A Situational Leadership Approach to Managing Larger-Sized Student Teams Working on Web Application Projects

Wilson Wong
wwong2@wpi.edu
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA, 01609, USA

Irv Englander
ienglander@bentley.edu
Bentley University
175 Forest St.
Waltham, MA, 02452, USA

ABSTRACT

A project team capstone course constitutes an important aspect of undergraduate information systems education because it requires students to synthesize the knowledge gained from many other computer courses. Capstone courses often attempt to give students a semblance of real world experience by having them work on substantial software projects. However, few capstone project teams consist of six or more members, team sizes which occur often in the industry. Because these projects with larger teams incur increased project complexity and coordination, such a course is difficult to administer and manage. Course challenges encompass incomplete projects, implementation failures, and negative student reactions embodying conflict, resentment, frustration and feelings of inadequacy. For courses with larger-sized student teams and real-life, loosely-structured projects, we present a pedagogical methodology to improving team management through 1) the use of Blanchard's situational leadership model 2) reduction of team communication complexity, and 3) selection of a software development methodology that improves team member utilization.

Keywords: situational leadership, web development, capstone course, team project, project management, software development methodologies, software development life cycle, Agile methodology

1. INTRODUCTION

Bentley University's Computer Information Systems undergraduate curriculum offers a capstone course, CS460 Applied Software Project Management, which combines classroom learning in project management with a major student team project in web application development. The course integrates software development methodologies, technology, and business aspects of the curriculum to provide

students with real-world experience in the development and management of web projects while working in teams of six to nine students. Each student team is independently responsible for the requirements gathering, analysis, design, and implementation of a project from start to finish, including documentation and presentation of the final application. Although most teams work on the course's pre-determined projects, student teams sometimes have the option to select their own projects with instructor guidance and approval.

The increased complexity arising from coordinating additional team members and their communications means that larger teams face greater challenges to successfully completing their project, especially within a single semester. These difficulties are compounded since most of the students lack experience working on a large team and experience developing an application according to a specified software development methodology, whether it is an Agile methodology or a traditional software development life cycle. In order to improve a team's chance of success, team management must be improved through effective management styles, streamlined internal communications, higher team member utilization, and better project guidance. Not only do the teams need better internal management, but their larger size means that the instructor needs to manage the teams externally at a higher level mirroring the hierarchical structure within a software company.

We propose a three-part pedagogical model to improving the manageability and success of larger-sized team projects. The first component is the instructor's application of the Blanchard Situational Leadership II model so as to employ the most effective management style for a team given their experience and motivation (Blanchard & Hersey, 2008). The second component is the streamlining of internal team communications by carefully reducing the number of two-way communication paths. The final component is the selection of a software development methodology that better supports a higher level of team member utilization. This three-part approach has helped the teams achieve successful outcomes and provided students with a high level of satisfaction and a strong belief in their own capabilities when they transitioned to the business world.

Over the course of six years from 2009 to 2014, student project teams have created web applications for non-profit organizations and the general public. These projects have included applications to simulate the organ transplant matching process, guide tourists along Boston's Freedom Trail, assemble physicians and medical equipment in a hospital, and administer a non-profit organization's program to provide Afghani citizens with online English lessons given by volunteers around the world.

2. LITERATURE REVIEW

The concept of an information systems capstone course with a real-life team project is well established in the literature (Gackowski, 2003; Keogh, Sterling, & Venables, 2007; Leidig & Lange, 2012; Lynch, Heinze, & Scott, 2007; Magboo & Magboo, 2003; Mew, 2014; Reinicke, Janicki, & Gebauer, 2012; Russell & Russell, 2014). The advantages of real-life team projects are the opportunities for students to fuse their knowledge of software methodologies, project management concepts and software development environments; engage in experiential learning (Keogh, et al., 2007) and increase their motivation and confidence in their abilities (Grant, Malloy, Murphy, Foreman, & Robinson, 2010).

Despite the advantages of student project team experience and the availability of various guidelines for conducting capstone project classes such as Keogh's student team project framework (Keogh, et al.,

2007), there is evidence that such classes remain difficult to administer, manage, and assess. Results can include failed or inadequate projects, uneven work effort, logistical problems among members of a team, student resistance to heavy workloads, and student dissatisfaction (Brandon, Pruett, & Wade, 2002; Lynch, et al., 2007; Mew, 2014; Vyver & Lane, 2003). Research on the outcome of student design projects in multiple universities have found a significant failure rate (Eckerdal, McCartney, Mostrom, Ratcliffe, & Zander, 2006; Loftus, Thomas, & Zander, 2011).
Some of the difficulties encountered by students may be attributed to the fact that information systems capstone courses can be pedagogically different than typical programming classes taken earlier in the curriculum. Rather than individual mastery of specific skills, success in a capstone course with a real life project requires collaborative teamwork (Keogh, et al., 2007) which presents a far more complex working environment than solo programming.

An aspect of team complexity can be seen through the lens of communications. Team communication complexity can be measured by the number of two-way communication paths that can exist within a team. Thus, team communication complexity can be calculated using the combinatorics formula $\binom{n}{2}$, or ½(n^2 – n) where n is the number of members on a team (Futrell, Shafer, & Shafer, 2002). As a result, team communication complexity can increase exponentially by the square of the number of team members if the team is not properly structured! For a four person team, there are only six two-way communication paths. However, for a nine person team, there are 36 two-way communication paths!

A common solution to reduce complexity and improve positive team outcomes is to offer smaller projects requiring fewer team members. In this light, it is not surprising that real-life capstone projects are typically done by small teams of two to five students (Holifield, Longenecker, & Feinstein, 2012; Leidig & Lange, 2012; Mew, 2014; Pollacia, Ding, & Yang, 2014; Tappert, Cotoranu, & Monaco, 2015). Other solutions are to limit the scope of the project to a tightly designed set of requirements provided by the instructor, or to use specifically designed case study assignments in place of real-life team projects (Brandon, et al., 2002; Tatnall & Reyes, 2005). However, each of these solutions limits the effectiveness of the team project experience.

A solution that would permit larger team sizes and project scopes would be to improve the management of the student teams in a way to increase their effectiveness. The Blanchard Situational Leadership II model proposes that the four basic leadership styles are directing, coaching, supporting and delegating (Blanchard & Hersey, 2008; Blanchard, Zigarmi, & Zigarmi, 1985). Rather than selecting a single leadership style as the best one, the situational leadership II model asserts that the manager's appropriate choice of leadership style is contextually dependent on the task competence and psychological commitment of the reporting individual. A directing leadership style, where the manager engages in one-way communication that dictates the details of how to accomplish the task, is appropriate when the subordinate has low task competence accompanied by moderately high commitment to the task. Individuals participating in new projects typically exhibit traits of low task competence and moderately high commitment. The coaching leadership style, where the manager consults with the subordinate and engages in two–way communication but makes the final decision, is most applicable when the subordinate has low-moderate task competence and low commitment. This situation often arises after the

beginning of a project when the subordinate becomes disillusioned as a result of a gap between initial expectations and reality. As the subordinate gains a level of moderate to high task competence, negative feelings dissipate and commitment improves although it is still variable. A supporting leadership style, where the manager shares the decision making with the subordinate and provides moral support, is the most suitable course. The final leadership style, delegating, is employed by managers when the subordinate has achieved a very high level of task competence combined with a high commitment to the task and is able to work autonomously. Managers employing a delegating leadership style transfer most, if not all, relevant decision making to their subordinates. The Blanchard situational leadership approach is considered applicable when applied to teams working in a high technology environment because they benefit from adaptive leadership styles (Silverthorne & Wang, 2001). This suggests that the situational leadership approach would be effective in an information systems project capstone.

3. WEB CAPSTONE COURSE

3.1 Primary Course Objectives

The primary purpose of the course is to provide students with real-life experiences that correspond to their expected duties and performance in the workplace. To do so, students participate in a real-world, loosely-structured software project as members of software development teams. The project teams start at ground zero and proceed through all the stages of the selected software development methodology. Each team is responsible for all aspects of its project, including application development, management of the project, and internal management of the team. The deliverables include working software, full project documentation, and a class presentation of the project by each team. Although the software development project is a primary objective of the course, there is also a related and synchronized classroom learning component that focuses on the various aspects of project management.

3.2 Student Background

Because students have access to different electives within the Computer Information Systems program, students participating in CS460 have different interests, strengths, and abilities. To register for the course, all students must have a knowledge base that includes database system design, Java programming, system analysis and design, and computer architecture. Although not required, many students have experience with web-based technologies such as HTML5, Cascading Style Sheets, JavaScript, and JQuery. Not surprisingly, we see a wide variation in student capability and interests. Some students will be strong web developers, others will be more interested in back-end database development; still others will be more focused on project management, to name just a few of the possibilities. This diversity is a valuable contribution to the criteria used for team selection, since the goal of team selection is to reflect the real world as realistically as possible.

3.3 Class Organization

As noted above, CS460 consists of two primary components, classroom lectures that introduce the various project management tools that support project development in the workplace, and the team projects themselves. Table I presents the major topics presented in the classroom setting. These topics correspond to the specific project tasks for which the team is responsible and are presented as closely as possible to the order in which they will be needed for team project work.

To accommodate both the classroom lectures and team project work, class periods are divided between lectures by the instructor and team meetings. Putting together a schedule that balances team

meeting sessions with lectures and presentations is challenging because project management material must be presented early enough so that students can apply it in managing their projects, but students need to meet as early as possible so as to have enough time to complete the project. The majority of the course concepts were presented in lectures in the first half of the semester except for except for topics that became necessary after the start of implementation – quality assurance, risk management and project tracking and control. Eleven of the twenty-six class sessions lectures are software project practicums where teams apply the lessons learned in previous classes. After the few remaining lectures are completed in the second half of the semester, the final exam is given approximately 3 weeks before the final project presentations, which take place at the end of the semester. This balance presents most of the pertinent information early enough for the teams to utilize it and gives them unfettered time in the three weeks before the project is due.

Table 1. Class Lectures

Managing software project teams
Requirements gathering
Project charter and software project management plan
Software development methodologies
Work breakdown structure
Software size, duration, and cost estimation
Project scheduling, including PERT and Gantt charts
Program verification and validation, dynamic testing
Risk analysis and management
QA Management
Project tracking and control

4. PROJECT METHODOLOGY

At the beginning of the semester, the instructor presents an overview of the general goals to be achieved by the capstone project, much as a director of software development might. The instructor may either give all teams the same pre-determined project or offer a selection of possible projects with the option of creating their own project. Part of the exercise for each team is to define the specifics of their projects through the process of gathering software requirements.

The instructor employed the Blanchard Situational Leadership II model to manage the teams at a high level. The four phases of the situational leadership model – directing, coaching, supporting, and delegating – were matched with the four quarters within a course semester.

The instructor began by adopting a directing leadership style to select individuals, based on their answers to a background survey, and placed them on teams ranging from six to nine students, depending on the size and difficulty of the anticipated project and the qualities and capabilities of each student. A directing leadership style is appropriate at the beginning of the course because students lack much experience working in a software development project, and their teams are in a formative stage. Initial software requirements were jointly determined by the client and the instructor and then given to the teams. Students were then directed to produce specific project management artifacts such as a software project charter, organizational chart, and a feasibility analysis. The process by which these documents were to be completed was carefully specified.

In the second quarter of the semester, a coaching leadership style is employed while determining the full scope of an appropriate sized project. Teams competed to create the best application, so each team brainstormed and prioritized a different set

of additional features that would enhance their project. The instructor applied a coaching leadership style by incorporating the team's suggestions for enhanced features, but making the final decisions on which features would be attempted. The scope of the application would be increased or decreased to match the capabilities and size of the team.

In the third quarter of the semester, the instructor switched from a coaching style to a supporting one. A difficult challenge was deliberately included for each team's project that required them to resolve the issue by learning new technology and engaging in problem solving. The teams typically needed to address the technical challenge after they had successfully created project management and design documents, and had passed the coaching stage by demonstrating a moderate level of task competency and an improving but variable level of commitment. A typical comment by a student at this stage would be "it's impossible to get this working!" At this point in the semester, the instructor would employ a supporting leadership style by informing the teams of his confidence that they would be able to solve the problem and that all previous teams had been successful, no matter what they had encountered. However, it would be up to the team to solve their predicament. The team's eventual successful solution to the problem is a critical component in transforming the team into one that can work autonomously and giving students confidence in a future situation in the industry. At this point, sometime around the fourth quarter of the semester, the instructor is finally able to apply a delegating leadership style and leave the teams to work mostly autonomously.

The second component of our pedagogical model was to devise a proper team structure to help the teams decrease internal communication complexity by reducing the number of two-way communication paths. The software development team structure consists of three components – project management, web development and database development. Project management positions included the project manager, quality assurance manager, and the documentation manager. The project lead developer primarily led the web development efforts but would also coordinate with the lead database developer. As a result of this structure, communication was simplified because an individual team member did not have to always communicate with every other member. Web developers and database developers primarily communicated within their groups leaving their leaders to communicate with team members outside the group. Of course, entire team meetings still had to be held on a regular basis but the amount of total communication was substantially reduced. This outcome was confirmed in final peer reviews of fellow teammates.

Careful attention was also paid to the distribution of team members' workloads. A common key difficulty with this team structure is workload imbalance that arises from the quality assurance manager having little to do throughout the semester and then being forced to perform all of the work at the end of the semester when the project is finally implemented. The selection of an appropriate software development methodology is critical to improving utilization of the quality assurance team member. An appropriate choice of software development methodology can be made taking into account whether the course is to use a traditional software development life cycle, an Agile development methodology, or a hybrid development methodology.

For a traditional software development life cycle, the V-shaped software development life cycle (SDLC) is used to balance team workload. A well-known variant of the waterfall SDLC, the V-shaped SDLC re-

orders the testing phase throughout all of the other software development life cycle phases. System tests can be created concurrently with requirements specifications, integration tests can be created alongside the architectural design and unit tests can be written as the detailed design is done (Futrell, et al., 2002). This permits the quality assurance manager of the team to work throughout the semester, rather than waiting until implementation is complete. An additional benefit of the V-shaped SDLC is greater application reliability (Futrell, et al., 2002).

Courses using Agile methodologies inherently increase the utilization of the quality assurance manager because development is performed in multiple iterations, so software testing is done with each cycle. Another approach is to use a hybrid software methodology that is based on an incremental SDLC, where the design of the entire application is initially created, but subsequent groups of features are implemented at regular increments. This software development methodology also permits the quality assurance manager to perform testing with each increment cycle. The authors have applied the V-shaped SDLC, incremental SDLC, and Agile methodologies – particularly SCRUM – to courses over the years and have found all three software development methodologies to be effective in improving overall team member utilization compared to the traditional waterfall SDLC.

Teams were required to produce the following deliverables – the final application, software project management plan, software requirements, design documents, test plans, a user manual, software size estimates, a project schedule, and a risk management plan. Midterm presentations served to give teams feedback on their software designs and project management. The presentations also helped teams gauge how they were progressing against other teams. The course schedule includes eleven team project practicums out of twenty-six class sessions. At each of these practicums, the instructor is available to the teams for guidance and feedback. Midterm peer reviews were administered to the students so any issues with team dynamics could be addressed before software development began.

5. PROJECT DESCRIPTIONS

The pedagogical methodology described in this paper was first applied to an organ transplant matching simulation application for Massachusetts General Hospital in 2009 and has been used in a variety of subsequent web capstone projects. The next section gives a description of the general minimum team project specifications that were offered to teams working on the organ transplant simulator followed by brief descriptions of some of the other web capstone projects teams implemented.

5.1 Organ Transplant Matching Simulator

The organ transplant process in the United States usually involves communications between hospital doctors and organ procurement organization (OPO) doctors. Attempts to match organs with patients are done initially at the local level, then at the regional level, and only if there is still a lack of good matches, are patients matched at the national level. Most of the communications between hospital and OPO doctors currently take place by telephone and are thus time consuming. During the entire process of organ matching, the organs deteriorate and a fraction of them expire before they can be transplanted to a patient. Although a nationwide organ repository with faster online communications may be more efficient than the current process, it is difficult to predict how doctors may game the new system to improve the odds that their patients receive organs.

The student teams' applications expanded upon Massachusetts General Hospital's earlier attempt to simulate a new liver matching system, in order to discover how doctors would use it. The CS460 liver matching web application consists of the following:

1. Participants who take on the role of either an organ procurement doctor who has available livers, or a doctor who has a patient requiring a liver
2. A repository of both available livers and patients who need them
3. A display of possible matches filtered by blood type
4. A list of liver locations and patient locations
5. Display of the remaining time before the liver expires
6. Instant messaging, email and discussion forums for online communications between players
7. Record of organ matches
8. A scoring system for successful matches, expired organs, and deceased patients
9. History of simulation participants' match scores

As with all projects, teams had the opportunity to go beyond the specifications if time permitted. As an example, one team implemented a feature that incorporated the speed of the helicopter used to transport organs between medical facilities and correlated that with the organ's expiration time, to create a dynamic circle on a Google map that would identify potential remaining organ transplant patients within a viable range.

An extension of this liver project was one that involved kidney transplant matches. Kidneys differ from other organ transplants in that patients can be put on dialysis after their kidneys fail but before they receive a replacement kidney. This simulation application also had to include the limited availability of kidney dialysis machines and the length of time that the patient was on dialysis.

5.2 Pax Populi: A System to Coordinate Online English Lessons for Afghani Citizens

The prototype application was developed to assist the non-profit organization, Pax Populi, with coordinating online English lessons for Afghani citizens given by individuals in other countries, particularly the United States.

5.3 Boston Freedom Trail Guide

This mobile web application was developed for the general public to guide visitors along the Boston Freedom Trail. The application provides information on each stop of the Freedom Trail, directions to the next location, and nearby points of interest such as restaurants.

5.4 Medical Resource Locator System

The prototype web application developed for Massachusetts General Hospital both locates clinicians and equipment in a medical setting and sends alerts to all clinicians and staff responsible for medical equipment to proceed to their required destinations such as an emergency room, laboratory or patient care room.

6. EVALUATIONS OF THE PROJECT EXPERIENCE

By every measure, the approach used in CS460 is successful. Instructor evaluation of a project was based primarily on the user interface, system functionality, software quality and class presentations. All of the organ transplant matching applications met the base project requirements with varying degrees of success. Of the eight organ transplant matching teams, there were five applications that exceeded expectations and only one team that barely met the requirements. Furthermore, every web capstone project between January 2009 and May 2014 was completed successfully except for one. The failed project resulted

from the team's selection of a project manager who had created the team's project idea but was then unavailable for a substantial portion of the semester.

At Bentley University, anonymous student evaluations are mandated for every class. These consist of two parts: a formal numerical form that students use to evaluate various aspects of the class and an informal essay form that allows students to directly address concerns and suggestions that is distributed to the instructor after grades have been issued. The CS460 class is consistently ranked highly in the department and university with an average score of 5.2 out of 6 points. Students have also indicated a high degree of satisfaction with the course after graduation through personal communications.

The following representative student quotes came from emails sent to the Bentley University Career Services Office in response to their curriculum satisfaction survey during the summers following the classes:

"I really enjoyed taking CS460 - 'Software Project Management'. The course took very hands on approach to project management and was the primary reason I was able to get an internship."

"This course really added to my job search and gave me some real life experience within software development."

In addition, the following emails were sent directly to the instructor:

"Everyone I have spoken to really got a lot out of this semester. I really enjoyed your class, and I took a lot away from it. It was not easy but I think it did manage to give a pretty accurate snapshot of what the work and responsibility that goes into a software project. I hope you will consider me as a resource in the future for the CIS department and for help to place students in jobs at [a large company]. I am really grateful for the opportunities the CIS program has given me and I would like to be able to give back."

"I am the only official project manager at the company and have been able to apply many of the concepts that you taught in your class. This has allowed me to introduce formal project management practices to our customers and this has been received very well by management. As the company has been growing and more customers are requesting formal project management we have come to the point of hiring additional project managers that would work for me."

".... We have a good number of spots open for Business Analyst interns in IS... If you have students who you think might be interested I would be excited to talk to them. I know anyone who can survive and succeed in your project management class can survive and succeed anywhere."

7. CONCLUSION

A pedagogical methodology has been presented to facilitate the delivery of information systems capstone courses with larger-sized project teams. The three-pronged approach addresses issues arising from larger team sizes regarding team management, team communications, and team member utilization. The instructor, using the Blanchard Situational Leadership II model (Blanchard & Hersey, 2008; Blanchard, et al., 1985), adjusts the leadership style with the teams throughout the semester to guide the teams from a state of low competence to one of high autonomy. A team structure was presented that reduces unnecessary communications within a team and thereby streamlines their operations. Finally, three software development methodologies are presented to reduce the low utilization of the quality assurance manager – V-shaped SDLC, incremental SDLC, and SCRUM Agile

methodology – and make completion of the project within a single semester feasible.

From our experiences of eight semesters of project management courses with web capstones and student feedback, we have concluded that the pedagogical methodology that we use is successful in having student teams apply project management concepts and software development methodologies in the creation of a real-life web application.

8. REFERENCES

Blanchard, K., & Hersey, P. (2008). Situational Leadership. *Leadership Excellence, 25*(5), 19.

Blanchard, K., Zigarmi, P., & Zigarmi, D. (1985). *Leadership and the One Minute Manager*. New York, New York: William Morrow and Company, Inc.

Brandon, D., Pruett, J., & Wade, J. (2002). Experiences in Developing and Implementing a Capstone Course in Information Technology Management. *Journal of Information Technology Education, 1*, 91-102.

Eckerdal, A., McCartney, R., Mostrom, J. E., Ratcliffe, M., & Zander, C. (2006). *Can graduating students design software systems?* Paper presented at the 37th SIGCSE Technical Symposium on Computer Science Education, Houston, Texas.

Futrell, R. T., Shafer, D. F., & Shafer, L. I. (2002). *Quality Software Project Management*. Upper Saddle River, NJ: Prentice Hall PTR.

Gackowski, Z. J. (2003). Case/Real-Life Problem-Based Learning with Information System Projects. *Journal of Information Technology Education, 2*, 357-365.

Grant, D. M., Malloy, A. D., Murphy, M. C., Foreman, J., & Robinson, R. A. (2010). Real World Project: Integrating the Classroom, External Business Partnerships and Professional Organizations. *Journal of Information Technology Education, 9*, 30.

Holifield, J., Longenecker, B., & Feinstein, D. (2012). *Developing Enterprise Information Systems: Experiences of a Graduate Class.* Paper presented at the 2012 Proceedings of the Information Systems Educators Conference, New Orleans, Louisiana.

Keogh, K., Sterling, L., & Venables, A. (2007). A Scalable and Portable Structure for Conducting Successful Year-long Undergraduate Software Team Projects. *Journal of Information Technology Education, 6*, 515-540.

Leidig, P. M., & Lange, D. K. (2012). *Lessons Learned From A Decade Of Using Community-Based Non-Profit Organizations In Information Systems Capstone Projects.* Paper presented at the 2012 Proceedings of the Information Systems Educators Conference, New Orleans, Louisiana.

Loftus, C., Thomas, L., & Zander, C. (2011). *Can Graduating Students Design: Revisited.* Paper presented at the 42nd SIGCSE Technical Symposium on Computer Science Education, Dallas, TX.

Lynch, K., Heinze, A., & Scott, E. (2007). Information Technology Team Projects in Higher Education: An International Viewpoint. *Journal of Information Technology Education, 6*, 181-198.

Magboo, M. S. A., & Magboo, V. P. C. (2003). Assignment of Real-World Projects: An Economical Method of Building Applications for a University and an Effective Way to Enhance Education of the Students. *Journal of Information Technology Education, 2*, 29-39.

Mew, L. (2014). *Improving an Information Systems Capstone/Consulting Course for Non-Traditional Undergraduate Students.* Paper presented at the 2014 Proceedings of the Information Systems Educators Conference, Baltimore, Maryland.

Pollacia, L., Ding, Y. Z., & Yang, S. (2014). *Why Phishing Works: Project for an Information Security Capstone Course.* Paper presented at the 2014 Proceedings of the Information Systems Educators Conference, Baltimore, Maryland.

Reinicke, B., Janicki, T., & Gebauer, J. (2012). *Implementing an Integrated Curriculum with an Iterative Process to Support a Capstone Course in Information Systems.* Paper presented at the 2012 Proceedings of the Information Systems Educators Conference, New Orleans, Louisiana.

Russell, J., & Russell, B. (2014). *A Systems Analysis and Design Case Study for a Business Modeling Learning Experience for a Capstone CIS/IS Systems Development Class.* Paper presented at the 2014 Proceedings of the Information Systems Educators Conference, Baltimore, Maryland.

Silverthorne, C., & Wang, T.-H. (2001). Situational Leadership Style as a Predictor of Success and Productivity Among Taiwanese Business Organizations. *The Journal of Psychology, 135*(4), 399-412.

Tappert, C. C., Cotoranu, A., & Monaco, J. V. (2015). *A Real-World-Projects Capstone Course in Computing: A 15-year Experience*. Paper presented at the 2015 Proceedings of the EDSIG Conference.

Tatnall, A., & Reyes, G. (2005). Teaching IT Project Management to Postgraduate Business Students: A Practical Approach. *Journal of Information Technology Education, 4*, 153-166.

Vyver, G. V. D., & Lane, M. (2003). Using a Team-Based Approach in an IS Course: An Empirical Study. *Journal of Information Technology Education, 2*, 393-406.