

Moving the Computing Curriculum toward Social Computing

Brian Hammond
Seidenberg School of CSIS
Pace University
bh72680w@pace.edu

The future direction of the Seidenberg School of CSIS

Pace University's Seidenberg School of CSIS has a strong curriculum that offers courses in both theory and practice. In the author's opinion, there's a stronger emphasis on practice than theory at the graduate CS level, which is where this paper focuses on, while the undergraduate CS curriculum at Pace is heavily based on theory. The focus of these curricula are designed *correctly* in the author's opinion. While "industry complains that graduates lack working knowledge of [relevant] technologies" [4], one may argue that at the undergraduate level, this is expected and not actually a problem as industry should probably try to hire from the graduate level more often than the undergraduate level to find workers balanced in theory and practice. The undergraduate student of Computer Science must first learn the theoretical underpinnings of Computer Science before tackling the state of affairs in practical industry applications of Computer Science.

The M.S.C.S. curriculum is not without its faults however. The Seidenberg M.S.C.S. curriculum catalog states that the curriculum "has been designed to accommodate both the student who is new to the field as well as the one with an undergraduate computer science degree" [9]. This is unfortunate. It is the author's opinion that a graduate degree should absolutely require a prospective student to have an undergraduate degree in a closely related (if not the same) field. It has been the author's experience that fellow students in the M.S.C.S. program with little Computer Science background have an immensely difficult time with the basics needed to effectively learn at the graduate level.

There is a growing change of focus for Computer Science curricula in recent years towards users of the technology instead of on technology itself [7]. It would be unwise for Pace University to not adopt a few changes in its CSIS curriculum to focus more on the growing focus on social computing. The concern is at what degree level should this refocusing occur at? The author's position is that the M.S.C.S. curriculum is the correct place to focus more attention on social computing issues since the M.S.C.S. curriculum is properly biased towards practice and social computing is itself biased towards practical uses of technology.

The added focus on social computing may help boost enrollment in the M.S.C.S. program.

The average (Computer Science or other) student is incredibly well connected to her peers, thanks to technologies such as cellular phones, texting (SMS), instant messaging (IM) and popular socially-focused websites such as FaceBook. A focus on social software in the curriculum might bring up enrollment as the student can see a natural fit between their studies and their socially well-connected life.

The following are three hypothetical courses have been designed to aid in making the aforementioned changes in direction a reality for Pace's M.S.C.S. curriculum.

Core requirement: basics of social computing

Basics of social computing is a core requirement course that offers students a deep understanding of the emerging world of social computing. Social computing and social software in general emphasize the inherent nature of humans as social beings. Social software allows interactions between people, employing one or more of the following: identity, presence, relationships, conversations, and groups [2]. Social software often includes wikis, blogs, email, forums, and mailing lists to name a few.

This course places special emphasis on social computing on CSIS-related topics including collaborative distributed software development, source code control, distributed pair-programming (xP), etc. Many software development teams in industry today are globally distributed. Using social computing tools and techniques, the student will gain an understanding of how to be an effective global team member, becoming quite desirable to industry in the process.

The described course fits the proposed points of focus for the new direction of Pace's M.S.C.S. curriculum by:

- emphasizing modern industry practices and environments,
- focusing on current trends such as social computing,
- focusing on using technology instead of simply learning theory with little given understanding of its uses.

Elective: localization and internationalization in software development

The focus on social computing includes the notion of *universality* [1] which emphasizes software that can be used by a myriad of peoples. One practical issue related to universality is developing software that is locale, language, and user aware. The most practical application of this in software development is referred to as *localization* and *internationalization*.

This course will provide a technical foundation for understanding how to globalize software. It is of paramount importance that nontrivial modern software not limit its user interfaces to one language or world region. The course covers an in-depth look at the history of text in computing including the development of ASCII, UTF-8, and in general UNICODE [8, 11]. The course will teach practical, common methods for supporting multiple languages (e.g. GNU gettext

[10]). Finally, localization is considered in a broader context than simple textual manipulation to include methods of understanding users and their cultural backgrounds.

Interdisciplinary elective: Identity and Group Dynamics in Social Software

The territory of the technology is not what's unexplored. What's relatively unexplored is the idea that groups are first class members of the system, entities unto themselves, and that the nature of group dynamics will powerfully effect the way the technology is and can be employed as a result [3].

Identity and Group Dynamics in Social Software is an interdisciplinary course with the Psychology department. Interactions between people are what drives new software, especially web software. It is crucial to understand the basics of human nature as it applies to social software.

This course is a study in *Group Dynamics* [5] as it applies to groups of people that use social software. The goal of the course is to provide a deep understanding of the psychology of people in group settings such that we as computer scientists can design software that is increasingly socially-based. A possible text for the class is [6].

The course also treats the notion of user *identity* in social software including a software developers' responsibility to protect her users from identity theft. Identity theft becomes more dangerous as software becomes more socially enabled. It is important to give students an understanding of the issues behind identity and how to protect it in software.

Course Replacements

To make room for the new courses recommended earlier, we should discard courses that do not seem relevant given our modernized objectives based on social computing, global awareness, and practical industry-driven skills.

The first course that comes to mind in the M.S. in C.S. curriculum is *CS 630 Intelligent Agents*. The course offers to teach students about "software that learns from its own experience" which is a laudable goal and might be of general use in designing software. In general, agent technology is far from a practical, proven, everyday technique used in industry.

CS 628 Automata and Computability is another class that I feel can be safely replaced given our objectives. While low-level theoretical knowledge may lay the foundation for computer science, a bias more towards practice will more benefit graduating students entering industry. Personally, in my 10 years of industry experience, I have never encountered a situation in which my understanding of pushdown automata or even Turing Machines has helped me accomplish a practical goal. Granted, theory helps shape one's overall world view so I am wary to downplay theoretical courses.

References

- [1] Tim Berners-Lee. The mobile web, 2007. [Online; accessed 30-June-2007].
- [2] Stuart Butterfield. Post on the devices in social software, 2003. [Online; accessed 30-June-2007].
- [3] Alicia Cervini. Network connections: An analysis of social software that turns online introductions into offline interactions, 2003. [Online; accessed 30-June-2007].
- [4] Peter J. Denning. The profession of it: crossing the chasm. *Commun. ACM*, 44(4):21–25, 2001.
- [5] Donelson R. Forsyth. Group dynamics resource page. [Online; accessed 30-June-2007].
- [6] Donelson R. Forsyth. *Group Dynamics*. Wadsworth Publishing, fourth edition edition, 2005.
- [7] Ben Shneiderman. Web science: a provocative invitation to computer science. *Commun. ACM*, 50(6):25–27, June 2007.
- [8] Joel Spolsky. The absolute minimum every software developer absolutely, positively must know about unicode and character sets (no excuses!), 2003. [Online; accessed 30-June-2007].
- [9] Pace University. Seidenberg school of computer science and information systems catalog, 2006. [Online; accessed 30-June-2007].
- [10] Wikipedia. Gettext — wikipedia, the free encyclopedia, 2007. [Online; accessed 30-June-2007].
- [11] Wikipedia. Unicode — wikipedia, the free encyclopedia, 2007. [Online; accessed 30-June-2007].