page. In particular, we grant access to individual pages only when the users are in authorized groups. Figure 5, show show to allow administrator, faculty members and students to access a specific page.

```
<?
$canView=0;
if($viewer->IsCurrentStudent() || $viewer->IsAdmin()
    || $viewer->IsFaculty()) {
  $canView=1;
}
if($canView == 0){
  include($gtsroot.'/student/AccessDenied.php');
  exit(0);
}
else {
  ...
}
?>
```

Fig. 5. Pseudo-code description of how to restrict access to certain groups.

b. User Information Related Classes

This set of classes provides support for maintaining user related information. We provided class to manage information such as gender, citizenship, ethnicity, U.S. residency, etc. It also provides tools to manage a user's specific information, e.g., educational history, affiliated institution, research interests, and various test scores.

The base user class manages information that every user will have. For example, name, email, contact information, affiliation, gender, citizenship, ethnicity, and U.S. residency. It is built on the top of other classes in this set.

c. On-line Application Related Classes

This set of classes provides support for the general needs of different on-line applications (graduate admission, faculty search, REU program application, etc).

The base application class stores basic information for an application and provides submission interfaces for the applicants.

The purpose of the base reference is to provide a general framework that can be used to implement customized reference questionnaires. Also, base reference can be paired with the base application to provide the entire application and reference framework. The base reference class stores information for a reference. It also provides interfaces for applicants to request references be submitted electronically on their behalf and to generate authentication support for the reference requests.

The base review class allows the administrators for a particular application to assign reviewers. It also allows faculty members to volunteer themselves as a reviewer and recommend other faculty members as reviewers. All reviews are available to the relevant parties.

Code Example. The base review class includes three main functionalities. First, for any application, it allows application administrator to assign an individual application to different reviewers. Second, it provides interface for individual reviewers to input their review (a numerical score and some comments) and to view other reviewers' review on the same application. Third, it provides statistics on each application's review, – how many reviews are done and what is the average score.

These functionalities meet the need for some applications and can be used directly. For example, in faculty search review and REU application review, we use all functionality from the base review class.

The graduate admission review is derived from the base review class with some extension. For example, it provides interface for individual reviewers to make comments on whether they want to provide research assistantship for the applicant or to volunteer themselves to be the contact person.

Annual PhD student review is the most complicated review we have supported so far. First, it includes annual review process for all PhD students, improvement process, and dismissal process for some PhD students. Second, it involves different types of reviews – review by individual faculty member, by the mentor, and by the department. Third, it needs to support two types of review comments: comments for students and confidential comments for faculty. Finally, it involves different types for time/deadline setting: open date and deadline for students to input their reports, deadline for faculty members to input their review, and the time to release the reviews to the students. Our current implementation addresses this by creating a set of classes with proper class hierarchy (Figure on page 57, Appendix B). The extensive review class is a web utility class and is derived from the base review class with additional support for different types of comments. The PhD review class is derived from the extensive review with additional support for various types of time/deadline setting. Other review related classes are used to provide support for different types of reviews in different processes.

d. Tool Related Classes

This set of classes provides support for different functionality needed in web interfaces. For example, the document manager provides support for document uploading and management for documents: edition on associated information and deletion on document itself. The email tool allows users to send emails to a particular user or a set of users automatically. It also provides a monitor for all emails sent by the system. HTML utility classes support various HTML input (text input box, drop down menu, etc).

e. Summary

These utility classes are general and can be directly used or customized for a particular application. Building on top of the web utility classes, the development for each application becomes easier since we only need to add in application specific functionality. For example, OSIS-grad uses almost all the classes from utility classes.

CHAPTER V
DETAILED CASE STUDY: FINANCIAL SUPPORT MONITOR (FSM)

Students receive various types of financial support throughout their degree programs. A department must have a convenient way to know how its students are supported. For example, a grant manager needs to know which students are supported on a specific grant because the grant is about to expire. When the funding source comes from department or individual faculty member, faculty and staff must monitor different procedures, such as putting all teaching assistants on the payroll for a certain semester or monitoring their tuition waivers.

In OSIS, we designed and implemented an application called Financial Support Monitor (FSM), which is used to manage all information related to student financial support. It is used by students, faculty members, and staff members in the department, and interacts with all OSIS components: OSIS-students, OSIS-faculty, and OSIS-dept.

In this chapter, we provide a detailed case study of FSM, including both the design and implementation.

A. Analysis of Requirements and Functionality

As the Waterfall model [20] points out, the design starts with the definition and analysis requirements so that the system can be divided into different components. To study the system requirement, we have had multiple meetings with advising office, accounting personnel, and faculty to know how they manage the financial support within the department. We followed the Iterative Design approach [8] and have had several initial releases with our users to receive feedback and refine our design. As a

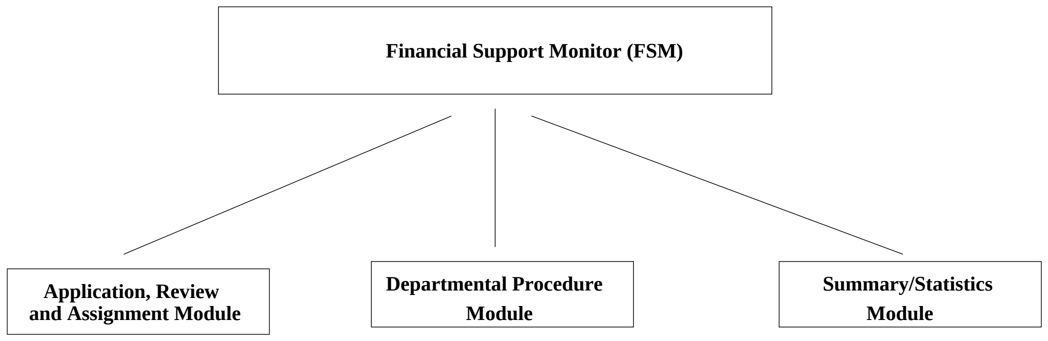
Financial Support Monitor (FSM)
Application, Review and Assignment Module
Departmental Procedure Module
Summary/Statistics Module

Fig. 6. Modules for Financial Support Monitor (FSM).

result, we partitioned the requirements into three modules (Figure 6).

1. Application, Review and Assignment Module

This module processes application, review, and assignment for the financial support funded and/or awarded by the department. It includes three main functionalities. First, it allows both current and incoming students to apply for both teaching and non-teaching assistantships, fellowships, scholarships, and student worker positions. Second, it allows faculty members to review student applications. Third, it allows the department to make decisions regarding financial support applications and make duty assignments for the awarded ones.

2. Departmental Procedure Module

This module provides interfaces for students, faculty members, and staff members to handle all departmental procedures related to a student's financial support. It includes three main functionalities: request new financial support, advising procedure, and accounting procedure.

- *Request New Financial Support* is used to request a new financial support. When the departmental decision regarding an assistantship or scholarship application is made, a corresponding financial support request is automatically generated. When the support comes from an individual faculty member, faculty and staff are able to request a new financial support. This module also can be used as a way to input students' past financial support or external financial support.

- *Advising Procedure* is used after a new financial support is requested. In particular, advising needs to generate an official offer letter and make sure it is signed by the relevant parties. The signed offer letter is stored in the system and available for all relevant parties: the student, the supervisor, and various staff members.
- *Accounting Procedure* needs to allow accounting personnel to place the student on payroll and verify the financial account information for their stipends, tuition, and fees. In each semester, accounting needs to do several things for student who is funded by the department. First, all financial accounts associated with students' financial support need to be verified by supervisors. Second, some students qualify for out-of-state tuition waivers and the accounting needs to make sure the waiver requests are made in a timely manner. Finally, some students' tuition is paid by the faculty, the department, or the University. The accounting first needs to pay the tuition out of a temporary financial account followed by a reimbursement process. Depending on the source of the tuition, the reimbursement comes from different places. All these procedures are recorded and their status is available to the relevant parties. Accounting also manages all the financial accounts used in the department and by associated authorized users. Thus, tool to manage financial accounts and authorized users is needed.

Note that some similar functionality is needed to generate summary pages for individual faculty, the advising, and accounting personnel. These summary pages are used to help manage what needs to be processed – provide reminders to users what they need to do.

3. Summary/Statistics Module

This module provides summary and statistics related to student financial support history. There are three categories: first, department-wide statistics on how students are supported during their degree program; second, financial support that is provided by a particular faculty member; third, an individual student's financial support history.

B. Design

As mentioned in Chapter IV, we split our design into the design of web interface, the design of the database and the design of the code. In FSM, we follow the same design principles for each part.

1. Web Interface Design

In this application, we provided the following web interface:

- *Financial Support Application (FSA) Input* is used by both current students and incoming students to apply for financial support, e.g., teaching and non-teaching assistantships or scholarships. Note that students are likely to apply multiple times during their degree programs. Most information that they need to provide is similar every time they apply for financial support. To minimize the duplicate input, we pre-populated students' latest financial support application data when they want to apply for new ones, then students only need update their information as needed.

- *FSA Monitor* is the summary page on financial support applications for various types of users. For all users, this page displays students' important information regarding their financial support application e.g., ELPE score, GPR, average review score by faculty. It also provides access to the applicants' various information e.g., student records and FSA. Faculty members are able to input their own reviews and view other faculty members' reviews. The financial support administrator is able to make decision on whether the applicant is awarded. The decision status is available for faculty members and applicants. The administrator can also assign duty to awardees.
- *Financial Account Monitor* is used to input and manage all financial accounts used in the department. Any faculty and staff member is able to add a new account and the authorized users associated with an account. But they can only view or edit the accounts that they are authorized to use. The accounting personnel is able to manage all information and access the system log e.g., the creator of an account or the modification time of an authorized user.
- *Financial Support Item (FSI) Request* is used for faculty or staff members to make requests for a new FSI e.g., a faculty member makes a request to provide a research assistantship opportunity for a student. Since accounting personnel need to follow certain procedures by semester (request for out-of-state tuition waiver or pay student's tuition), we automatically generate entries for each semester and allow users to specify detailed information or keep track of the related procedure.
- *Financial Support History (FSH) for a Student* is available for faculty members, staff members and students.
- *FSI Monitor for Advising* is for advising to process their procedures for students' FSI. In particular, it tells advising how many new FSIs they need to process. It

allows advising to upload signed offer letters and manage information related to offer letter. To simplify the file uploading operation, we allow for file update to multiple FSIs. All other information related to students' FSI is available here.

- *FSI Monitor for Accounting* is for accounting to process their procedures for students' FSI. In particular, it tells accounting how many new FSIs they need to process and provides an easy way to put students on the payroll. It also allows accounting to track financial account information each semester, submit information related to out-of-state tuition waivers, and provide support for tuition payment/reimbursement procedures.
- *FSI Monitor for Supervisor* provides lists of FSIs that are associated with a particular supervisor.
- *Statistic* shows different information/statistics related to financial support, ranging from departmental statistics to students' financial support history.

Note that in all the monitor pages (FSA monitor, financial account monitor, FSI monitor for different users) we provide search and sort by different criteria to make information access convenient.

2. Database Design

We follow the ER model for database modeling (Figure 7). The following are the tables used in the financial support monitor.

- *Financial Support Application* table is used to store financial support applications. Each record corresponds to a user's application for a certain semester.
- *Financial Support Application Review* table stores review information for an application. Note that an application may receive multiple reviews.

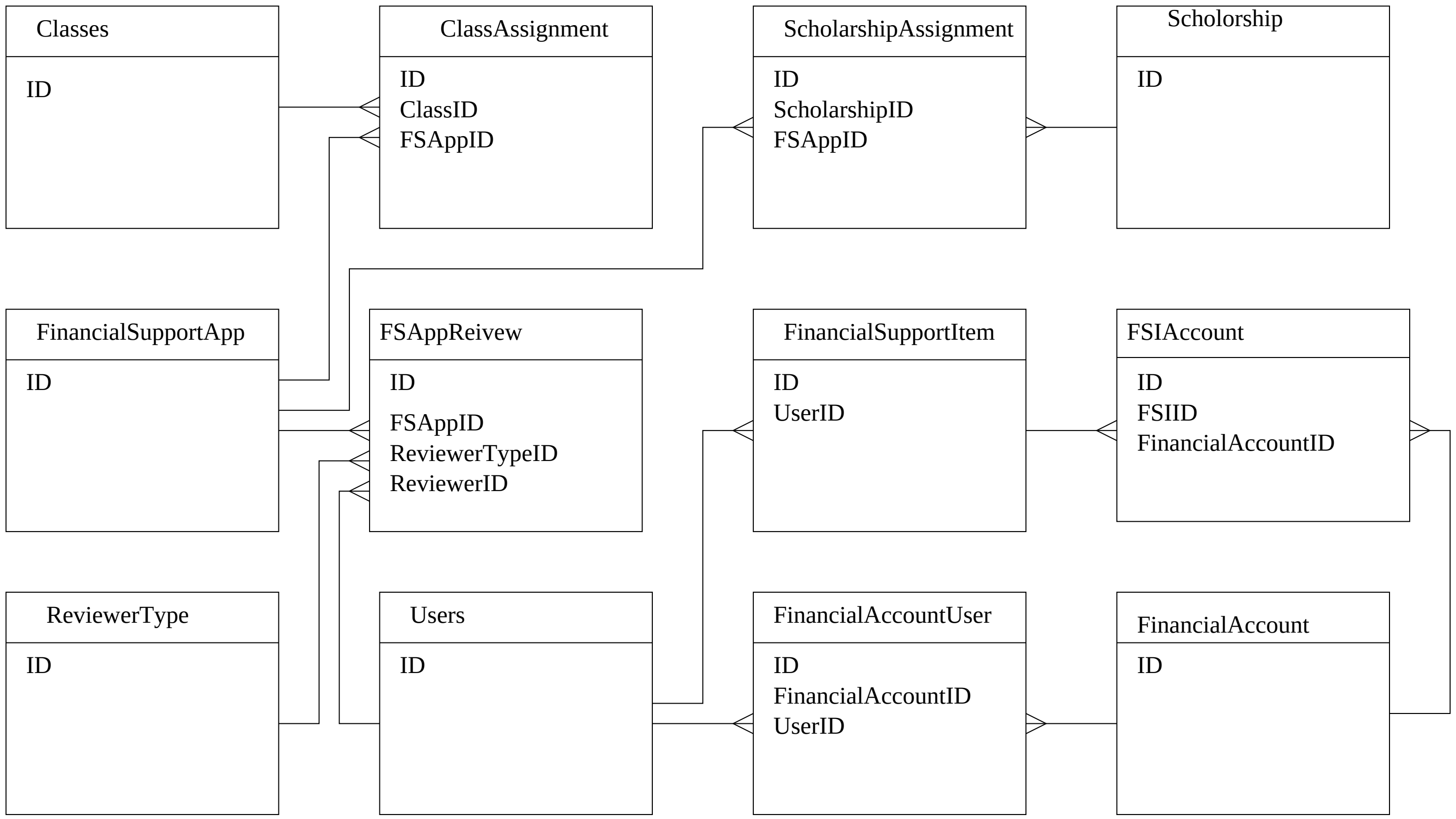

Fig. 7. ER diagram for FSM.

- *Financial Support Application Assignment* tables include two tables: "Teaching Assistant Assignment" table and "Scholarship Assignment" table. Each table is used to store assignments for the application awardee.
- *Academic Class* table stores the academic class offered in each semester. In addition to the class information, it also stores the number of teaching assistants that are needed and is used for FSA assignment.
- *Scholarship Class* table is similar to the academic class table. It stores the scholarships offered in each semester.
- *Financial Account* table stores information for financial accounts used in the department. Since multiple users could be authorized for an account, we use a relationship table called *Financial Account User* to store authorized user information.
- *Financial Support Item* table stores all information associated with one FSI. Since there are different procedure in each semester, we use a relationship table called *FSI Account Information* to store FSI's financial account information in different semester.

3. Code Design

FSM consists of the following main classes (See Figure 8 for class hierarchy). Each has its own design purpose and all of them work together to carry out the functionality required for FSM.

- *FSA class* corresponds to the "Financial Support Application" table in the database. It provides functions to handle database operations, (insert data into the database, load data from database and update data to database, etc). It also provides functions to generate the interface for applicants to fill in their application information.

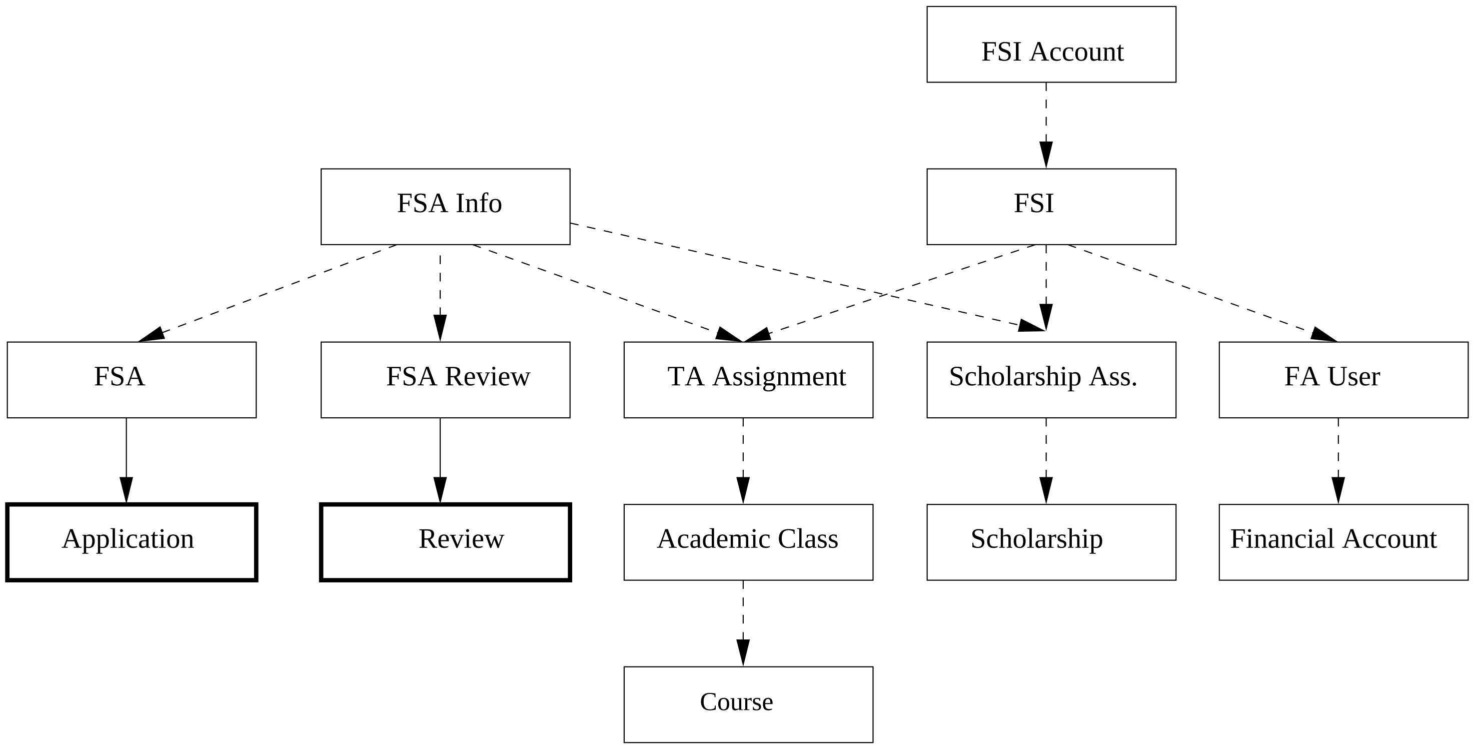

Fig. 8. Class hierarchy (bold box represents the web utility class, solid line represents inheritance and dashed line denotes a “use” relationship) in FSM.

- *FSA Review class* derives from the “Review” class (a web utility class) and provides specialized functions related to FSA review.
- *FSA Info class* is a data management class which stores all information related to a financial support application. An instance of the FSA class is a data member of this class. In addition to the FSA information, it stores other information which is related to this FSA e.g., a user’s personal information, reviews from faculty members, and assignments for awarded applications. It also provides functions to generate different FSA summary pages which provide different criteria for searching and sorting.
- *FSA Assignment classes* are used to make assignments for FSA. There are two classes: assistantship assignment and scholarship assignment.
- *FSI class* corresponds to the “Financial Support Item” table in the database. In addition to the database management functions, it also provides functionality to request a new FSI. It provides functions to generate different FSI summary pages in which searching and sorting by different criteria are provided.
- *FSI Account class* corresponds to the “FSI Account” table in the database. This class provides an interface for supervisors to verify financial account in each semester. It also provides functions related to out-of-state tuition waivers and tuition payments and reimbursements.
- *Other classes* are other classes used to manage specific data e.g., academic class data (instructor, location, the number of teaching assistantships that are needed, etc.), scholarship data, and funding account data.

Code Example. In Figure 9, we show example of how to provide FSI summary pages for different groups of users. We query the database based on the users’ type. For example, students can only view their own FSIs, (as in lines 7–8); supervisors

```
 1 <?
 2 //fsi set to store a set of fsi
 3 $fsiSet = new FinancialSupportItemSet();
 4 //open a new database connection
 5 $dblink = new Connect($DatabaseId,$DatabasePass);
 6 //restrict search condition based on user’s type
 7 if($userType == ’student’)
 8   $whereClause = "FSI.UserID = $stuID";
 9 else if($userType == ’supervisor’)
10   $whereClause = "FSI.SupervisorID = $viewerID";
11 else
12   $whereClause = "";   //get all FSIs
13 //load data from database
14 $fsiSet->loadAllDataByWhereClause($dblink, $whereClause);
15 if($userType == ’supervisor’ || $userType == ’payroll’
16    || $userType == ’advising’)
17   $bEnableEdit = ’true’;
18 else
19   $bEnableEdit = ’false’;
20 $fsiSet->displayRecords($dblink, $userType, $bEnableEdit);
21 ?>
```

Fig. 9. Pseudo-code description of FSI monitor for different group of users.

can only view the FSIs they supervise (as in lines 9–10); and payroll staff or advising staff can view all FSIs (as in line 12). In the example, we allow edit permission only to certain groups (as in lines 15–19).

CHAPTER VI
RESULTS AND IMPLEMENTATION STATUS

This chapter discusses results and usage statistics on OSIS in the first year of production and implementation status.

A. Results

OSIS has made a big impact on the department's day-to-day operation and receive recognition from different types of users.

Within the first year of development, we have successfully released the following main applications: graduate admission, graduate student information monitor, annual PhD student review, financial support application and monitor, faculty search, and REU program. We have implemented around 160 classes (around 86,000 lines of code) in PHP and several tools for publishing and data importing from OAR, SIMS.

The total number of users is around 9,000. In particular, 63% of our current users are references for various types of applicants, 29% of our current users are various types of applicants, and the remaining 8% of our current users are current members in the department.

About 92% of our current users are using the web server that is outside the TAMU firewall, that is, the external web server. The average monthly hits total 51,000 and range from 3,000 to 120,000. These statistics reflect the nature of the applications supported in the external web server: Graduate program admission dominates the usage, but it only occurs from November to April during the year. Statistics on daily usage and hourly usage for a recent month are shown in Figures 10 and 11 respectively.

Although only 8% of our current users use the internal web server, the average

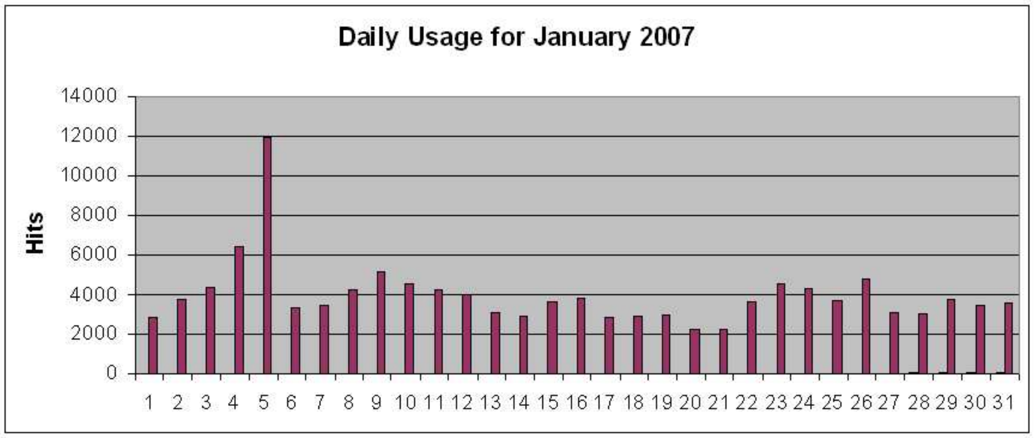

Fig. 10. Daily usage for the external web server.

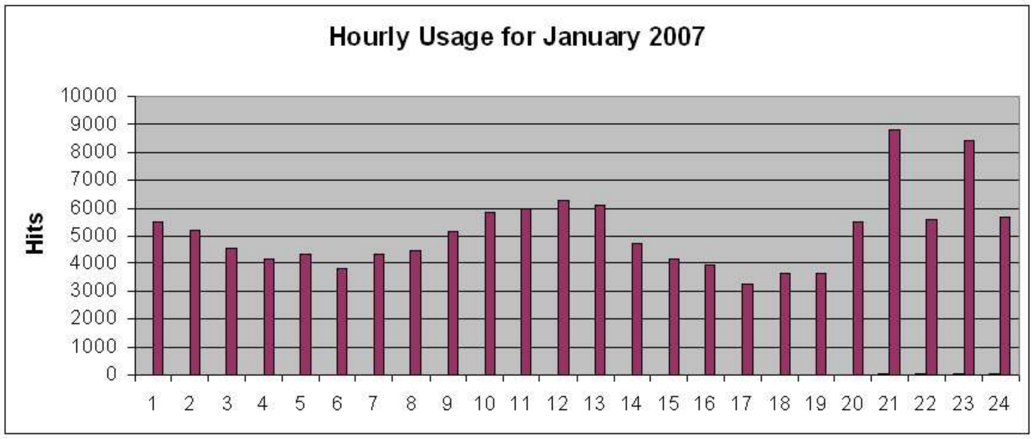

Fig. 11. Hourly usage for the external web server.

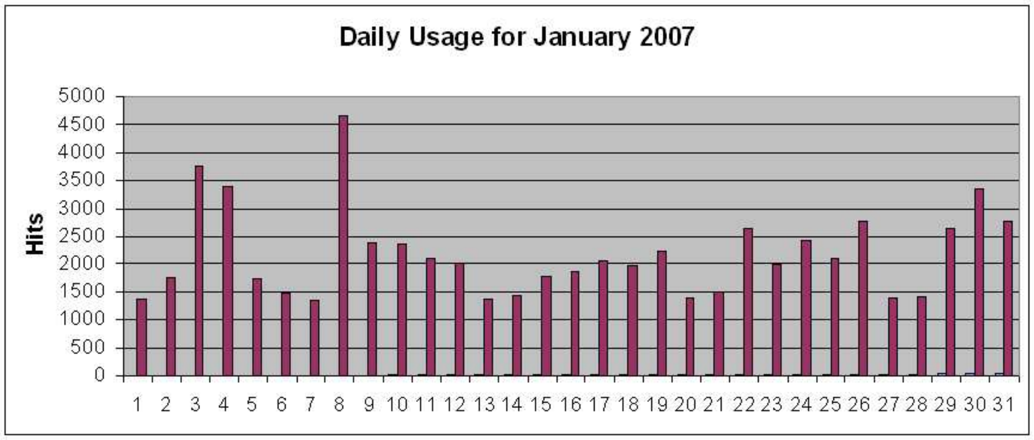

Fig. 12. Daily usage for the internal web server.

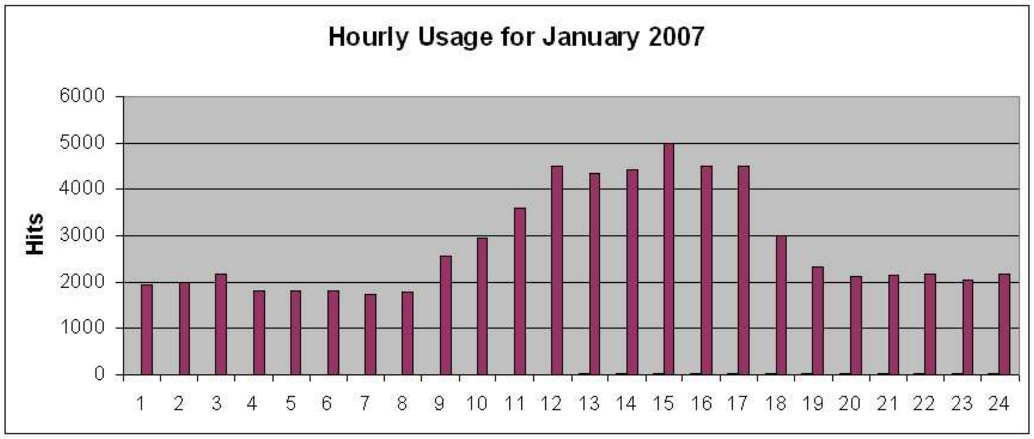

Fig. 13. Hourly usage for the internal web server.

monthly hits are around 46,000 and ranges from 15,000 to 67,000. This is because the internal web server holds both the departmental applications and the administration work for all the external applications (the review and the departmental administration). Statistics on daily usage and hourly usage for a recent month are shown in Figures 12 and 13 respectively.

B. Implementation Status

Chapter III describes the requirements we determined were necessary for a system such as OSIS to support the information management for our department. In this section we outline the status of the current implementation of OSIS.

1. Students (OSIS-grad & OSIS-ugrad)

Common functionality for current students:

- Implemented.
  - Student basic information.
  - Resume. Submission and its use in financial support application evaluation and other programs, e.g., the IAP. The student is able to specify how it can be used (restricted to the department, viewable to the IAP, etc).
  - Degree progress monitor.
  - Financial support application, history, selection, and evaluation.
  - Email support: mailer tool and email history.
  - Statistics and report generation (admission, graduation, and enrollment).
- Not Implemented.

  - Email support: listserv management.
  - Letter request and generation (good standing, travel, etc).

Common functionality for former students:

- Implemented.
  - Listing of alumni and their academic information.
- Not Implemented.
  - Tracking of alumni.
  - Alumni events, newsletters, and other services.

OSIS-grad specific functionality:

- Implemented.
  - Graduate admissions.
  - Annual PhD review.
- Not Implemented.
  - Petition for degree level change.

OSIS-ugrad specific functionality:

None of the requirements for OSIS-ugrad specific functionality are implemented.

2. Faculty (OSIS-faculty)

- Implemented.
  - on-line submission of application materials by applicants and references.
  - on-line review by faculty members.

  - on-line decision by the search committee.

- Not Implemented.

  - interview support for faculty search.
  - submission of annual progress reports by all faculty and on-line evaluation.
  - promotion and tenure materials submission.

3. Department Programs and Processes (OSIS-dept)

Academic programs and departmental committees:

- Implemented.

  - Course planning and scheduling.

- Not implemented.

  - Course Requests and Committees.

REU program:

All requirement for the REU Program except the program administration are implemented.

IAP program:

None of the requirement for the IAP program are implemented.

Award Tracking:

None of the requirement for award tracking are implemented.

Technical Report Tracking:

None of the requirement for technical report tracking are implemented.

CHAPTER VII
CONCLUSION

We presented a departmental information management system called OSIS. OSIS was developed following Information Science design paradigm, Software Engineering techniques, and Computer-Human Interaction guidelines. It is a well-designed object-oriented application. We also used Financial Support Monitor (FSM), which is a tool to monitor student financial support history and support different departmental procedures, as the case study for system requirement analysis, system design, and implementation.

OSIS has made a big impact on the department's daily operation. It improves the efficiency of many operations. The difference in processing time is significant. There are many things we can do now with OSIS that we could not do before.

In OSIS, we designed and implemented a set of web utility classes which are general and self-contained. They greatly increase the re-usability and maintainability of our codes. The web utility classes can be used by other web applications. We plan to make them open source in the future.

Although OSIS is a customized information management system for the Department of Computer Science at Texas A&M University, it can be easily configured to be used by other academic departments. Also, we can extend OSIS to support much larger user groups e.g., an academic college or even the whole university.

REFERENCES

[1] The Apache Software Foundation. Accessed January 2007. [Online]. Available: http://www.apache.org/.

[2] R. M. Baecker, J. Grudin, W. A. S. Buxton, and S. Greenberg, *Readings in Human-Computer Interaction: Toward the Year 2000.* New York: Morgan Kaufmann Publishers Inc., second edition, 1995.

[3] P. P.-S. Chen, “The entity-relationship model – toward a unified view of data,” *ACM Transactions on Database Systems*, 1:9–36, 1976.

[4] Introduction to Data Modeling. Accessed January 2007. [Online]. Available: http://www.utexas.edu/its/windows/database/.

[5] C. J. Date, *An Introduction to Database Systems.* Boston, MA: Addison-Wesley, fifth edition, 1990.

[6] P. J. Denning, “A new social contract for research,” *Communications of the ACM*, 40:132–134, 1997.

[7] C. A. Ellis, S. J. Gibbs, and G. Rein, “Groupware: Some issues and experiences,” *Communications of the ACM*, 34:38–58, 1991.

[8] J. D. Gould, How to Design Usable Systems. In *Excrept from: Readings in Human-Computer Interaction: Toward the Year 2000.* New York: Morgan Kaufmann Publishers Inc., 1995. pp 93-121.

[9] J. Grudin, “Why CSCW applications fail: Problem in the design and evaluation of organizational interfaces,” In *Proc. of the 1998 ACM conference on Computer-supported Cooperative Work*, pages 85–93, 1988.

[10] J. Grudin, “Computer-supported cooperative work: history and focus,” *IEEE*

Computer, 27:19–26, 1994.

[11] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science research in information systems," *MIS Quarterly*, 28:75–105, 2004.

[12] T. W. Malone and K. Crowston, "The interdisciplinary study of coordination," *ACM Computing Surveys*, 26:87–119, 1994.

[13] S. T. March and G. Smith, "Design and natural science research on information technology," *Decision Support System*, 15:251–2661, 1995.

[14] MySQL. Accessed January 2007. [Online]. Available: http://www.mysql.com/.

[15] J. Nielsen, *Usability Engineering*. San Francisco: Morgan Kaufmann, 1994.

[16] S. Opper and H. Fersko-Weiss, *Technology for Teams: Enhancing Productivity in Networked Organizations*. New York: Van Nostrand Reihold, 1992.

[17] D. Petrelli, A. D. Angeli, and G. Convertino, "A user-centered approach to user modeling," In *Proc. ACM, International Conference on User Modeling*, pages 255–264, 1999.

[18] The PHP Hypertext Preprocessor. Accessed January 2007. [Online]. Available: http://www.php.net/.

[19] H. A. Simon, *The Science of the Artificial*. Cambridge, MA: MIT Press, third edition, 1996.

[20] I. Sommerville, *Software Engineering*. Boston, MA: Addison-Wesley, fifth edition, 1996.

[21] Universally Unique Identifiers (UUID). Accessed January 2007. [Online]. Available: http://www.itu.int/ITU-T/studygroups/com17/oid.html.

APPENDIX A

CLASSES FOR OSIS

In this appendix, we show the classes hierarchy in the applications we have successfully released: graduate admission, graduate student information monitor, annual PhD student review, faculty search, and REU program. Note that the class hierarchy in financial support application and monitor is shown in Figure 8 in Chaper V.

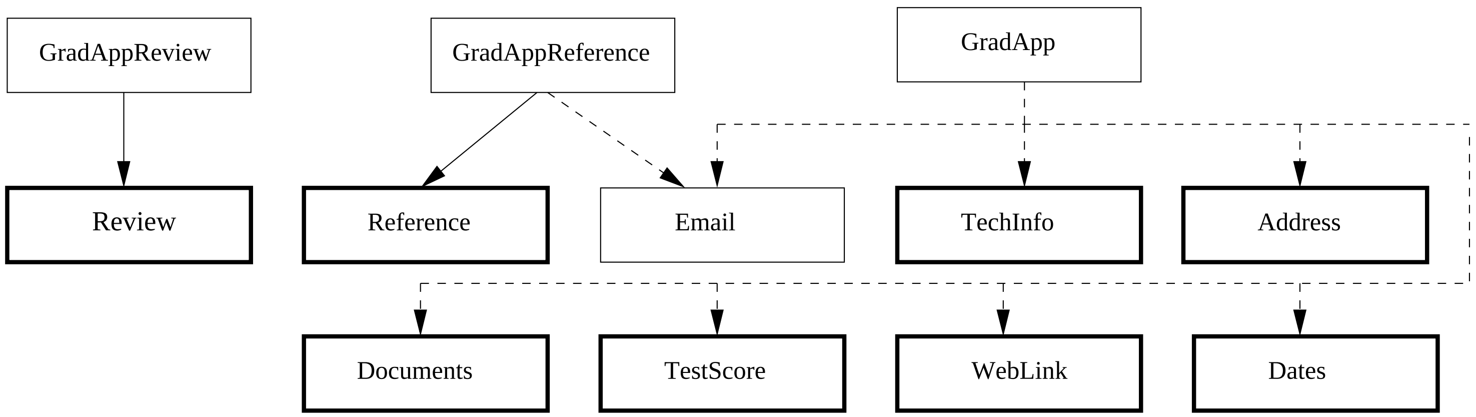

Fig. 14. Class hierarchy (bold box represents the web utility class, solid line represents inheritance and dashed line denotes a "use" relationship) in graduate admission.

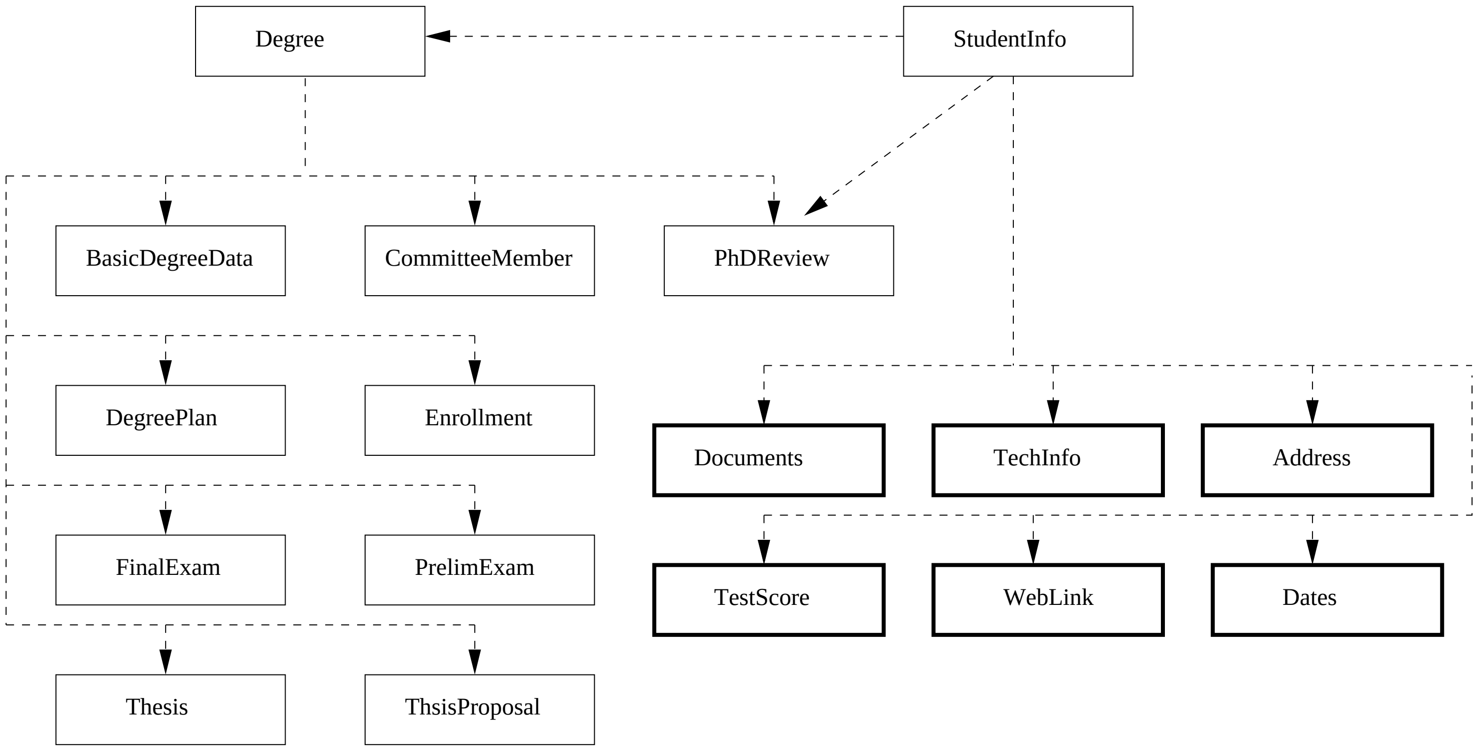

Fig. 15. Class hierarchy (bold box represents the web utility class, solid line represents inheritance and dashed line denotes a “use” relationship) in student information management.

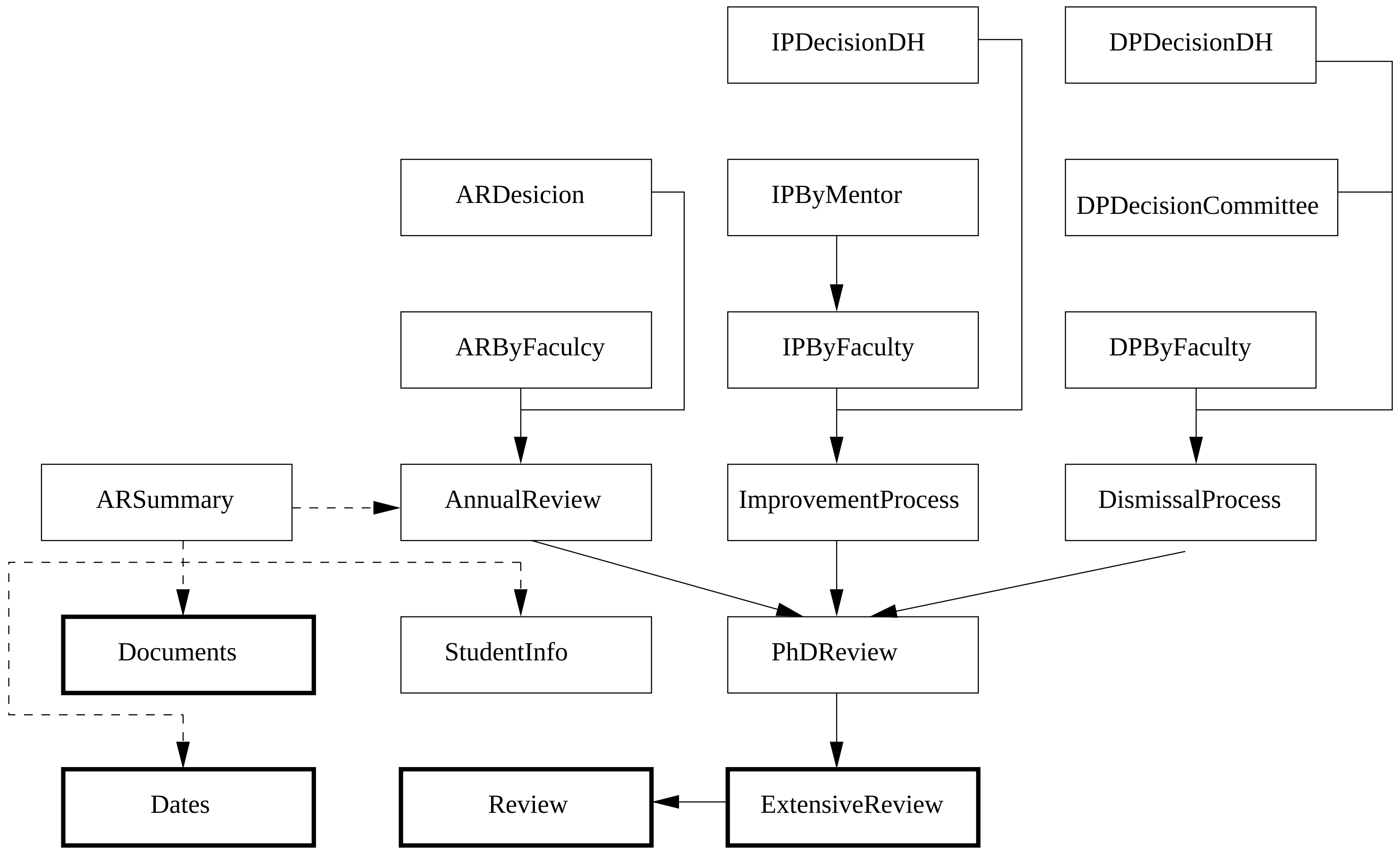

Fig. 16. Class hierarchy (bold box represents the web utility class, solid line represents inheritance and dashed line denotes a "use" relationship) in PhD Student Annual Review.

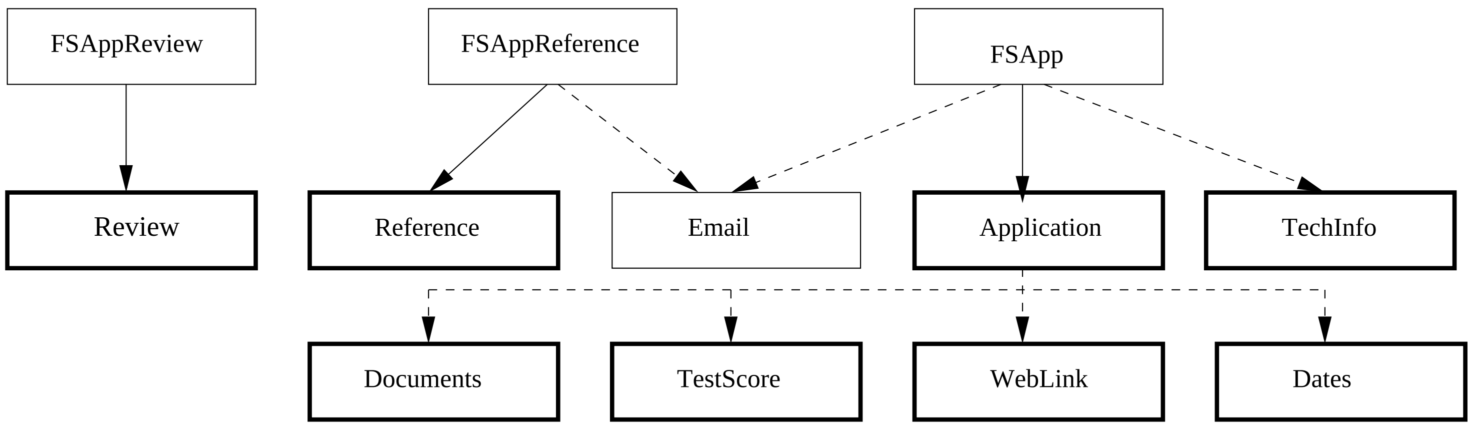

Fig. 17. Class hierarchy (bold box represents the web utility class, solid line represents inheritance and dashed line denotes a “use” relationship) in faculty search application.

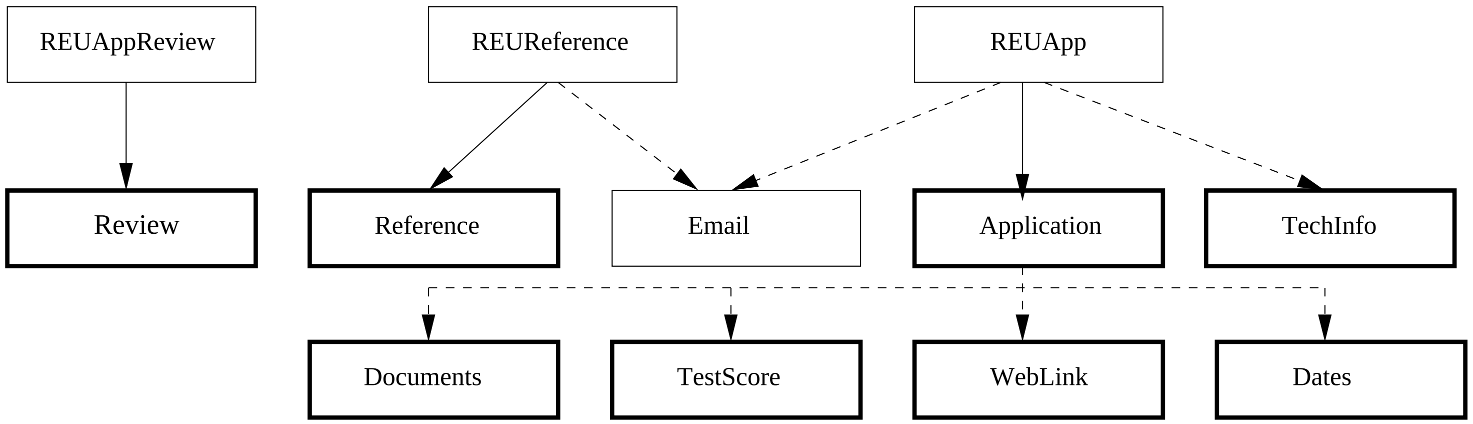

Fig. 18. Class hierarchy (bold box represents the web utility class, solid line represents inheritance and dashed line denotes a “use” relationship) in Research Experience for Undergraduate (REU) program.

APPENDIX B

DATABASE FOR OSIS

In this appendix, we show the Entity-Relationship (ER) diagram for the applications we have successfully released: graduate admission, graduate student information monitor, annual PhD student review, faculty search, and REU program. Note that the ER diagram for financial support application and monitor is shown in Figure 7 in Chaper V.

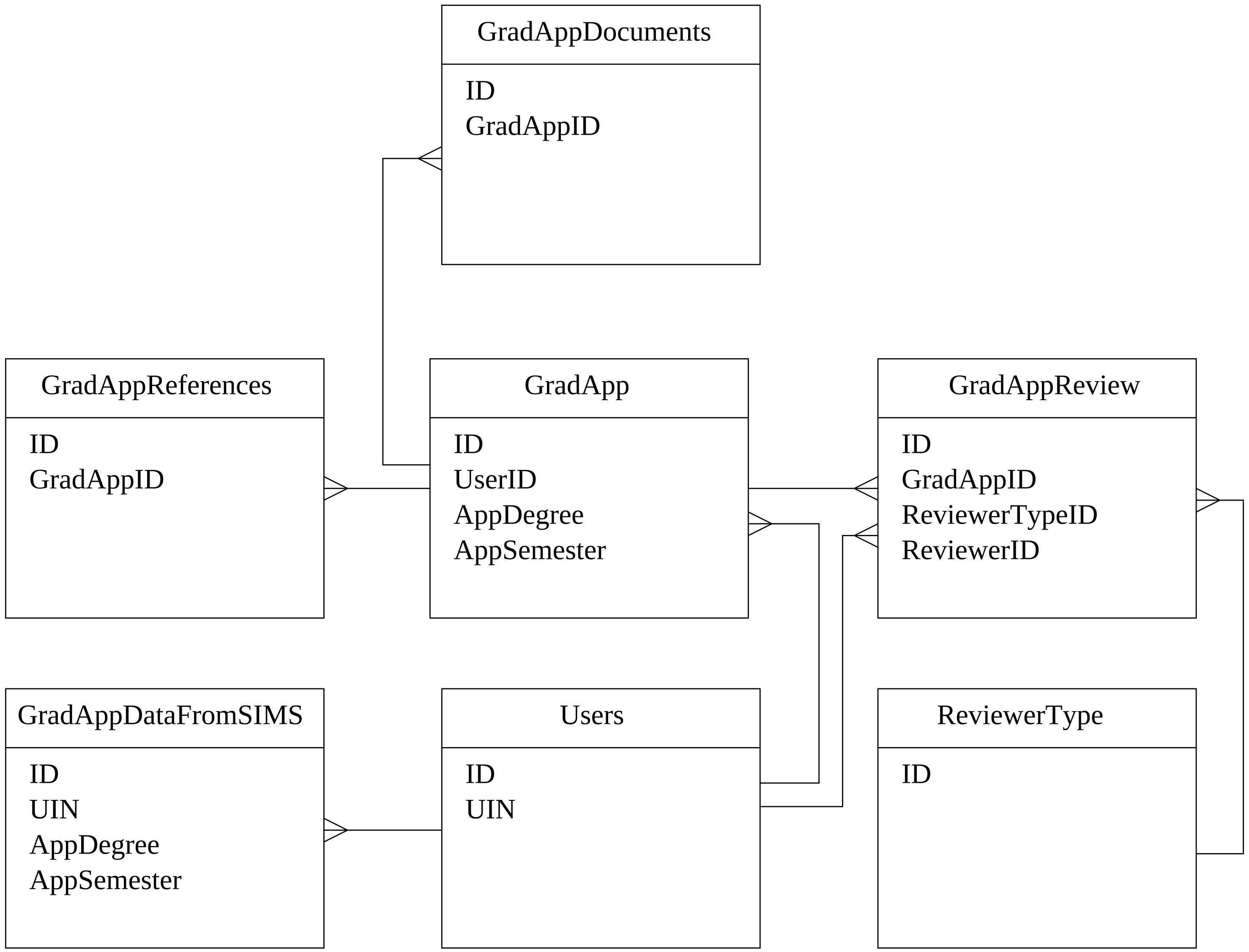

Fig. 19. ER diagram (box represents the entity, line represents the relationship and "crow's feet" denotes the many sides of a relationship) for graduate admission.

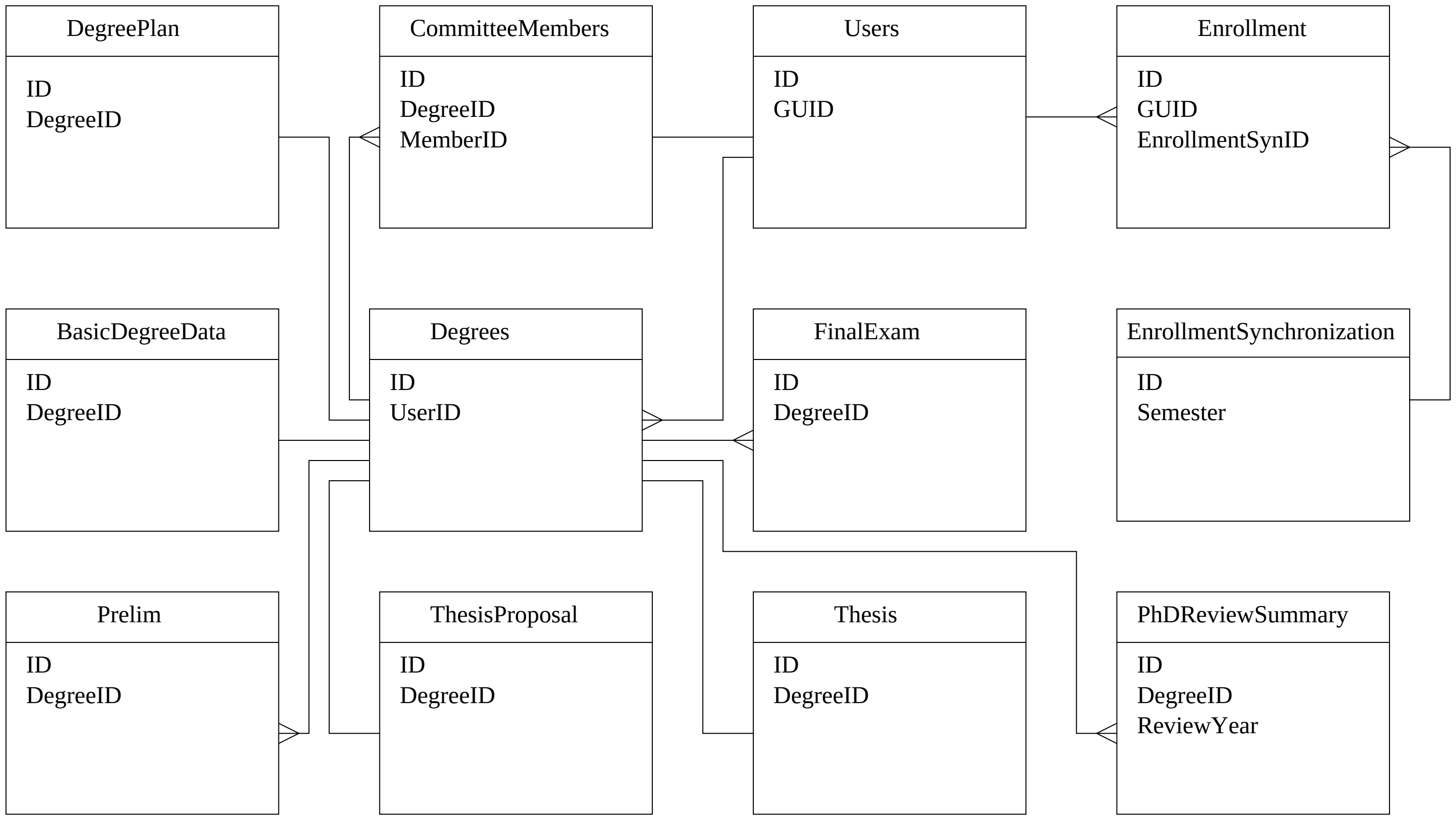

Fig. 20. ER diagram (box represents the entity, line represents the relationship and "crow's feet" denotes the many sides of a relationship) for graduate student information monitor.

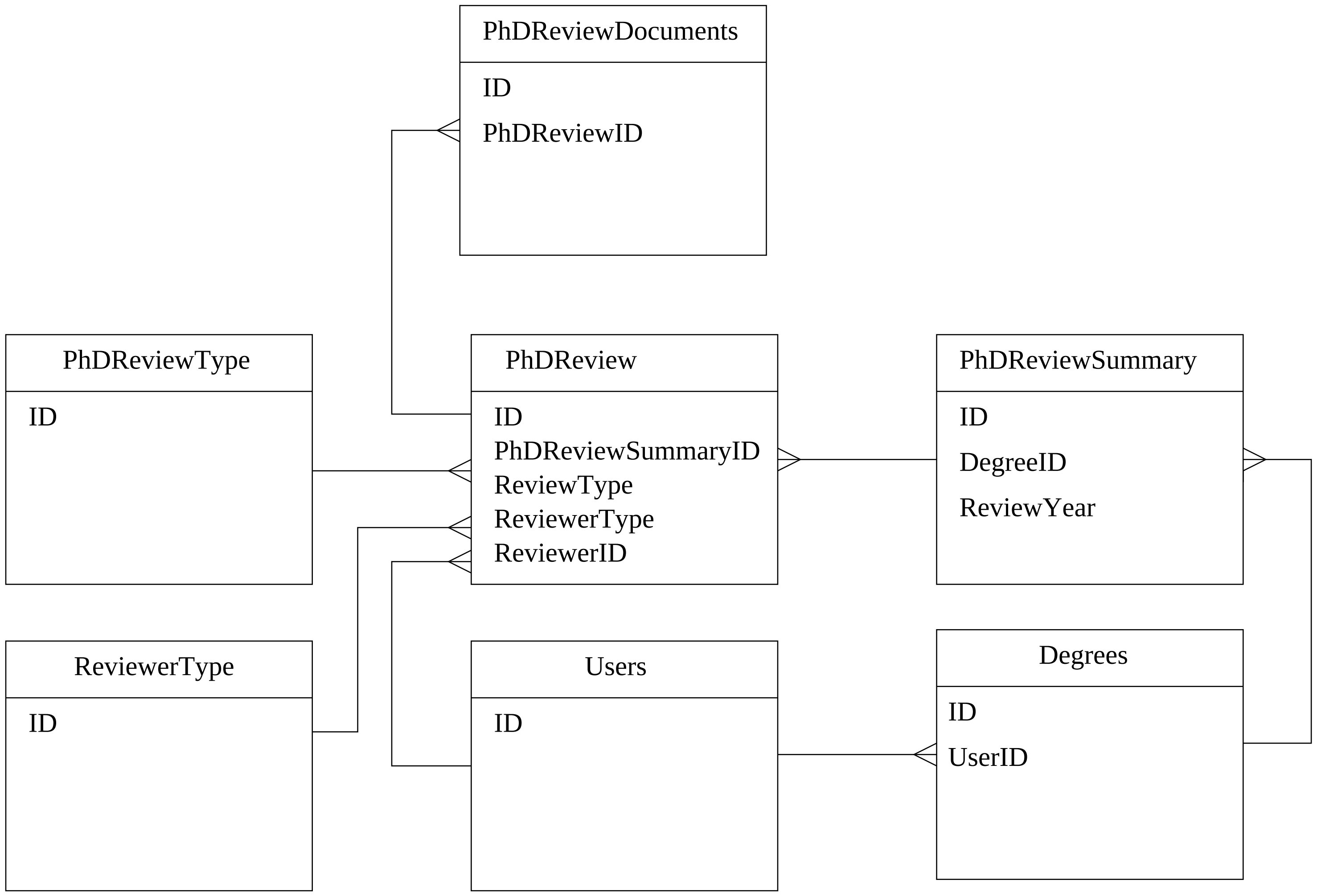

Fig. 21. ER diagram (box represents the entity, line represents the relationship and “crow’s feet” denotes the many sides of a relationship) for annual PhD student review.

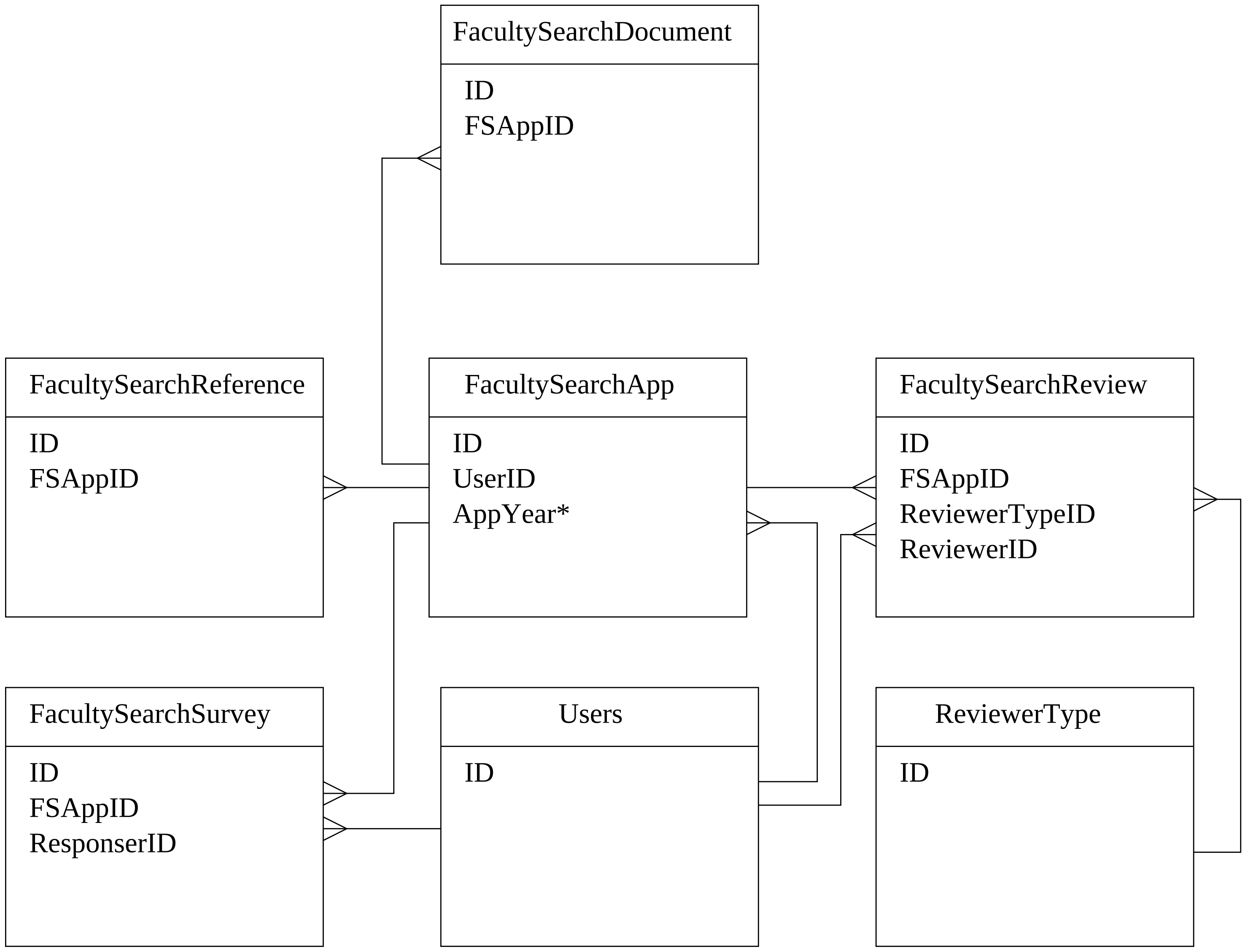

Fig. 22. ER diagram (box represents the entity, line represents the relationship and “crow’s feet” denotes the many sides of a relationship) for faculty search.

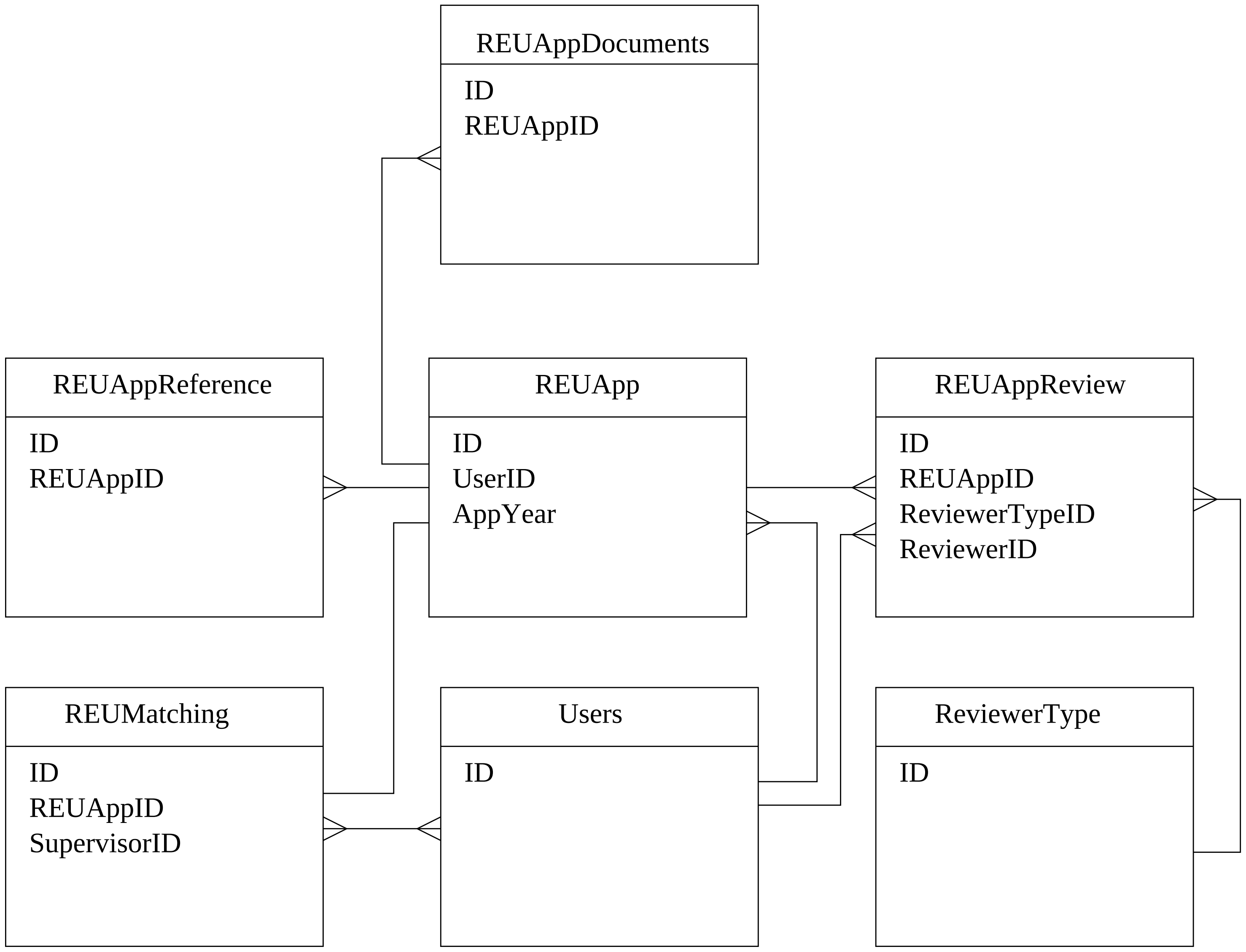

Fig. 23. ER diagram (box represents the entity, line represents the relationship and “crow’s feet” denotes the many sides of a relationship) for Research Experience for Undergraduate (REU) program.

VITA

Dawen Xie received his B.S. in computer science at Nankai University, Tianjin City, in 1999. He has worked on motion planning research for a few years and published three papers.

[1] D. Xie, M. A. Morales, R. Pearce, S. Thomas, J.-M. Lien and N. M. Amato, "Incremental map generation (IMG)," in *The Seventh International Workshop on the Algorithmic Foundations of Robotics*, Jul. 2006.

[2] B. Bayazit, D. Xie and N. M. Amato, "Iterative relaxation of constraints: a framework for improving automated motion planning," in *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2005, pp.586-593.

[3] D. Xie and N. M. Amato, "A kinematics-based probabilistic roadmap method for high DOF closed chain systems," in *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, Apr. 2004, pp. 473-478.

Dawen Xie may be reached at 418 Price Hall, Virginia Tech, Blacksburg, Virginia 24061. His email address is dawenx@gmail.com.