SENIOR PROJECT DESIGN METHODOLOGY

Ali Sekmen[1], Ismail Fidan[2], and Ahmet Koku[3]

[1]Department of Computer Science
Tennessee State University
Nashville, TN

[2]Department of Manufacturing and Industrial Technology
Tennessee Tech University
Cookeville, TN

[3]Department of Electrical Engineering
Vanderbilt University
Nashville, TN

Abstract

Senior and Capstone Design Project courses are widely adapted in engineering and computer science curricula to prepare students for the industry and give them the opportunity to work in real-world projects. One of the motivations behind offering such courses is to satisfy the ABET criteria emphasizing the importance of functioning in multi-disciplinary teams. Senior project courses in computer science/engineering are usually designed to be one or two semester-long and they traditionally involve large-scale software systems development. There is also a tendency to assign hardware/software integration projects to prepare students for the competitive job market. This paper introduces an innovative approach to enable students to work in multi-disciplinary teams and develop large-scale software/hardware systems. Programming language and operating system independent component-based software development is extensively discussed and explained with some complex senior project examples.

1. Introduction

Senior and Capstone Project Design courses are very special courses that provide students with the opportunity to work in large-scale team projects. The expectations from those courses can be summarized as *gaining experience on working in interdisciplinary teams, forcing to use the accumulated curricular background, interactive learning, gaining project management skills, and developing written and oral communication skills.* These expectations overlap with some of the attributes that engineering graduates must have in order to satisfy some of the ABET criteria [1,2,3]. *The ability to function on multi-disciplinary teams* is one of the required attributes of engineering graduates [1]. Working in interdisciplinary team projects may be very difficult for students if the overall task is not equally divided among the group members and each group member does not have the flexibility to work independently on his/her part.

Senior and Capstone Design Project courses for Computer Science/Engineering departments almost always include large-scale software systems development. These projects may be on software development or hardware software integration especially for Computer Engineering students. The main obstacle in implementing large-scale software systems among multi-disciplinary team members is the lack of use of the software environments that enable to develop independent but highly coupled software components. Students tend to use the programming tools that they are familiar with and this may lead a conflict on deciding what tool to use as the whole group. Most of the projects require developing interrelated software pieces that need to communicate each other. For instance, implementing a voice-controlled navigation system for a mobile robot involves *low-level hardware control, speech recognition, behaviors such as obstacle avoidance, and integration of those with a graphical user interface.* The student working on obstacle avoidance behavior needs to use the low-level hardware control developed by another student. The student working on speech recognition should be able to easily integrate it to the low-level hardware control.

This paper investigates the efficient means of developing large-scale software systems in an interdisciplinary environment. It is proposed that Senior or Capstone Design Project students should be forced to follow component-based software system development approach in order to get all of the group members equally involved. Such a software development technique let students work independently yet provides very elegant means for software component integration. This paper is organized as follows: Section 2 gives a detailed description of the problem. Component-based software systems development is discussed in Section 3. Some sample senior project samples along with proposed solutions are given in Section 4. Finally some conclusions are drawn in Section 5.

2. Problem Statement

Senior and Capstone Design Project courses can nicely fulfill the ABET criteria emphasizing the importance of interdisciplinary teamwork if the followings are satisfied:

- The task among the team members should be distributed evenly
- The team members should be able to have the flexibility to work independently as much as possible
- Yet, they should be able to integrate their developments easily

This is extremely important especially for software systems development. One group member should be able to integrate a software component developed by another member to his software without having to know any implementation details of this component. Otherwise, the teamwork will progress slowly since group members need to spend time to learn the details of each other's developments. The current problems that prevent efficient teamwork in Computer Science/Engineering Senior and Capstone Design Project courses can be listed as:

- Students do not usually use component-based software systems development approach. Hence they get into problems at the software integration level.
- Students tend to use the same programming language. This helps in integrating their software pieces. However, since there is not a single programming language that is

suitable for all purposes, this usually leads inefficient teamwork. For example, Visual Basic is a very suitable language for graphical user interface development whereas C++ is more efficient for real-time hardware control. Java may suit best for Internet-based developments. In addition, not all of the team members may have expertise on the same programming language. This forces some of the team members to use a programming language that they do not feel comfortable.

- Students tend to develop large amount of source code since they cannot make efficient use of existing software components that are freely available. They do not usually know how to integrate those components to their software.

The solution to the problems mentioned above is the use of programming language and preferably operating system independent component-based software systems development approach as described in the following section.

3. Component-Based Software Development

Programming techniques have evolved from low-level procedural structures to high-level component based distributed architectures as shown in Table 1. Component-based programming possesses several major advantages over conventional programming. It makes fast integration of software component of different complexity easier. This is extremely important for large-scale software systems development. For example, let us assume that a team of five (5) students works in a team project to develop a voice-activated seek-and-destroy system using a mobile robot. This requires implementation of four (4) major software components – speech recognition, text-to-speech, image processing, and low-level hardware control for the robot. The vocal command given by a user is recognized by the speech recognition component that triggers a seek-and-destroy behavior on the robot. Seeking is accomplished by the image-processing component. When the target is found, the text-to-speech component informs the user and waits for further commands. If the user asks the robot to destroy the target, the laser pointer attached to the pan-tilt head of the robot points to target to simulate destroying. The components' interaction is shown in Figure 1. The overall task should be distributed among the group members as follows:

- One student develops low-level hardware control for the robot.
- One student develops image-processing component including image grabbing from the camera(s).
- One student develops speech-recognition component. He/she may make use of existing speech recognition engines.
- One student develops text-to-speech component. He/she may make use of existing text-to-speech engines.
- The last student takes all of the components developed by the other students and integrates them to build the overall system. He/she is called as **manager** in Figure 1.

In component-based software systems development approach, each student can almost always use the programming language and operating system of his choice. Several different technologies such as COM (**C**omponent **O**bject **M**odel) and CORBA (**C**ommon **O**bject **R**equest **B**roker **A**rchitecture) allow researchers to develop programming language and operating system independent software components that can be upgraded without breaking the overall software

compatibility [4,5,6]. One developer does not have to know implementation details of the components developed by some others. This gives flexibility to work as a team on the same project. For example, an ActiveX (a COM component) written in Visual Basic can be called from Visual C++ or Visual J++. The components can be considered as black boxes and the project developers simply need to know the inputs and outputs of these components. Within the component-based software systems paradigm, numerous pluggable software components have been created and some of them are even integrated with operating systems to facilitate programming (such as DirectX released by Microsoft).

Table 1. Analysis of programming techniques.

	Low Level	**Procedural**	**Object Oriented**	**Component Based**
Ease of use	Low	Low	Medium	High
Inter-Operability	None	None	Low	High
Hardware transparency	None	None	Low	High
Ease of Distributivity	None	Low	Medium	High

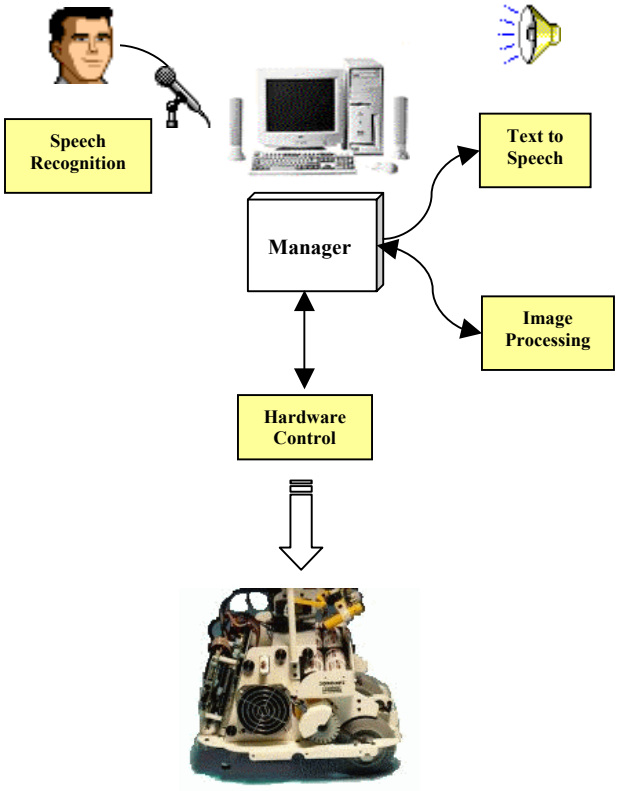

Figure 1. Component interaction in seek-and-destroy behavior.

Web Services is another technology that provides programming language and operating system independent software component development and communication. In other words, it provides software interoperability and extensibility. For example, a software component written in Java programming language under Linux platform can easily communicate with another software component written in Visual Basic under Windows platform. J2EE and .NET are two commonly used middleware to develop Web Services.

4. Sample Projects and Proposed Methods

Senior Project course offered by the Department of Computer Science in Tennessee State University proves students with a variety of project selection. Many of the projects involve software systems development and hardware/software integration. The common approach that students are suggested to use is the component-based software development as explained previously. The overall task is very clearly divided among the team members so that they can individually work and then integrate their work to produce the final product. Some examples are given below:

Voice-Controlled Surveillance Robot: It is evident that the current trend in robotics research is to make mobile robots an integral part of our daily life. An important criterion to achieve this goal is the socially acceptance of these robots. Among many factors that facilitate this is the natural human-like communication. As the communication between human and machines becomes more human-like, they will be more integral parts of our daily life. In this project, a robot with a wireless camera and various sensors attached is commanded by a human user via speech. For example human commander may tell the robot to go to the other room. While the robot autonomously moves, it starts broadcasting live images over the Internet to the security office if an intruder is detected. The security officer can also warn the intruder via the speech synthesizer of the robot.

A component-based architecture is developed for this project by using COM technology. Four (4) team members develop four software components. First is the low-level hardware control component for the robot [7]. Second is the camera component for grabbing and broadcasting images. Third is speech recognition component by using Microsoft's Speech Engine. Fourth is autonomous navigation component. The team members can independently work on each component and then integrate their work. For example, the developer of the autonomous navigation component can use the low-level hardware component and camera component without having to know any implementation details. He/she does not even care about the programming languages in which they are developed.

Distributed Pattern Recognition System: There are several reasons for motivating distributed software systems. First of all, it is a means to increase computational power by making use of several processors. For example, image processing is a powerful tool and is widely used in robotics for object recognition, tracking, obstacle avoidance and etc. However, it is two-dimensional processing and computationally expensive. If several image-processing algorithms need to be run in a robotics application, it may be required to distribute them on several processors. Second motivation for distributed systems is the distributed hardware. Third reason is to decrease the failure rate. If the hardware is distributed over several computers, the system can

be kept in a safe state even though some of the hardware and/or software crash. In this project, a distributed pattern recognition system that uses three (3) cameras attached to three (3) mobile robots is developed. Pulse-Coupled Neural Network (PCNN) for feature extraction from images captured by the cameras. The remote recognition decision results were transferred to a central processing machine using COM/DCOM technology via Internet connection. A central fuzzy logic system was designed to make the final decision about the object recognition.

This project requires development of several software components. First is an ActiveX control or ActiveX DLL for grabbing images. Second is an ActiveX control that can create an instance of MATLAB and make use of MATLAB's image processing toolbox. Third is an ActiveX exe that is encapsulated the first two components to create instances of those components on remote machines.

Human-Robot Interaction via PDA (Personal Digital Assistant): PDAs or handheld computers - with increasing speed and computational power, wireless capabilities, and elegant size - have very promising future in robotics. In this project, a highly mobile security system is developed by using a mobile robot and a PDA. A security officer can remotely command the mobile robot via a PDA. Both security officer and robot are mobile; this leads a very efficient security system. In this project, several software components are developed. Hardware control component, socket communication component, image transfer component, and a graphical user interface for PDA.

A Gadget For Helping Instructors: Most of the instructors need a software tool that keeps track of their students. A software package that provides instructors with full flexibility to see and change all the information about their students would be very useful. For example, when an instructor grades final tests, he should be able to use a GUI to store the grades into a database. This GUI should provide all the information about the students in the database including their pictures so that the instructor would know who the student is. This package should also give an option to send emails to the student(s). In addition to having a standalone software package, it would be very useful for the instructor to access his/her students' database over the Internet with password protection. The instructor should be able to modify some information about his/her students over the Internet. Moreover, the instructor may not like using Microsoft Internet Explorer or Netscape and may ask for a custom design web browser for accessing the database over the Internet. In this project, the team members develop the following software components: database connectivity component, web browser component, email component, and etc.

5. Conclusion

Senior and Capstone Design Project courses for Computer Science/Engineering departments can provide students with better team working skills if students are asked to use component-based software systems development methodology. Today's software technology provides a lot of environments for such software development. The most commonly used technologies are COM, DCOM, CORBA, and Web Services. Two of the commonly used middleware are Microsoft's .NET and Java Enterprise (J2EE). The approach proposed in this paper will also help educators to satisfy some of the ABET criteria.

Bibliography

1. ”2003-2004 criteria for accrediting engineering programs”, Accreditation Board for Engineering and Technology, November 2002 (http://www.abet.org/criteria.html).
2. ”2003-2004 criteria for accrediting engineering technology programs”, Accreditation Board for Engineering and Technology, November 2002 (http://www.abet.org/criteria.html).
3. ”2003-2004 criteria for applied science programs”, Accreditation Board for Engineering and Technology, November 2002 (http://www.abet.org/criteria.html).
4. D. Rogerson, *Inside COM*, Microsft Press, 1997.
5. T. Pattison, *Programming Distributed Applications with COM+ and Microsoft Visual Basic*, Microsoft Press, 2000.
6. J. Pritchard, *COM and CORBA Side by Side: Architectures, Strategies, and Implementations,* Addison-Wesley, 1999.
7. A.B. Koku, A. Sekmen, and S. Zein-Sabatto, “Development of a software package for teaching robotics”, *Proceedings of SCI2001,* July 2001.

Ali Sekmen, Ph.D.

Ali Sekmen is an Assistant Professor of Computer Science at Tennessee State University. He received his Ph.D. degree in Electrical Engineering from Vanderbilt University, Nashville, Tennessee. He holds B.S. and M.S. degrees in Electrical and Electronics Engineering from Bilkent University, Ankara, Turkey. He has published over 40 research papers in robotics, intelligent systems, and signal processing. He was a member of Intelligent Robotics Laboratory of Vanderbilt University between 1997-2000. Previously, he was an Assistant Professor of Electrical and Computer Engineering at Tennessee State University. He has been involved in research projects including human-robot interaction, intelligent systems, mobile robots, humanoid robots, and component-based software systems development. Dr. Sekmen is a member of the Institute of Electrical and Electronic Engineers (IEEE).

Ismail Fidan, Ph.D.

Ismail Fidan is a faculty member at the MIT department of Tennessee Tech University, Cookeville, TN. He began his academic appointment in August 2000. Dr. Fidan received his PhD in Mechanical Engineering from Rensselaer Polytechnic Institute, Troy, NY in 1996. He is a senior member of IEEE and SME, and member of ASEE, NAIT, ASME, TAS and SMTA. Dr. Fidan also serves as an associate editor for the IEEE Transactions on Electronics Packaging Manufacturing and editorial board member for the Journal of Industrial Technology and SAE Transactions. His research interests are computer integrated design and manufacturing, electronics manufacturing, and manufacturing processes.

Ahmet Bugra Koku

Ahmet Bugra Koku (B.Sc. in ME and M.Sc. in systems and control engineering, Bogazici University, Istanbul, Turkey) received his a Ph.D. degree from Vanderbilt University. He is a senior member in the Intelligent Robotics Laboratory at Vanderbilt. He is a student member of IEEE (since 1997). His current research interests are memory organization of robots, qualitative/topological navigation, low cost Mechatronics equipment design for robotic applications and robotics education.