DEVELOPING ONLINE OBJECT-ORIENTED INFORMATION TECHNOLOGY CURRICULA: COLLABORATING THROUGH PRACTITIONERS WITH BASE-CLASSED BASE CLASS EXPERIENCES

Dr. Paul M. Chalekian, University of Nevada, pmc@equinox.unr.edu

ABSTRACT

The development of online education curricula for object-oriented (OO) classes can demonstrate key principles of OO. The academic framework used by the institution could approximate the concepts being described. Collaboration, like the forms used by most software development teams, can be the primary component in establishing a vital curriculum. As such, advanced OO concepts can be validated both by students and instructors, each from a "user" perspective. The instantiation and substantiation of base classes can occur by drawing from the practical experiences of contributors. The best of those experiences can be shared with relevant examples.

Keywords: Enterprise Framework, Online Education, Learning Repository, Base Class, Educational Objects

INTRODUCTION

At the start of one "on-ground" graduate-level course on advanced programming, the instructor asked the following question of his students: "How many of you have inherited from a *Base Class*?" In their eagerness, about ½ the hands sprang up. As the seconds passed, and the instructor eyed many more than he expected, the hands slowly dropped. This indicated that many of the students did not know what a true "*Base Class"* meant. The problems of teaching object-oriented (OO) classes are widespread [3, 7]. From the student's previous assignments, it is quite possible that they had never encountered or used the foundational *Base Class* concept. From the instruction side, the world of academia can be quite attractive for information technology practitioners. Unlike university-based instructors who have spent much of their time working with theories, those that grapple with day-to-day problems in large organizations can provide unique perspectives. Experience grappling with frameworks is key. They might also see newly employed students and how they are not prepared to be productive in software development. But how can the gap between simplistic academic assignments and complex OO programming be narrowed? Collaborating between online practitioners who explain *Base Class* concepts, while collectively drawing from base-classed lectures, could help.

This article will look at how students and/or new developers may view enterprise frameworks as they arrive "out of the box." The existence of advanced online education frameworks will be described as an example for students to try and actively comprehend. The author's experience with the largest online educator provides the basis for the discussion. Some post-education employment scenarios will be presented, as well as the need to reexamine information technology curricula. Lastly, some factors will be suggested which could enable the development of a foundation class library containing educational objects.

BASE CLASSES

A *Base Class* has been thought of as the most generalized class structure [2]. The size of an employing entity may be very influential to a student in terms of understanding a *Base Class* framework's significance. For instance, if an organization is large scale, they may have a sizeable enterprise framework already intact. From this foundation, the entity could be poised to rapidly grow and expand. Since these organizations typically have larger development teams, they may hire more frequently, and thus they may also provide the first position for a new graduate.

Key aspects of organizations involve the protecting of "core" technologies and the degree of routine or non-routine variability. Organizational Theorist James D. Thompson wrote about these technical "cores" of technology and how they could become a central organizing element of organizational efforts. According to him, administrators should go to great lengths to protect it. A "core" technology is one that provides leverage to the organization to fulfill its mission and grow [11]. However, priorities often dictate what applications will be developed and, as a result, the "core technologies" may become vulnerable to short sightedness. The full development of a "core" framework may be rushed and/or

compromised. If an organization's online education is unable to grow and extend, enrollment problems could result.

Base-classed libraries are segments of code that are common to many functional areas. Examples of these libraries may include code for operating system interfaces, third party middleware and/or security. Functionally specific code, although more voluminous, is the remaining code that cannot be feasibly shared among teams of developers. If the code or processes can be shared, perhaps through a process owner [4], then it should be considered for inclusion in the framework. But one problem is that these OO techniques are often developed independently from other software engineers [10]. Unless the history is known about why code segments were *classed*, the student or inexperienced developer may become frustrated and less effective.

One influential researcher has empirically shown how the concepts relating to OO have been far less than consensual [1]. It is quite possible that students could fail to realize the significance of *base classes*. From many software packages, inexperienced developers may be unguided and yet be tasked with systems development. But by re-examining most online education frameworks (perhaps the one from which they had been taught), a student may be able to discern modularity and code reuse capabilities. Educational objects are examples of modularity and reuse. According to Booch Grady, whenever there is a group of programs that all solve substantially similar problems, there is an opportunity to create an application framework [2]. In terms of their development, this is true for the students, as well as the code-based/lecture-based artifacts. As is currently the practice and as suggested below, a set of educational objects, with a learning repository of experiences, could be developed for online delivery.

ONLINE EDUCATION FRAMEWORKS

As is commonly used, educational objects could be placed into a learning repository and become the basis for an educational framework. An online curriculum can be thought of as an enterprise framework for gathering lectures, gleaning the best and discarding the rest. But more so, they can be thought of as being *in situ* examples of an enterprise framework. Many instructors have done mid-course corrections over the years and, with a team of instructors, more refinements can occur. An asynchronous mode is how the majority of the online education course content is delivered [8]. With online collaboration, the most advantageous lectures can be refined and used repeatedly. These concepts have already been established for online education [12], but this article suggests a greater emphasis on providing examples for teaching object-oriented development. From a decentralized development approach, the perspective of "the users" is of utmost importance [9]. If a lecture is unclear, it can be pulled from the framework, scrutinized as an educational object, and reapplied with another set of examples that are more effective.

LIKELY POST-EDUCATION EMPLOYMENT

If a student leaves the academic world and is employed by a small organization, they may not have exposure to an inheritance, object or class-based framework. However, their learning potential may be available "out of the box." Whether that potential is used presents another issue. The smaller the entity they are employed by, the less likely the functions of their work would be recognized as being repeatable and encapsulated into a base class. Yet, a student who is strong in OO principles can be pivotal to the success of a small organization [13]. If a student graduates and gains employment in a large organization, chances are they would be exposed to some type of enterprise framework. Their challenge becomes how to learn the framework's functionality, while at the same time being productive in a functionally specific area and juggling priorities.

If the student joins a small development environment and is lacking the experience of team building, they may not discern the importance of base-classed code as they begin to program and perform maintenance. As a result, their code may lack modularity and be "long-linked." A long-linked technology is characterized by serial interdependence, in which the actions of Z can only be performed after the actions of Y, which in turn is contingent on the actions of X, and so on [11]. An example of this is one or more organizational entities linking information or products together. It has been well identified how students benefit by understanding encapsulation [13]. If the student is in the larger environment and works with a team, they will soon begin to appreciate the value of code reuse. They would find that the common code is in the base classes and further, how productive programmers work predominantly in their functionally specific areas, while being mindful of what is available in the common base class libraries.

REEXAMINING ONLINE CURRICULUM DEVELOPMENT

Online classes that draw from practitioners may stress the "core" versus "functionally specific" differentiation. These practitioners may come in three forms: instructor availability, instructors with "lessons learned," and academics that venture into research about large organizations. Each has unique perspectives to provide to the learning repository and these categories will be presented below.

If an instructor from the programming world is flexible enough to teach, the organization where he or she is employed must be successful to some degree. It is quite probable that strong development processes are in place, as well as an enterprise framework. That framework may have been purchased or developed in-house. These individuals may add to a curriculum that expounds the virtues of frameworks, but they may also address the stability of processes.

Other instructors may realize the importance of frameworks, but from another perspective. If their organization lacks an enterprise framework from which to extend and grow, that entity may have encountered problems, setbacks and/or cutbacks. These stories are worthy of mention under "lessons learned" and, due to outsourcing, these experiences could become more common. However, the personality traits of the instructors should be taken into consideration [5].

If an academic ventures into the practitioner's world, it is quite probable that they will be exposed to the varied complexities of one or more organizations. For prestige purposes, it is more likely that they will seek larger organizations, either those associated with grants or more positions. Within these complexities, and as they peel away the layers to do productive work, they often will be forced to use a *Base Class* framework. From these layers, the functionally specific code can be inherited or derived and, whether it is text-based or multimedia, the engineering must be efficient [14].

In an "on-ground" course, the lectures about OO programming and *Base Classes* are limited to the instructor's delivery. The lectures may not be fully prepared. There is usually some gain by those that hear the lecture and a loss by those that do not. In an online education course, the lectures are archived to some degree and reusable. However, some faculty may find this a threat to their promotion or tenure [8]. In terms of course structure, a class can possess some of the attributes of object-oriented concepts such as abstraction, polymorphism, inheritance and encapsulation. However, if the institutional IT infrastructure is not scalable, some risks to growth can be incurred [6]. The organization must be ready to grow and educational objects can be a facilitator.

ENABLING A FOUNDATION CLASS, CLASS LIBRARY

Pooling the experiences of these categories of instructors could prove prolific. This is especially so if they can provide examples that approximate both the online educational framework as well as non-educational organizations. The first can speak from successes, the second can speak from challenges, and the third can validate theories. As they develop lectures, the online-based academic organization can compile those insights and provide educational objects to other instructors. Subsequent instructors can asynchronously review the lectures of the previous staff and learn from those experiences. Even as student questions come up (perhaps about the importance of protecting "core" areas) the instructors can be in a better position to respond. And, as a relevant example, they could relay their practical experiences, using an educational object based class.

As with reusable code segments, documents that detail practical development experiences can be presented to students repeatedly. In many ways this represents a type of organizational memory. The practical experience can be encapsulated into educational objects until they can be applied to other problems. The lectures can be tailored to functionally specific questions in a polymorphic-like way. Lastly, some abstract lectures can be simplified, until substantiated and instantiated with practical examples provided by one or more instructors.

CONCLUSION

Enterprise frameworks, especially those of large organizations, can be quite complex. With high employee turnover, it is possible that few developers would be available to transfer knowledge (or organizational memory) to others who seek to understand the framework. This is especially so as frameworks mature and the original developers are pulled away to work on functionally specific tasks.

A detailed knowledge of at least a few frameworks would be very beneficial to students, and educational objects can be used as examples. Online education, unlike its "on-ground" counterpart, can provide a valuable framework example for teaching students. Those with only small company experiences can

reexamine the software they develop from a more OO-like approach. An important focus could be achieved by using examples from a large company, or at least the educational framework from which they were taught.

Further research could be done to articulate specific OO concepts with applications to online education and information technology curricula. Studies could also be done on how organizations are impacted as frameworks mature. As suggested above, the educational framework can be used to teach enterprise frameworks. The need for framework experience could be learned and applied from a "user" perspective. The technology has finally arrived to facilitate a quicker delivery of results, while incorporating the practical experience of seasoned instructors.

REFERENCES

1. Armstrong, D. (2006). Quarks of Object-Oriented Development. *Communications of the ACM, 49*(2), 123-128.
2. Booch, G. (1994). *Object-Oriented Analysis and Design with Applications* (2nd ed.). New York: Addison-Wesley.
3. Christensen, H. & Caspersen, M. (2002). Frameworks in CS1: A Different Way of Introducing Event-Driven Programming. *Proceedings of the 7th Annual Conference on Innovation and Technology in Computer Science Education ITiCSE, Aarhus, 34*(3), 75-79.
4. Hammer, M. & Stanton, S. (1999). How Process Enterprises Really Work. *Harvard Business Review* (November-December), 108-118.
5. Kim, E. & Schniederjans, M. (2004). The Role of Personality in Web-based Distance Education Courses. *Communications of the ACM, 47*(3), 95-98.
6. Laws, R., & Howell, S. (2005). Scalability Factors in Online Learning. In M. Khosrow-Pour (Ed.), *Managing Modern Organizations with Information Technology: Vol. 2.* (pp. 1366-1369). London: Idea-Group.
7. McKinney, D. & Denton, L. (2004). Houston, We Have a Problem: There's a Leak in the CS1 Affective Oxygen Tank. *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education SIGCSE, Norfolk,(36)*1, 236-239.
8. Schell, G. (2004). Universities Marginalize Online Courses. *Communications of the ACM, 47*(7), 53-56.
9. Schuff, D. & St. Louis, R. (2001). Centralization vs. Decentralization of Application Software. *Communications of the ACM, 44*(6), 88-94.
10. Seffah, A. & Metzker, E. (2004). The Obstacles and Myths of Usability and Software Engineering. *Communications of the ACM, 47*(12), 71-76.
11. Thompson, J. (2003). *Organizations in Action: Social Science Bases of Administrative Theory.* Somerset, NJ: Transaction.
12. Uden, L. (2003). An Engineering Approach for Online Learning. *Journal of Distance Education Technologies, 1*(1), 63-77.
13. Ward, R., Laitinen, M & Fayad. M. (2000). Management in the Small. *Communications of the ACM, 43*(11), 113-116.
14. Zhang, D., Zhao, J., Zhou, L. & Nunamaker, J. (2004). Can E-Learning Replace Classroom Learning? *Communications of the ACM, 47*(5), 75-79.