INTRODUCING SOA AND WORKFLOW MODELING TO NON-TECHNICAL STUDENTS

Bruce J. Neubauer
Public Administration Program, Government & International Affairs Department
University of South Florida
Tampa, Florida 30060
813 974-7630
neubauer@cas.usf.edu

ABSTRACT
Introducing workflow modeling into a course can create new pedagogical challenges. This article reports experiences with two lab assignments involving workflow modeling using AquaLogic™BPM Designer 5.7 software. The author reports experiences teaching workflow modeling in the context of a graduate course in public information systems. Similar challenges may be experienced in the context of MIS and computer programming courses.

INTRODUCTION
Flowcharts, data flow diagrams and sequence diagrams are widely taught in computer science programs to enable students to master the analysis and design of new software applications. In many computer science and software engineering programs the mastery of business processes has largely been considered out of scope. Generally, the expectation has been that business analysts have responsibility to work with managers and end users to assess business needs. Those needs then become incorporated into written specifications before computer programmers and other IT specialists assume responsibility for implementation and testing of actual applications.

Computer science departments and others now face the challenge of embracing Service Oriented Architecture (SOA). The challenge of teaching SOA is that it
requires programmers and generalist managers to learn to think of what organizations do in terms of business processes, workflows and "services" [1, 4]. The notion that "services" can be a convenient bridge between the perspectives of generalist managers and computer programmers is doubtful. In fact, both groups of students may find "services" to be unfamiliar and not particularly intuitive.

This paper reports the initial experiences of the author introducing Web services, SOA and workflow modeling to a class of graduate students in a public administration program in the United States. The following sections of this paper include a brief overview of Web Services and SOA, followed by explanations two computer lab assignments; information about the software used, the observed experiences of students and suggestions for teaching workflow modeling.

WEB SERVICES AND SERVICE ORIENTED ARCHITECTURE

From a technical perspective, a Web service is software that can be accessed using particular technologies including SOAP, WSDL and UDDI [1]. The major technical benefit of Web services are that they are platform-independent and can be accessed using freely available resources rather than by using more specialized or proprietary resources [4]. The downside is that these distributed applications tend to run relatively slowly, in part because of a dependence upon XML. Web services enables specific functions of legacy systems to be exposed in a way that facilitates the construction of applications that span across departmental "stovepipes" and can be the basis of strategic networking among organizations.

Service oriented architecture (SOA) refers to the design of new crosscutting applications that involve the incorporation of "services" exposed by multiple existing systems. It is like component-based software architecture, except the idea is to think at the level of business services rather than more fundamental software components. While it is difficult to clearly draw the line between services and components, services are relatively "chunky" and are meaningful to general managers, for whom technical components may not be comprehensible. For example, to the members of an admissions department, "evaluate academic credentials" might be a service because it is a bounded set of activities within the context of a business process. An organization could package that sub flow of computer-supported activities with an eye toward possibly outsourcing it to a more specialized organization, or possibly even removing the human discretion and automating the entire activity. For example, colleges and universities frequently have to evaluate academic credentials earned at institutions in other nations and cultures. The prospect of receiving this work as a specialized service performed by a specialized organization, using modern information technology to integrate systems, may be attractive.

The design of software systems has tended to focus upon supporting departmental functions in organizations rather than business processes that cross multiple departments [3]. While such applications can certainly benefit the productivity of departments it may tend to contribute to situations in which the handoffs required to perform business processes are not optimal. Some traditional applications have sometimes been referred to as stovepipe applications or as systems
contributing to organizational silos. Most large organizations have large investments in such information systems and cannot or will not walk away from such systems. The current emphasis on the need to optimize business processes rather than business functions creates a need to create information systems that somehow tap into existing systems rather than creating an additional set of competing or redundant applications within the organization [2, 5].

By blurring the distinction between designing software applications and designing organizations, SOA creates an opportunity to begin helping students learn to work across the organizational divide between generalist managers and technical specialists. Workflow redesign is at the intersection created by SOA [2]. The ability to think in terms of "services" is a challenge to both managers and programmers. Services are building blocks that relate to activities in organizations and to the business functions supported by software systems. SOA is the key to the convergence of organizations and software and to the challenges of achieving alignment of human and machine systems. From the perspective of management theorists, this challenge is on the order of making the mental transitions from thinking of organizations entirely in terms of bureaucratic hierarchies to thinking of organizations as fluid networks of strategic alliances.

From the perspective of software engineers, this challenge may be the equivalent of the transition from structured procedural programming to the object-oriented paradigm, tending to force a closer association between programming and organization theory. The pedagogical challenges are clearly in the arena of those who teach management information systems, as well as those who are more "on one side or the other" of the distinction between teaching programming and teaching organizational design/management. In the author's opinion, the teaching of workflow (re)design using computer-based modeling tools is the appropriate respond to the pedagogical challenge at this intersection between application design and organization design.

**LAB ASSIGNMENTS**

The author approached BEA employees asking for help in using their CASE tools for teaching purposes. The author had previously explored Microsoft BizTalk Server and the beta of Microsoft's Windows Workflow Foundation with the intent of using Microsoft software in the classroom. The author found BEA's AquaLogic Designer 5.7 (previously part of the Fuego 5.5 Suite prior to a merger between BEA and Fuego) to be more feasible to be used to introduce students to workflow design. AquaLogic Designer 5.7 is part of the AquaLogic Suite, including developer software that produces source code from visual models. Both the designer and the entire suite are presently available on the BEA web site for sixty-day trial download. With the knowledge and permission of BEA representatives, the author installed AquaLogic Designer 5.7 trials on each machine in a university classroom/lab. The installation requires the prior installation of a recent Java Virtual Machine. The installations were not difficult and took about twenty five minutes per machine.
The author as teacher faced a personal learning curve that included three significant hills. The first (and easiest) was the need to teach students what business processes are and how to approach business process (re)design. This is familiar territory in MIS, including the literatures related to business process reengineering. The students read part of a textbook by Harmon [3] that introduced them to basic concepts such as the identification of sequential activities, organizational roles, and visual models involving "swim lanes." To go beyond this level of instruction was more difficult.

The second hill involved becoming familiar with the software itself. The author paid for live online training involving the use of the software. BEA offers such courses using Microsoft Live Meeting. Some students are physically present with the instructor and others are remote participants, using application sharing, the telephone, and virtual PC's for completing labs. In the author's experience, this instruction was helpful but not adequate to master the software.

The third hill on the learning curve involved how to actually think through the visual modeling of a real business process and to make the cognitive connections to SOA for pedagogical purposes. There is clearly a potential in this area to add value to the course on public information systems for students in a graduate public administration program. There is a large overlap between the elicitation of knowledge needed to build traditional software applications for organization and the elicitation of the knowledge needed to create service oriented architectures, and to represent them in visual models. BEA does provide a set of visual representations of business design patterns, which is valuable. The case studies available online tend to be reports of successful installations rather than explanations of how new systems are designed.

In the spirit of two previous lab assignments in this course involving the students' creating a Web service and then a Web application consuming a Web service, the author wrote a step-by-step assignment in which the students created a visual model using AquaLogic™BPM Designer 5.7. Each students exported the result of their work as a specialized file type, and sent the resulting file to the commercial Web server using an FTP client. The written instructions for this lab were supplemented by a multimedia presentation made available to the students on the Web. The students breezed through the assignment with very few problems. In retrospect, this led the author to a false anticipation that the next lab assignment would also be relatively easy for the students to complete.

The author then prepared a lab assignment asking students to create the "As-Is" model regarding how the U.S. Federal Elections Commission (FEC) responds to alleged violations of election laws. The description of that process is available at the following URL. http://www.fec.gov/pages/brochures/complain.shtml
It was the author's intent to then give the students an assignment involving the creation or a "could-be" model, by introducing automation and/or Web services to an "As-Is" model. This additional intended assignment was not attempted.
SOFTWARE USED

Figure 1 shows the AquaLogic™BPM Designer 5.7 interface. The first thing one might notice is the use of vertical rather than horizontal swim lanes. In most textbooks business process swim lanes are horizontal. See Harmon [3] for an example. AquaLogic™BPM Designer 5.7 will display them either way. It is said to be simply a matter of preference, although reason to prefer for the somewhat less intuitive vertical swim lanes is not apparent. Those familiar with use case diagrams may prefer horizontal swim lanes, because Actors correspond to roles and each role gets its own swim lane. In use case models, Actors are usually displayed to the left of the symbols representing use cases.

There are also multiple "themes" of symbols, including UML symbols. The names of symbols are pretty intuitive, although the name for an activity is an "Interactive." An activity performed by a computer system without human intervention is called an "Automatic." The look is very similar to Microsoft products, although the behavior of the software may take some getting used to. Designers can use a check function to find certain kinds of errors in their models. The final model can be exported and the resulting file then imported into AquaLogic™BPM Developer 5.7. It is possible to include multiple sub flows in one model and then drill down into sub flows for additional detail, similar to the way that an Excel Workbook can contain multiple sheets representing different levels of drill-down.

STUDENT EXPERIENCES

The students had little or no difficulty with a computer lab assignment that walked them step-by-step through the creation of an existing model and exporting it as a file that could be submitted as evidence of completion of the assignment. However,
when asked to create the As-Is model of a moderately complex business process, they had difficulty. The author's attempts to prepare them for the task by describing it and by showing them analysis design patterns were not adequate preparation for the assignment. There were at least three major kinds of problems the students faced in the second assignment.

First, students tended to approach the task "head on" rather than first stepping back from it and gaining a perception of the "big picture." As a result, their models tended to be very complicated and too detailed. Although the author as teacher showed the students examples of use of sub flows to "chunk" groups of related activities, students tended not to use sub flows. As a result, their models tended to be visually overwhelming. It is unlikely that the code produced by such a model using a CASE tool would be a useful step toward coding an actual application.

Second, the students and the author had some difficulty with the behavior of the software itself. While the software is visually intuitive, it frequently automatically creates patterns of transitions between interactives (activities) which are not the intention of the person using the software. This behavior is annoying and forces the user to frequently delete transitions and manually create the intended conditional and unconditional transitions. While this behavior by the software may in some cases be an intentional feature that assumes an experienced user, it can be very annoying to inexperienced modelers. It would be nice if this behavior could be turned off.

Third, and most importantly, the students needed more specific instruction in how to model business processes. Showing the students many examples of models of business processes would have helped them in this regard. While there is value in learning through "immersion" in a challenging assignment, these students had not seen enough examples of good models prior to their trying to complete the second computer modeling lab assignment.

CONCLUSIONS

Introducing students who do not have a programming background to business modeling using a CASE tool proved to be both rewarding and challenging. The students, many of whom are mid-career professionals, affirmed the importance of understanding and modeling workflows. Employees of BEA Corporation were very responsive to the requests of the author/teacher. One BEA employee met with the class twice using WebEx conferencing. The use of step-by-step instruction supplemented by audio-video demonstration of the assignment online (using Camtasia Studio) enabled students to use the visual modeling tool quickly. But the full intent of the two lab assignments involving visual modeling of workflows was not achieved. The use of more examples in class and more explicit materials about how business analysts actually elicit business rules and model business processes probably would have led to a better pedagogical outcome.
REFERENCES


