December 1998

A Three-Pronged Approach to an Object Oriented Computer Information Systems Curriculum

Wayne Staats
Stetson University

A Three-Pronged Approach to an Object Oriented Computer Information Systems Curriculum

Wayne J. Staats
Department of Mathematics and Computer Science
Stetson University

Abstract

As a Computer Information Systems curriculum, most majors enter corporate America instead of opting for graduate school. Although they are entering industry, many of these students are not prepared for to be instantly productive in commercial software development. To address this short coming, some of the faculty at Stetson University have begun a three-prong approach to incorporate the object oriented paradigm into the undergraduate computer information systems curriculum. The three prongs are 1) the adoption of Java and the Object Oriented paradigm for the introductory courses, 2) self directed, web based courses on software engineering topics, and 3) the formation of a consulting firm to provide real world, hands-on experiences. All three components of the curriculum are needed to help the students (and faculty) to fully realize the power of the object oriented paradigm, since it's part computer programming, part software engineering, and part philosophy. The three prongs of our curriculum address the entire paradigm while providing a hands-on learning experience.

Introduction

In the AIS, DPMA, and ACM suggested undergraduate curricula, the basic computer concepts and theoretical constructs are well organized and time-tested. The switch to the Object Oriented paradigm provides some interesting challenges to the traditional curriculum since it requires the incorporation of many software engineering concepts. From the author's view, having developed object oriented systems professionally for 8 years, the object oriented paradigm is seen as a combination of computer science, software engineering, and philosophy. Without addressing all three components of the paradigm, programmers are unable to apply the object oriented paradigm correctly. One need not look too far to see the pitfalls of mixing different paradigms; many companies have incorporated an object enabled language into their software development process while ignoring the other two aspects. The results have been anything but encouraging with 90% [Ohn98] of the "object oriented" development actually being some mixture of paradigms. Even Booch [Boo91] refers to the philosophical side of the object oriented paradigm when he states it takes an expert's knowledge to properly model a complex system using objects.

In order to address these three components of the object oriented paradigm, we have begun to implement a three-pronged approach for incorporating the object oriented paradigm into the undergraduate computer information systems curriculum. The three prongs include the adoption of Java as the introductory language, the creation of web based courses on software engineering topics, and the formation of a consulting firm, the Stetson Software Institute, run by the faculty and staff of the university. By creating this approach, students (and faculty) get a full understanding of the power and fun of the object oriented paradigm in an exciting educational program. The remainder of this paper discusses in greater detail the three aspects of our curriculum. Table 1 provides an outline of the courses, the web based courses, and how each level of student can be involved in the software institute. Because of the large number of courses, this paper will not discuss these courses in any detail except for the first four in-class courses where the object oriented paradigm is introduced and developed.

Traditional Programming Courses

The switch to the object oriented paradigm in the introductory courses provides many avenues which a program could follow. The ACM Special Interest Group for Computer Science Education (SIGCSE) has been debating the traditional development of data structures (stack, queues, etc.) versus the use of existing classes provided by a class library. At Stetson, the first four courses in the computer science curriculum are also used by two other majors: Digital Arts for Computer Scientists and Computer Information Systems. In these two majors, the focus is on having the students understand how/when to apply these data structures instead of understanding their inner workings. For this reason, the faculty have begun to shift towards using existing classes instead of developing them. We are planning on pushing the development and analysis of the basic data structures out to the junior year when the students can better grasp the meaning of algorithm analysis, however, we have not done this yet.

In CS101, the focus remains on teaching the basic syntax of the language, Java in our case, and proper use of classes and objects. In CS102, more object oriented concepts (coupling, cohesion, polymorphism, encapsulation, etc.) are introduced through the design and development of numerous applications (Applets) utilizing preexisting data structures (stacks, queues, etc) provided

by Java. Student are also introduced to Rational Rose [Ros97] for designing their applications. In CS221, the third course, the language is switched from Java to C++. Half the semester is spent teaching the syntax of C++ and the implementation of the object oriented concepts and data structure usage that was taught in CS102. The faculty feel the size and complexity of C++ warrants the amount of time spent on it. The remainder of the class is used to introduce the students to Design Patterns [Gam95] and software engineering concepts. In the fourth course, CS320, the student's knowledge of object oriented programming, design patterns, and software engineering are further reinforced through the design and development of larger applications.

After the first four courses, the students should have strong programming skills and a good understanding of the object oriented paradigm and software engineering concepts. The remainder of the curriculum fills out the student's understanding of information systems with courses in Object Oriented Development using Visual Basic (IS388), Databases (IS397), and Client/Server Development (CIS305). Because of the student's strong base in software engineering, each of the courses requires the use of the software engineering concepts taught in the earlier classes.

Web Based Courses

Even with the reorganization of the first four courses, not all of the software engineering concepts needed to prosper in a software development environment are covered. With the limited number of faculty and limited time in the first four courses, many of the software engineering concepts are given a short amount of time. In order to provide the students with a mechanism to further explore these concepts, we have proposed the development of a set of self directed, web based courses on software engineering concepts. These courses would be required with each student taking one per semester. Besides completing the online course, the students would be required to attend periodic meetings to discuss the material. This approach allows the computer science faculty to require the use of software engineering concepts in other courses where students are developing software. For a proposed list of online courses, refer to Table 1.

The Stetson Software Institute

Through the use of courses, both classroom and online, the faculty at Stetson University should be able to provide the students with the theoretical concepts of computer science, the object oriented paradigm, and software engineering concepts. These courses, due to the limited time for assignments, can not provide enough large problems where the students can gain insight into the philosophical aspects of object oriented development. It's one thing to design a bank example, where the application has 5 or 6 classes, and another to design a fully integrated accounting package that contains hundreds of classes. It's in the larger programs that more interesting object oriented analysis and design is encountered, since it's not always black-and-white with respect to the design, as the design is not usually straight forward. This is where the software institute comes in. By contracting with corporations, the institute provides a controlled environment where students can team with professionals and faculty members to develop commercial grade software.

The partnership between the institute and a company is beneficial for all parties involved. It provides the students with real world experiences, the faculty with a mechanism to maintain and enhance their skills, the departments with income and guidance to maintain state-of-the-arts programs, and the company commercial grade software at reduced rates. Naturally, because the bulk of the work is being completed by students, the time required to develop the software will be longer than that of a professional consulting agency.

Conclusion

By creating a three-pronged approach to the undergraduate computer information systems curriculum, we expect to be able to better prepare students for not only the work place, but also understanding the theoretical concepts of advanced computer science. This approach was originally presented and refined at the NSF-CISE summer workshop for integrating research into the undergraduate curriculum held at The Evergreen State College, which was funded by the National Science Foundation, CDA 931-26-48.

References

[Boo91] Grady Booch. *Object-Oriented Design With Applications*. Benjamin/Cummins, London, 1991.

[Gam95] Gamma, E., Helm, R., Johnson, R., Vlissides, J. *Design Patterns, Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.

[Ohn98] Ohnjec, Viktor and Kranz, Don. *Developing Distributed Object Systems. An Architectural overview for managers*. Object Magazine. Vol 7, No. 12, February 1998.

[Ros97] Rational Rose. Rational Software Corporation, Cupertino, CA, 1997.

Table 1

	Courses	Web Based Courses	Software Institute
Freshman	**CS101** - Java **IS 191** - Intro. to IS **CS102** - Java	Code Maintenance Code Reading Testing	Maintenance Testing
Sophomore	**CS221** -C++/OO Development **IS294** - Fund. Of IS. **CS320** - OO/Design Patterns	Requirements Class Design Abstract Modeling Program Verification	Class coding Class/Object design
Junior	**IS393** - Telecommunications & Networking **IS397** - Database Design & Implementation **IS388** - OO Development w/Visual Basic **CIS305** - Client/Server Development	System Design Design Reviews	Object Modeling System Design
Senior	**CIS498** - Sr. Project Proposal **IS431** - Managing Information Resources **CIS499** - Sr. Project	Project Management Risk Management Metrics Estimation	Project Management Student Mentoring Proposal Writing and System Architecture