Building Real World Domain-Specific Social Network Websites as a Capstone Project

Kwok-Bun Yue
Department of Computer Information Systems
University of Houston-Clear Lake
Houston, Texas, USA
yue@uhcl.edu

Dilhar De Silva
AtLink Communications
1331 Gemini Ave. Suite 300,
Houston, TX 77058
dilhar.desilva@atlinkcom.com

Dan Kim
Mirac Aktepe
Stewart Nagle
Chris Boerger
Anubha Jain
Sunny Verma
Department of Computer Information Systems
University of Houston-Clear Lake
Houston, TX

ABSTRACT

This paper describes our experience of using Content Management Software (CMS), specifically Joomla, to build a real world domain-specific social network site (SNS) as a capstone project for graduate information systems and computer science students. As Web 2.0 technologies become increasingly important in driving business application development, information systems programs will benefit from utilizing these leading edge technologies in realistic and exciting team projects. Two capstone teams built a SNS for a swimming team by configuring, extending and creating Joomla components in a two semester sequence. Appropriate social networking features were integrated with domain-specific application requirements to create an online community to support swimmers in achieving their goals. The prototype received very good reviews from the swimming coaches and the mentor. A software process based on a subset of Rational Unified Process (RUP), agile software development, and scrum was used. The paper discusses the relative merits and suitability of building a domain-specific SNS as a capstone project. It also presents the points of view and experiences of the project's industrial mentor and students.

Keywords: Web 2.0, Content Management Software, Social Networking Sites, Capstone Project, Information Systems

1. INTRODUCTION

Real world domain-specific applications have frequently been used as capstone projects in many universities (Beasley 2003; Conn 2004; Hadfield and Jensen 2007; Leidig et al. 2006; Myers 2003; Reichlmay 2006). In these projects, students frequently worked in teams to solve problems for non-profit organizations (Leidig et al. 2006) or commercial corporations (Myers 2003). These capstone projects allowed students to integrate their knowledge in computing and information technology to encompass the complete analysis, design and implementation phases of software development.

Perceived advantages of real-world capstone projects include the following:

- Provide the necessary bridge between academic study and the professional software development world (Hadfield and Jensen 2007).
- Promote the development of leadership, communications, collaboration and organizational skills that are essential in successful software development (Leidig et al. 2006; Hadfield and Jensen 2007).
- Provide a culminating and integrative educational experience that unifies diverse technologies and theories learned in the information systems or computing curricula (Clear et al. 2001).
- Foster the understanding of organizational processes for which the software is developed (Conn 2004).

Because of these perceived benefits, many universities have capstone courses that are project-oriented (Clear et al. 2001). Similarly, IS 2002, the latest undergraduate model IS curriculum developed by AIS, ACM and AITP (Gorgone et al, 2002), recommended a team-oriented project course as a core course: IS 2002.10 Project Management and Practice. More recently, in the update of the latest graduate IS model curriculum from MSIS 2000 to MSIS 2006 (Gorgone et al. 2006), an important change was the creation of the required core course, MSIS 2006.7 Integrated Capstone.

Depending on the course objectives, capstone projects may emphasize different aspects of software development. Many software engineering capstone projects demand that students adhere to formal processes and models, such as the Capability Maturity Model (CMM) (Hadfield and Jensen 2007), or an industrial flavored software engineering process (Conn 2004). They may span the entire software lifecycle (Beasley 2003), or focus on a specific application area such as database (Tuttle 2000). Educators have also devised different strategies to enhance student learning in capstone projects (Kumar 2006).

In summary, team-oriented capstone projects are highly beneficial and rich in diversity. Researchers and educators have created a wealth of literature on varying approaches and experiences conducting effective capstone projects. The goal of this paper is to enrich this collective knowledge by describing our experience conducting information systems capstone projects in an important and exciting application area that has not yet been explored. It is based on the work of two teams building a real-world domain-specific Social Network Site (SNS) in two sequential semesters.

The remainder of this paper is organized in the following manner. Section 2 provides background information about domain-specific SNSs. The organization and process of the capstone project course at the University of Houston-Clear Lake is detailed in Section 3, along with a discussion on metrics for measuring the effectiveness of capstone projects. Section 4 provides an overview of the domain-specific as well as general social network requirements for building a website serving a swimming team. Furthermore, it highlights why a domain-specific SNS is the most effective approach in cultivating a supportive community for the swimmers. The mentor of the project was the Chief Technology Officer (CTO) of a high tech company who was also a board member of the swimming team. Section 5 discusses our experience in using a Content Management System (CMS), specifically Joomla, to design and implement the project. Section 6 describes software methodology and project management. Section 7 discusses the effectiveness of the capstone project based on the metrics delineated in Section 3. Section 8 presents the views of the mentor and the students in order to develop a comprehensive perspective of the project. Finally, we draw our conclusions in Section 9.

2. DOMAIN-SPECIFIC SOCIAL NETWORK SITES

The well publicized popularity and success of sites such as MySpace, Facebook, and YouTube firmly established SNSs in both public and academic forums one of the most important phenomena of the Web 2.0 era. Companies rushed to utilize social networking features to encourage community-based collaborative creation of rich content for the benefit of their websites. In March 2008, according to the web traffic tracking site, Alexa.com (2008), five of the top ten global websites were general SNSs: YouTube (#2), MySpace (#5), Facebook (#6), Hi5 (#8) and Orkut (#10). Four other top sites had significant social network features: Yahoo! (#1), Google (#3), MSN (#7) and Wikipedia (#9). Windows Live was #4.

Boyd and Ellison (2007) defined SNSs as "web-based services that allow individuals to (1) construct a public or semi-public profile within a bounded system, (2) articulate a list of other users with whom they share a connection, and (3) view and traverse their list of connections and those made by others within the systems."

In essence, SNSs seek to connect their users through various services. Many traditional websites focus on interests to build a community. For example, ESPN is a site based on interests in sports and although it is not a SNS, it has some community features. On the other hand, SNSs focus on people (Boyd and Ellison 2007). General purpose SNSs, such as MySpace and Facebook, are basically open to anyone. They provide a wide range of general social network features to serve a broad demographic. Some popular features include: profile management; friend management; video, audio and photo sharing; and blogging. There are also many SNSs that target specific demographics. These sites provide niche features not found in general SNSs. For examples, care2.com (2008), a green living and social activism SNS, has a feature to allow members to create or sign petitions. Gaia Online (2008), an anime and game SNS, provides Gaia Golden Marketplace that allows members to set up virtual shops. It also provides auction features to sell their comic related items. Although the scopes of these niche SNSs are generally smaller than general SNSs, they can still be broad. Both care2.com and Gaia Online have millions of registered users. The cost of creating and maintaining these sites can be prohibitive for smaller organizations.

The recent advances in SNS technology make the development of smaller scale SNSs with limited resources a feasible alternative. One option is to use web-based SNS creation sites, such as Ning.com (2008), which provide tools to either configure and host basic sites quickly, or customize more sophisticated sites by programming with their Application Programming Interfaces (API). For added flexibility and control, developers can use Content Management Systems (CMS) to develop their SNSs in a cost effective manner. Many CMS provide social network features that only require configuration rather than programming. The arrival of mature open source CMS such

as Joomla and Drupal makes the entry threshold even more accessible. As a result, small organizations can now afford to develop domain-specific SNSs targeting small communities with very specific interests and social connections. For example, the target domain of the SNS may be a church, a local swimming team, or a high school's alumni association. Because domain-specific SNSs have more focused users, objectives, and common interests, their requirements can differ significantly from general SNSs. These differences include the following:

1. The active community may be significantly smaller. Using a swimming team as an example, the swimming community may have only a few hundred swimmers.
2. The visibility and access structures may be more hierarchical and restrictive. For example, a swimming team may have many types of users such as swimmers, friends, coaches and parents.
3. There may be tighter integration of back end enterprise data and service to create contents catering to the domain-specific applications. For example, a swimming team may want to integrate its swimmer training database into its site.
4. Different design and composition of social network features may be chosen to satisfy the domain requirements.
5. The site may need to support offline activities and connections. There may be more users with offline connections, known as "latent ties" (Haythornthwaite 2005). For example, swimmers may carpool to competition meets.

Compared to general SNSs, the different requirements of domain-specific SNSs provide a new set of developmental challenges.

3. CAPSTONE PROJECT ORGANIZATION

The Masters Degrees in Computer Information Systems and Computer Science at the University of Houston-Clear Lake provide an extended coursework option or a thesis option. In the extended coursework option, students must complete 24 graduate hours to ensure maturity before taking the required capstone project course. During the capstone project course, they work on real-world projects mentored by the surrounding high technology companies.

Before the start of the semester, mentors send the initial requirements to the instructor of the course. Projects are selected, refined, and posted in the course's website. Students form teams and select projects during the first class. They are expected to fulfill two sets of requirements: course requirements and project requirements. The course requirements are specified by the instructor and include technical presentations, technical reports, project site maintenance, meeting minutes, deliverable documentation, prototype demonstration, etc. The project requirements are specified by the mentors with input from the instructor. They are project specific and usually encompass a significant portion of the software life cycle.

In a typical project, students conduct frequent mentor meetings to refine requirements; develop design solutions; discuss technical issues; ensure proper documentation; and demonstrate prototypes. Students are entirely responsible for every aspect of project management. To ensure a successful start, the instructor attends at least the first two mentor meetings and provides feedback to the teams immediately after the meetings. The course does not designate a specific software development methodology. Instead, teams can use a methodology deemed most appropriate to their projects. Mentors sometimes have preferred methodologies. If not, students can select a reasonable methodology with the consensus of the mentor and the instructor. Once a methodology is selected, it must be applied throughout the whole project consistently.

Projects can generally be divided into two types. Many of them are development projects concerned with extending the functionality of a mentoring company's software products. The deliverables are usually directly integrated into the products in some manner. Other projects involve proof of concept prototypes where companies want to try out new ideas not immediately compatible with the underlying platform of their products. There are also cases where the end products are not used by companies but by organizations where the mentors served. In nearly all situations, the deliverables are expected actually to be used.

Mentors are usually high ranking technical personnel. For example, there were two chief technology officers and two senior software managers mentoring capstone projects in the spring semester of 2008. Mentors may also be non-technical managers. In the same semester, one project was mentored by a company president without formal technical background. Teams working on projects with non-technical mentors are expected to refine their 'soft skill' even more.

Our experience indicated that the proper selection of projects was crucial to the success of the course. Sun and Decker (2004) identified seven criteria for ideal capstone projects. Some of these criteria, such as students working in teams and the provision of a research option, were automatically satisfied by the program and course design of our university. Other criteria useful in evaluating our projects included the following:

- Challenge student proficiency in IS and CS subject areas
- Require sound software engineering methodology
- Produce high quality real-world products that are actually used
- Motivate students through important and relevant client-oriented projects

We added two criteria on project selection:

- Enhance student proficiency in real-world team software development
- Excite students and prepare them for the job market with leading edge technologies and emerging application areas

The course tried to select projects with diverse characteristics that satisfied these criteria well. For example, in the last two years, there were leading edge projects on workflow applications, communications, data integration, mobile applications, graphics development, simulation, etc., using a wide range of technologies such as AJAX, VoIP, SMS, Rich Internet Applications, XML, web services, Mashup API, Google Android, etc.

In the fall semester of 2007, the capstone project course for the first time included a domain-specific SNS project to develop a site for a local swimming team of which the mentor was a board member. The end product was a working

prototype that was well received by the mentor and the coaches of the swimming team. A second capstone team worked on refining and improving the domain-specific features in the spring semester of 2008.

4. A DOMAIN-SPECIFIC SNS PROJECT

The Space Center Aquatic Team (SCAT) provided residents of Houston with swimming programs for all age groups. It managed a dated site that was constructed using a potpourri of technologies including HTML, PHP and Perl. Figure 1 shows a partial screenshot as an example. The site was maintained by a volunteer webmaster with input from the swimming coaches and was updated irregularly. As a result, content was added inconsistently and the navigational structures became increasingly ineffective with many broken and redundant links. As the organizational memory and expertise of the separate components gradually deteriorated, the maintenance of the site became increasingly difficult. Furthermore, infrequent updates by a very small group of personnel varying in skills and dedication resulted in minimal content that was neither exciting nor personal.

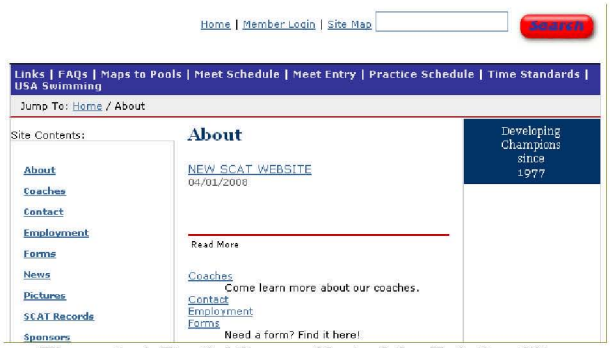

Figure 1. A Partial Screen Shot of the Existing Site

In the summer of 2007, a capstone project mentor approached the instructor to develop a new prototype website based on social network technology to replace the existing site. It would use the SNS technologies to create a thriving online community to further the goals of the swimming team. As a comparison, the existing site provided only static and sometimes stale information not intended for building a community.

Swimming teams are very goal driven to improve performance and to participate in open swim meets. Swimmers train hard for long hours on a regular basis. The advantage of joining a swimming team includes the help of qualified coaches. Equally importantly, instead of training alone, a swimming team provides a physical community for mutual support that is crucial for long term commitment and success. The new site would extend the physical community to a much broader online community that can provide even better support for swimmers to achieve their goals. For example, through the new site, a caring grandfather living in India can provide much support to his granddaughter swimming in Houston by becoming an active partner in her quest.

Some example use cases of social networking for building such a supportive community in the site include the following:

- Swimmers can announce their goals, such as the number of training sessions per week, time improvement, meet participation, etc., in very specific ways on their personal web spaces. It is known that simply announcing goals to the world can be a powerful motivator.
- Swimmers can post videos, audios, photos, blogs, comments, etc. while achieving their goals. For example, swimmers can discuss swimming techniques among themselves; meet information and preparation; training strategies, etc. A video of a quality competition win can serve as a powerful incentive for other swimmers.
- Friends and families can visit the personal spaces of the swimmers to provide encouragement, comments, information, and suggestions. They can become partners with the swimmers. A word of encouragement from someone special can go a long way to motivate a swimmer. Future versions of the site may allow more than just cheerleading but more substantial involvement such as donations to swimmers to buy equipment.
- Parents can visit the personal web spaces of the swimmers to play the role of cheerleader. They can also get help with routine activities, such as paying monthly fees, arranging for carpools, etc. They can view the progress of their children and discuss it with the coaches.
- Besides providing coaching information to the swimmers through their personal web spaces, coaches can also perform many administrative tasks and broad channel communications, such as announcements, setting up master training templates, etc.

Many of these community building features could be supported by general SNS technologies, such as user profiles, friend management, video and audio sharing, blogs, discussion forum, photo gallery, etc. There were also many domain-specific features, such as the creation and management of swimming goals, training sessions and meets, interaction between coaches, swimmers and others, etc. These features must be custom-built.

After discussing preliminary requirements, both the mentor and the instructor saw the advantages of such a domain-specific SNS project for the students. These include:

- Working on a trendy and exciting application area.
- Mastering leading-edge industrial strength technologies.
- Gaining experience in web rapid prototyping.
- Integrating technologies to satisfy real-world business needs.

The project was quickly formulated with major requirements identified. Besides the community building features, the new site also needed to capture essential information in the existing site in a structure that would be easy to update and navigate. This essential information included schedules for practices and public competitions, registration and entry forms, and competition meet results. All public meet results could be obtained from USA Swimming (2008), the national governing body. All competition swimmers were members of USA Swimming and their individual profiles and meet results could be obtained through their accounts there. Integrating these meet

results with local practice results for analysis and display would be very important in helping swimmers to set goals and monitor their progresses. It could also be an important motivational tool. However, the existing site inconsistently published partial competition results in PDF format. It nevertheless provided some scattered information to assist and motivate swimmers. This included maps to the swimming pools, links to external swimmer resources, time standards for different levels of competitions, a swimmer of the month program, etc.

These requirements highlight the differences between general SNSs and domain-specific SNSs. Many general SNSs have only one type of generic users. On the other hand, in the planned swimming team site, four user types were identified:

1. Swimmers: They must be registered with the swimming team and usually have unique USA Swimming IDs. Swimmers can set goals; blog; post photos, videos and audio; invite friends; control visibility of the site contents, etc.
2. Parents: They must be parents of one or more swimmers and have special privileges on the swimmer's account, such as paying monthly fees.
3. Coaches and board members: They have special privileges on the entire site, including posting news and announcements, setting goals, etc.
4. Friends and family: They are general users who can view and participate contingent upon swimmer's approval.

As another example of domain-specific features, a coach responsible for a certain level of swimmers usually sets the training goals for the day for these swimmers, filling in such details as planned number of laps. After the training, the swimmer supplies individual performance data, such as time taken, performance condition, etc. The new SNS would need to integrate these kinds of domain specific requirements.

5. DESIGN AND IMPLEMENTATION IN JOOMLA

Early on, it was decided that Joomla (2008a) would be used as the Content Management Systems (CMS) for this project. Joomla is a popular open source CMS with a reported community of 150,000 users and developers. It is designed as a user-friendly system that non-technical professionals can install and configure for straightforward applications. It has a simple and expandable template and component system to allow easy display customization and function enhancement. Developers can also expand core Joomla functions by developing extensions. The community is growing very rapidly and Joomla (2008b) claims to have 3,469 extensions. This represents an increase of 1,180 extensions or 51.3% in eight months since the authors last checked in November 2007. There are other popular open source CMS, such as Drupal (2008), with their own strengths. However, Joomla was selected because of its relative ease of use, abundance of extensions, and availability of expertise in the mentor's work environment. A significant portion of the remaining section applies to mature open source CMS in general, not just to Joomla.

Software development in Joomla can be quite different than traditional software development. Web design of Joomla's sites includes templates, CSS, and uses of the Joomla's display architecture. For functional requirements, it involves Joomla's components. If a functional requirement is completely satisfied by a Joomla component, only configuration is necessary. No programming is needed. The issue becomes trickier when no component has been identified to satisfy the requirements completely. There are several solutions, each with its relative merits.

(1) Avoid development: adjust the requirements to fit the functionality provided by the most optimal component. Changing requirements on the fly may seem outrageous to some. However, we are now in the era of Web 2.0 with 'perpetual beta' in which "the product is developed in the open, with new features slipstreamed in on a monthly, weekly, or even daily basis" (O'Reilly 2005). It is more important to deploy a function quickly to gather community feedback for refinement than to carefully plan and implement every detailed requirement that the users may or may not want. This software development model also works well with the "bazaar model" of open, rapid, and concurrent development as opposed to the "cathedral model" of close, planned, and hierarchical development described in the book "The Cathedral and the Bazaar" by Eric Raymond (2001). With possibly the least amount of effort and no programming, this option may be a suitable choice for many circumstances.

(2) Development through the Joomla community: request the Joomla community to develop new components or modify existing components to provide the needed functionality. Many open source developers, Joomla included, are responsive to community needs. However, even if the request is accepted, there is no control of when it will become available. This problem is especially serious for domain-specific SNSs in which many requests may be too specific and not as useful to general Joomla users, thus drawing less attention from the developers.

(3) Home-grown development: extend existing components or create new components to implement the needed functions. The open source nature of Joomla easily allows creation and modification of its components. If a mature component with compatible features and software architecture is available, software reuse can be maximized and reliability enhanced. The amount of programming effort may also depend on the specific need of the current project's stage, with quick and dirty implementation in earlier stages. For domain-specific SNSs, this may involve the custom coding of a general purpose component to satisfy a particular vertical need. Like other software extensions, this option creates a code branching problem in which the extended code may need to be ported when a newer version of the selected component is installed.

Alternatively, the team can develop an entirely new component to satisfy the needed functionality. This option provides the most flexibility. There is no code branching problem. On the other hand, development and testing are more intensive.

In the first semester, a capstone team was charged to build a prototype SNS rapidly for a proof of concept implementation. As a result, design and implementation were focused on identifying the most compatible component for a given requirement. If the selected component did not fully support the requirement, the unfulfilled parts were simply

left unimplemented, with notes recorded for future remedial actions. In some cases, various options in extending existing components or creating new components were studied but not implemented. The team had not coded. At the end of the semester, the team successfully built a SNS integrated with many features including news, announcements, profile and friend management, registration, advertising, video sharing, photo sharing, blogging, forums and commenting systems, etc, The site was very well received by the mentor and the coaches, thus prompting further development in the next semester.

The following example provides a glimpse of the team's effort. To support photo sharing and gallery functions, there were 105 related Joomla extension components for this purpose at the time. The team studied these extensions online and selected to download, install and test 12 of them. Eventually, the team decided to select one of the open source components, Expose4, because it satisfied all photo gallery requirements; was easy to install and administer; and provided a visually pleasing way of deploying photographs.

In the second semester, a new capstone team was charged to add several functions to the site. Because of the domain-specific nature, no existing components were found to satisfy some of these requirements and the team extended existing components and created new components.

An example of code extension is the registration system. As discussed earlier, there were four major user types: swimmers, parents, coaches and board members, and family and friends. However, the built-in Joomla community builder supported a registration system with only a single user type. As a result, the site built by the first team forced every user to fill in the same registration form, even though some fields were not appropriate. For example, there was a field for parents to specify the name of their child. This field was also visible to all other users thus causing confusion. A new registration system was needed to prompt the user for his intended user type before directing him to the right form.

A team member extended the basic community builder component of Joomla for this purpose. A user type field was added to the user data to indicate the type. Joomla's master registration table was modified to store the superset of all data fields of all user types. The referrer program was rewritten to redirect the user to the right registration form in which inapplicable registration fields were made invisible. This quick and dirty solution was thus a fragile one that required proper recoding of the referrer and database population as well as appropriate configuration to set the user type field.

An example of creation of extension components is the implementation of a training log. Joomla provides a convenient architecture and programming guidelines for developing extension components. The team captured the data in the existing paper training log into a web form and developed the code for data validation and population of the underlining MySQL database. It also developed the SQL code to create the initial database. Joomla extension components must include the development of administrative units. The team developed an administration unit to allow the coach to set up target training goals. The swimmers could then enter the training results, or they could personalize their training goals.

Besides component development, the second team also spent significant effort on identifying, installing, and testing extensions. Interestingly, two requirements were eventually satisfied by commercial Joomla extensions and not open source extensions. This is a popular way to 'monetize' Joomla development. The first requirement was the development of a multiple vendor (e.g. Paypal, Google Outlook, etc) electronics payment system with some built-in inventory and account functionality. The other was a chat server that utilized Joomla user authentication for restriction control. In both cases, many open source extensions were explored and rejected before selecting the commercial extensions.

Eventually, the second team produced a refined, rich, and favorably reviewed domain-specific SNS. Figure 2 shows a portion of the main page of the new site.

Figure 2. A Partial Screen Shot of the New Site

6. SOFTWARE PROCESS

The capstone project course does not require students to adopt the same software engineering process. However, they are expected to use a software process suitable for the nature of the project and agreeable to the mentors. In most cases, modeling using use case diagrams and class diagrams of the Unified Modeling Language (UML) were employed. Most teams took a flexible approach and selected a less formal process that still satisfied the mentors, such as a subset of the Rational Unified Process (RUP) (Kruchten 2003) or agile programming (Martin 2003).

Much research has been done on engineering web applications. Newer approaches, such as those based on Model Driven Architecture (Fujikawa and Matsjtsjika 2004) or agile programming (Souza and Falbo 2005), have been proposed for developing newer generations of web applications. In this project, we took advantage of the rich industrial experience of the mentor, who happened to be a member of the Core UML team of OMG (Object Management Group), the standard body that defined UML. Both capstone teams followed the mentor's recommended approach. The approach had flavors from RUP and agile software development. There were two major focuses. First, in the initial six to seven weeks, both teams defined and refined the requirement and design models using use case diagrams, class diagrams, and sequence diagrams in consultation with the mentor, the head coach and the instructor. Second, they used scrum lists to manage the software project (Schwaber 2004; Wikipedia 2008). Scrum is a technique in agile software development that focuses on concurrent tasks from overlapping development phases, as opposed to more traditional models that deal with phases

sequentially. We found that scrums and agile software development fit the nature of domain-specific SNS very well, especially when a CMS like Joomla was used. They are also suitable for the short 14 week time frame in which the capstone projects must be completed.

7. PROJECT EFFECTIVENESS

We did not conduct an empirical analysis to measure scientifically the project effectiveness. After all, there were a total of only nine students involved with the two capstone teams, which was too small for valid statistical analysis. On the other hand, the feedback from all parties, including the students, instructor, mentor and users, was all very positive. At the end of the semester, all students and mentors assessed course learning outcomes. The students were expected to acquire the abilities indicated by these outcomes upon course completion:

1. Develop real-world computer and information systems projects
2. Analyze real-world problems to devise requirement specifications
3. Construct effective software solutions for real-world problems
4. Collaborate professionally with team members, customers, mentors and supervisors
5. Manage and participate in software projects
6. Present technical presentations effectively
7. Write technical reports effectively

On a scale of 1 to 5, 5 being excellent, mentors and students assessed these outcome satisfactions as either 4 or 5, with the majority being 5. Student evaluations of the course and the instructor produced similar ratings. In the remainder of this section, we briefly analyze the effectiveness of a domain-specific SNS as a capstone project using the criteria described in Section 3.

7.1 Proficiency in CIS and CS Subject Areas

The project required good requirement analysis, business analysis, project management, UML modeling, customer interaction, and web design and development. All of these are important IS and CS subject areas. The first team was not involved in coding and testing. However, the team did research numerous extension components for customization. The second team developed and extended components with intensive coding. Extending existing code was perceived to be a very useful exercise since most homework assignments in other courses require students to develop new code instead. Only one student had prior Joomla administration experience but both teams had no trouble in learning Joomla quickly enough to complete all requirements.

7.2 Sound Software Engineering Methodology

Although the teams had not applied a formal software process, the informal approach based on agile software development and a subset of RUP was found to be effective.

7.3 Production of High Quality Real-World Products

The head coach and the mentor participated in several sessions of requirement analysis and site demonstrations. Although it was still a prototype, the site received excellent reviews. The mentor and the instructor emphasized code portability in extending and developing Joomla's components so that the site could be ported easily.

7.4 Important and Relevant Client-Oriented Projects

The target swimming team had about 200 swimmers. The new SNS was targeted to support a community of about 1,000 users, most of them family members and friends of the swimmers. This is small in comparison with general SNSs. The SNS satisfied the original requirements and provided many new functions for community cultivation.

7.5 Enhancing Student Proficiency in Real-World Team Software Development

The students found that the nature of the project, the technologies and the software processes have significantly enhanced their real-world experience. The rich experience of the mentor also helped immensely.

7.6 Leading Edge Technologies and Emerging Application areas

Joomla, scrum and UML are industrial leading edge technologies. While SNSs have become ubiquitous, domain-specific SNSs now appear to be at the forefront of the next wave.

Our experience also indicated that domain-specific SNSs have other advantages. Compared to other projects, they provide a relatively broad spectrum of software development activities from simple configuration to component development. This mimics many real world IT projects. Furthermore, the component nature of Joomla made it easy to break down tasks and assign them to individual team members.

8. MENTOR'S AND STUDENTS' VIEWS

To provide a full perspective of the project, this section describes the views of the mentor and the students.

8.1 The Mentor's View

As the CTO of a startup company, the mentor was very busy. To mentor the project, he needed to maintain a delicate balance between the sponsoring company, technologies, and student interests. The time limitation of one semester was especially crucial. The mentor indicated that developing a domain-specific SNS using an open source CMS seemed to provide that delicate balance well. The three to four month period appeared to be in line with what a normal industrial group of developers could do. The challenge of the project was to find out "What can we do in 3-4 months to develop a domain-specific SNS from scratch?" The mentor and the end users (coaches) were eventually very satisfied with the end products. The mentor attributed the success to three crucial decisions and arrangements made early in the semester.

(1) Used an open source CMS with an active community.

Building the SNS on a proven infrastructure significantly shortened the development cycle since many general Web 2.0 services would be ready for adaptation. The team could thus focus more on domain-specific features. In many situations, the rich set of existing Joomla components allowed the team simply to reuse rather than develop components. When development was needed to push the

limit, the active community provided strong support. The open source nature also meant that extending components was easier than in closed systems. Finally, picking a CMS that could be installed and configured easily on any commercial hosting company was also a deciding factor. Instead of installing the web server, database and application server separately in a messy manner, the installation of Joomla was easy and straightforward, allowing the team more development time.

(2) Used an appropriate software process.

Utilizing an appropriate software process is crucial for a software project of any size. This was even more pronounced since most capstone project students had never worked in a team environment. An early decision was made to use agile scrum with a mixture of UML diagrams. This turned out to fit the project nature well. It facilitated communications among team members and the mentor. At any time, the students knew from the scrum list what tasks need to be accomplished and the mentor knew the accurate overall picture and was able to provide advice and help on crucial tasks in a timely manner.

(3) Provided easy access to the domain-specific content and the community.

A domain specific SNS is only as good as the content and the support/features supporting the natural interactions of the community. The mentor and the instructor decided that access to domain experts and users would be crucial and ensured a quick communication channel early in the project. The use of an agile scrum type of process also facilitated the constant interaction and feedback loop.

8.2 The Students' View

The students indicated that working with a CMS such as Joomla presented many new and exciting opportunities including having a predefined framework, reusing existing components, and developing and extending plug-ins that required minimal technical maintenance. Based on their experience, developing a feature rich and well designed site from scratch could be very time consuming and tedious. The use of Joomla and its framework simplified the first few steps in the web design, thus making possible the building of such a site in the relatively short time frame of a semester.

The students found that mastering Joomla, its framework, and the underlying scripting language, PHP, did not create as much of a challenge as anticipated at the beginning of the semester. On the other hand, the quantity and quality of existing components greatly affected the development effort. They found that time spent in researching existing plug-ins according to requirement metrics was very worthwhile and wished that they might have been even more thorough. Once applicable components were identified, the team could determine the right solution to be employed. This allowed them to set priorities and budget their time more efficiently.

The teams indicated that one of the most exciting aspects of working with Joomla was the possibility of open sourcing the developed plug-in. Projects done in an academic setting are often of little use outside their limited scope. However, a well designed and implemented plug-in created for Joomla would almost certainly find life on other websites.

Overall, the students felt that they gained experience in requirement analysis, CMS, SNS, and a leading edge component development technology. The project provided them with working experience on a relatively complete software lifecycle in a short period of time. They were very satisfied with the end results and felt comfortable showcasing their work to enhance their careers.

9. CONCLUSIONS

This paper presents our experiment of building a domain-specific SNS by two separate capstone project teams during sequential semesters. We found that a domain-specific SNS was attainable for a one semester capstone project. Using a CMS tool like Joomla, rich domain-specific SNSs could be developed quickly. Although only one student had any prior Joomla experience, the teams were able to effectively build a feature rich SNS prototype. Aided by the use of an appropriate software development process, it satisfied many criteria for good capstone project candidates. Thus, our experience indicated that developing domain-specific SNSs could be good candidates for capstone projects.

10. ACKNOWLEDGEMENTS

We would like to thank Christine Paul, Gary Thomson, Chloris Yue, the anonymous reviewers, and the assistant editor for their invaluable input.

11. REFERENCES

Alexa.com (2008), "Global Top 500 sites." Retrieved March 20, 2008 from http://alexa.com/site/ds/top_sites?ts_mode=global.

Beasley, R. E. (2003), "Conducting a successful senior capstone course in computing." Journal of Computer Sciences in Colleges, Vol. 19, No. 1, pp. 122-131.

Boyd, D. M. and Ellison, N. B. (2007), "Social network sites: Definition, history, and scholarship." Journal of Computer-Mediated Communication, Vol. 13, No. 1. Retrieved April 17, 2008 from http://jcmc.indiana.edu/vol13/issue1/boyd.ellison.html.

Care2.com (2008), "Care2.com petition site." Retrieved March 19, 2008 from http://www.thepetitionsite.com/.

Clear, T., Young, F., Goldweber, M., Leidig, P., and Scott, K. (2001), "ITiCSE 2001 Working Group Reports - Resources for Instructors of Capstone Courses in Computing." SIGCSE Bulletin, Vol. 3, No. 4, pp. 93-113.

Conn, R. (2004), "A reusable, academic-strength, metrics-based software engineering process for capstone courses and projects." ACM SIGCSE Bulletin, Vol. 36, No. 1, pp. 492-296.

Drupal (2008), "Drupal home page." Retrieved March 19, 2008 from http://drupal.org/.

Fujikawa, Y. and Matsjtsjika T (2004), "New Web Application Development Tool and its MDA-based support methodology." Fujitsu Sci. Tech. J., Vol. 40, No. 1, pp. 84-101.

Gaia Online (2008), "Gaia online starting page." Retrieved March 20, 2008 from http://www.gaiaonline.com/.

Gorgone J., Davis, B., Valacich, S., Topi, K., Feinstein, D. and Longenecker, H. (2002), "IS 2002 Model Curriculum

and Guidelines for Undergraduate Degree Programs in Information Systems." Retrieved October 7, 2008 from http://www.aisnet.org/Curriculum/IS2002-12-31.doc.

Gorgone J., Gray P., Stohr E., Valacich J. and Wigand R. (2006), "MSIS 2006: model curriculum and guidelines for graduate degree programs in information systems." SIGCSE Bulletin, Vol. 38, No. 2, pp. 121-196.

Hadfield, S. M. and Jensen, N. A. (2007), "Crafting a software engineering capstone project course." Journal of Computing Sciences in Colleges, Vol. 23, No. 1, pp. 190-197.

Haythornthwaite, C. (2005), "Social networks and Internet connectivity effects." Information, Communication, & Society, Vol. 8, No. 2, pp. 125-147.

Joomla (2008a), "Joomla's home page." Retrieved July 29, 2008 from http://www.joomla.org/.

Joomla (2008b), "Joomla's Extensions." Retrieved April 14, 2008 from http://extensions.joomla.org/.

Kruchten, P. (2003), The Rational Unified Process: An Introduction. 3rd Edition, Addison-Wesley Professional, Boston, MA.

Kumar, A. (2006), "Strategies to Enhance Student Learning in a Capstone MIS Course." Issues in Information Sciences and Information Technology, Vol. 6, pp. 328-332.

Leidig, P., Ferguson, R. and Leidig, J. (2006), "The use of community-based non-profit organizations in information systems capstone projects." Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education (ITiSCE '06), Bologna, Italy, June 26-28, pp. 148-152.

Martin R. (2003), Agile Software Development: Principles, Patterns, and Practices/ First Edition, Prentice-Hall, Upper Saddle River, NJ.

Myers, M. (2003), "An IS Capstone Project: The Mywick Property Management System." Journal of Information Systems Education, Vol. 14, No. 3, Fall 2003, pp. 235-239.

Ning.com (2008), Retrieved March 19, 2008 from http://www.ning.com/.

O'Reilly (2005), "What Is Web 2.0? Design Patterns and Business Models for the Next Generation of Software." Retrieved April 15, 2008 from http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html,

Raymond, E. (2001), "The Cathedral & the Bazaar." First Edition, O'Reilly, Sebastopol, CA.

Reichlmay, T. (2006), "Projects and solutions for industry: Collaborating with industry: strategies for an undergraduate software engineering program." Proceedings of the 2006 international workshop on Summit on software engineering education, Shanghai, China, May, pp. 13-16.

Schwaber K. (2004), Agile Project Management with Scrum. Microsoft Publishing, Seattle, WA.

Souza, V & Falbo, R (2005), "An Agile Approach for Web Systems Engineering." Proceedings of the 11th Brazilian Symposium on Multimedia and the web, Pocos de Caldas - Minas Gerais, Brazil, December 5-7, Article No. 3.

Sun, N., & Decker, J. (2004), "Finding an ideal model for our capstone experience." Journal of Computing Science in Colleges, Vol. 20, No. 1, pp. 211-219.

Tuttle S. M. (2000), "A capstone course for a computer information systems major." Proceedings of the thirty-first SIGCSE technical symposium on Computer science education, Austin, USA, March 7-12, pp. 265-269.

USA Swimming (2008), "USA Swimming's Home Page." Retrieved April 14, 2008 from http://www.usaswimming.org/.

Wikipedia (2008), "Scrum (development)." Retrieved April 16, 2008 from http://en.wikipedia.org/wiki/Scrum_(development).

AUTHORS' BIOGRAPHIES

Kwok-Bun Yue (B.S., M.Phil., Chinese University of Hong Kong, M.S., Ph.D., University of North Texas) is a Professor of Computer Information Systems and Computer Science at University of Houston-Clear Lake (UHCL). He is the chairperson of the Division of Computing and Mathematics. His research interests are in Internet computing, semi-structured data, and information systems and computer science education. He had published more than 30 technical papers. Dr. Yue was a recipient of the UHCL Distinguished Teaching Award, the UHCL Piper Award and the UHCL Fellowship Award, and had served as a CTO of a startup company.

Dilhar De Silva (B.S., M.S., UHCL) is the CTO of AtLink Communications, Inc. He was a member of the Core UML team that standardized Unified Modeling Language (UML). He has more than 15 years of industrial experience, including more than 10 years serving in senior managerial positions such as CTO of Atlantis One Technologies, Inc., Vice President of Development of Computer Associates, Inc., Director of Development Platinum Technology, Inc., etc. Mr. De Silva hired hundreds of IT/CS professionals during his industrial career. He received the UHCL Distinguished Alumni Award in 2008 and served as a mentor of many UHCL capstone projects for the last five years.

Dan Kim (Ph.D. SUNY Buffalo) is an Associate Professor of Computer Information Systems at UHCL. His research interests are in multidisciplinary areas such as electronic commerce, mobile commerce, information security and assurance. Recently he has focused on trust in electronic commerce, wireless and mobile commerce, and information security and assurance. His research work has been published or in forthcoming more than 70 papers in refereed journals and conference proceedings including Information Systems Research, Journal of Management Information Systems, Communications of ACM, Decision Support Systems, International Journal of Human-Computer Interaction,

Journal of Organizational and End User Computing, IEEE Transactions on Professional Communication, Electronic Market, IEEE IT Professional, Journal of Global Information Management, and International Journal of Mobile Communications, ICIS, HICSS, AMCIS, INFORMS, ICEC, ICA, etc. Dr. Kim received the best-paper runner-up award at the International Conference on Information Systems (ICIS) 2003 and the best research paper award at Americas Conference on Information Systems (AMCIS) 2005.

Mirac Aktepe (B.S., Istanbul University, M.S., UHCL) is a software engineer currently working for a marketing agency in Houston, Texas. His research interests are in Web application development and UI design.

J. Stewart Nagle III (B.S., M.S., UHCL) is a Houston-based professional software engineer. Mr. Nagle currently is employed with GHG Corp developing extensions for business applications for the company's software product. Prior to this Mr. Nagle developed .NET based Web-based EDI document translation applications for DiCentral Corp.

Chris Boerger (M.S., UHCL) is a senior software engineer for Lockheed Martin Space Operations. He currently works on the Consolidated Planning System for NASA at the Johnson Space Center.

Anubha Jain (B.S., M.S., UHCL) is a software developer focusing on business analysis, .NET and C++ programming, and Internet applications. Her research interest is in the development of Rich Internet Applications.

Sunny Verma (B.S., UPtech University, M.S., UHCL) is a system analyst with EMMES Corporation in Rockville, Maryland, working on the Bone Marrow Transplant project and Autism Network Project with New York University. His research interest is in social networking.

Information Systems & Computing
Academic Professionals

STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 1055-3096