Chapter Three

Teaching Computational Literacy Through Game Design

Ingrid Sturgis and Todd Shurn

> The age of ubiquitous computing clearly has arrived, and computational thinking has become a valued and necessary intellectual commodity if one is to be literate in the 21st century (Way, Cassel, Pearson, Wolz, Tatar, and Harrison, 2010).

In today's pervasively connected world, it's imperative for journalists to develop computational literacy. For journalism students, it's even more critical to develop such competency because the digital revolution has demolished business models that have sustained the profession for more than a century. Despite this disruption, scholars, journalists, and computer scientists say the same technological advances can also empower a new generation of professional programmer-journalists and assist them to engage online news audiences in the process.

We address this reality by presenting an experiment in Cooperative Expertise in which Computer Science and Journalism faculty collaborate to share cross-discipline experiences and subject-matter expertise. It explores the challenge of this prototype for team-teaching cross-disciplinary game-design courses – one for journalism students and one for computer-science students. We describe the prototype collaboration, including the approach used, faculty analysis of the outcomes and the potential for future course collaborations.

The digital revolution has transformed the media, disrupting traditional methods of reporting, distribution, and news presentation. It has led to an environment where anyone can be a publisher, and where the barrier to entry for those publishers has greatly diminished. In addition, the development of high-speed Internet-connected tablets, smartphones, touch-screen displays, very large screens, and small monitors provide journalists with an evolving array of platforms to imagine, redesign, and create innovative multimedia news products.

One result of these changes is a call for schools to teach students to broaden their journalism skills to become "programmer-journalists" in order to take advantage of this ubiquitous self-publishing, technology-driven marketplace (Winer, 2011). One way to start, according to Winer, is for journalism schools to encourage students to manage their own publishing infrastructure, eventually making students more comfortable with infrastructure development.

Even now, it is clear that major and minor news organizations are tapping programmer-journalists to produce multimedia stories that incorporate reporting, analytical data, and relevant social media. For example, traditional news organizations like *The Washington Post* and *The New York Times* seek to hire journalists with computer-programming skills to develop tools that facilitate news aggregation, data visualization, human collaboration or crowdsourcing, and mobile computing.

Progressive journalism schools like Northwestern University's Medill School of Journalism, Media, Integrated Marketing Communications offer scholarships to teach programmers to become journalists. Students admitted to these graduate programs have backgrounds in computer science, mathematics and quantitative sciences. Their common background thread is demonstrated problem-solving involving extensive data analysis. According to Flew, Spurgeon, Daniel, and Swift (2012), programmer-journalists are in demand because of several factors: Big data — the availability of large datasets released by the government, database query software, social media ubiquity, and a continually expanding digital economy. The programmer-journalist is establishing new media value, according to Francisco-Revilla (2012), by devising methods to analyze digital artifacts such as Twitter "tweets" and Facebook "likes," comments, and memes to explain complex ideas.

As a result of new media trends, today's journalism students must master traditional newsgathering skills as well as develop technical expertise to become literate in computational journalism, which involves using analytical techniques for real-time social-media parsing. Although technology has fundamentally altered the way journalists gather, create, and disseminate news, it doesn't change the central purpose of journalism, which is to provide citizens with what Kovach and Rosenstiel (2001) call the "accurate and reliable information they need to function in a free society." Consequently, programmer-developers are essential to the future of journalism.

Given the impact of these media trends on journalism careers, students can benefit from developing the logical thought patterns of computer scientists and from a better understanding of computer science's impact in the world today. In addition, researchers, scholars, and instructors must begin to define what is meant by computational fluency and how it can help improve journalism student outcomes.

This article explores an effort by two Howard University faculty members to collaborate to teach game-design to provide journalism students with a project-oriented programming experience. The class also exposes computer science and journalism students to collaborative cross-disciplinary problem solving that is

prevalent in today's networked society. The course sought to incorporate techniques from technology innovation, game creation, and social media, which some scholars claim can improve student engagement (Heiberger and Junco, 2011). Through course topics, discussions, and assignments, students were immersed in the new media technology trends that have brought transformational change to journalism.

Several important takeaways can be gleaned from the outcomes of this relationship and applied to future courses and to areas of research. It is clear that this pilot can yield important results that benefit computer science and journalism students, including broadening the thinking of the instructors and their respective departments. This article describes our experience in introducing cross-disciplinary collaborative projects to nurture group innovation.

Cooperative Expertise

Cooperative Expertise is a one of three collaborative-teaching models that derive from the Distributed Expertise approach in which faculty from computer science collaborate with those from other disciplines to share knowledge and experiences that eliminate barriers to computational fluency. (Way, Cassel, Pearson, Wolz, Tatar, and Harrison, 2010). In an era when technology has had a transformative impact on disciplines from journalism to teaching to public affairs, there remains a shortage of computing expertise in computer science programs as enrollment has declined. In addition, students must learn to operate outside silos in a more matrixed environment in which teams may work on several goals.

The Distributed Expertise model can offer access to such expertise on campuses, particularly Historically Black Colleges and Universities and other colleges that may not be able to offer new courses because they experience limited resources. Those impediments may include weak course enrollment, shortage of experienced computer science faculty, or other specialized expertise. Today, faculty may be able to utilize electronic solutions to solve such limitations, including the use of course management systems or new media tools such as Skype or Google Hangouts to bring in specialized expertise (Way et. al., 2010).

Other models of collaboration in the Distributed Expertise approach include Remote Expert with Local Facilitator (RE), in which an expert faculty member who teaches at a one institution benefits the faculty and students at another institution. Special Resource (SR) is a format in which a faculty expert is brought in to guest lecture on a specialized topic in a regular class or through a recorded format.

According to Wolz, Cassel, Way, and Pearson (2011), the Cooperative Expertise model permits "faculty from two distinct courses identify and share expertise with faculty and students in other courses to support learning goals in both courses" (p. 333). Each instructor takes a turn serving as a facilitator, and

students benefit from exposure to the concepts of the collaborative relationship as well as sharing subject matter expertise. All three models offer research and teaching innovation. Wolz et. al. acknowledged the trends in team teaching, interdisciplinary collaboration, and the expansion of computing into all disciplines, and offered a roadmap for faculty members to follow in introducing this concept to the Howard campus.

For the purpose of this chapter, the Cooperative Expertise example focused on two Howard University instructors who were on the same campus but in different disciplines and with different expertise. Two upper-level courses are described: A Game Engine Programming course for computer science majors and a multimedia News Game design course for communications students.

This project fit with Howard University's strategic planning efforts as outlined in its Presidential Commission on Academic Renewal (PCAR):

- To promote new interdisciplinary programs;
- To encourage all students to conduct research;
- To infuse technology thoroughly into its teaching and learning processes; and
- To enhance student learning by developing pedagogies that will take advantage of new and emerging technologies (Howard University, 2010).

The gaming course also dovetailed with a collaborative effort by Arts & Science, Communications, and Engineering faculty to develop a cutting-edge interactive media and gaming undergraduate curriculum. It would include a professional certificate training program and innovative research center for the study of interactive media and gaming.

Background

The project received support from a National Science Foundation CPATH: Distributed Expertise grant that is a collaboration with Villanova University, Virginia Tech, and The College of New Jersey (TCNJ). The CPATH Distributed Expertise Project started in 2008. Its goal was to diffuse computer expertise across disciplines, as well as to develop resources to support the exploration of computing and its impact on other disciplines. According to co-Principal Investigator Associate Professor Kim Pearson of The College of New Jersey:
Part of our vision is that this could be a way of providing CS expertise to disciplines that are becoming computing dependent, such as journalism, while helping CS students understand the nuances of working with content from different knowledge domains (Para. 4).

The CPATH support allowed Howard University faculty to plan and develop the Cooperative Expertise project. It was piloted as a directed study course in

spring 2012, with a four-student subgroup. (Originally, the group included two journalism majors, one public relations major, and two computer science majors. One journalism student later dropped the course.) Each instructor was responsible for his or her course, including developing its content, assignments, and grades, and retained responsibility for syllabus and course administration. The instructors used separate syllabi appropriate for the accredited course descriptions.

The computer science students were enrolled in Game Engine Programming, a computer science course; and journalism students were enrolled in a course called News Game. The four-student interdisciplinary team worked collaboratively using Scratch, a visual-programming language developed by Massachusetts Institute of Technology to teach computing skills to middle school students. Scratch can be used to create computer games, interactive stories, graphic artwork, and computer animation with music and other sound effects.

The News Games class explored the emergence of games as a tool for journalistic storytelling. Students analyzed other games and learned to use the genre as a news storytelling technique. Students also examined how gaming techniques were being incorporated in public relations and marketing campaigns. The course ultimately focused on two things:

1. Discovering, documenting, and evaluating the design components of a variety of games in news, public relations, and education.
2. Learning to use Scratch to create a simple game based on the Associated Press Style Book.

The journalism team was tasked to develop an Associated Press style game for instructional use. The resulting game's story line featured a college-student band traveling to gigs during spring break. The Scratch characters moved from one place to the next on a map of the United States when a player successfully answered the AP style question. The journalism students also generated specific questions related to the theme, and researched design elements such as music, background images, and sprites. The computer science students were to help build it. The journalism students were to contribute 10 multiple-choice questions each, but the number was cut to five questions as it became clear that programming character movements in Scratch was more time consuming than expected. The class emphasized the value in journalism students being able to experience the type of thinking computer science requires — a very specific, step-by-step problem solving. The combined class met once each week for a 90-minute session. The computer science students were part of a larger class that was assigned to subgroups to complete various projects.

The goal of the project was not only to integrate emerging technologies into the journalism curriculum, but also to help journalism students become familiar with the culture of the computer science and to understand core computing concepts through the use of Scratch programming language. News Game also was conceived to help improve student engagement in learning to appropriately use Associated Press style. The project sought to test whether the use of gaming

technology would motivate students to acquire and apply knowledge to become better editors and journalists. The ultimate goal was to improve student persistence, which would lead to mastery of AP style and more success in editing.

The Associated Press style was used because some faculty members have more or less discovered that some students did not learn the basics by the time they were enrolled in the upper-level copy-editing course taken by juniors and seniors. Although some publications, like *The New York Times* and *The Washington Post*, have developed their own guidelines, a basic knowledge of AP style is an important editorial skill required to work in most newsrooms, public relations, and advertising jobs. This knowledge gap could severely affect student outcomes in the copy-editing course as well as have a negative impact on the quality of student portfolios, resumés, and job applications prepared by seniors as they entered the journalism job market.

The course also endeavored to provide journalism and computer science students an opportunity to collaborate. The computer science students were to contribute technical guidance and programming expertise and the journalism students were to provide journalistic ideas, writing, research, editing, and promotional expertise. Neither journalism student had experience in developing games or writing code, nor prior knowledge of game design or programming. No prior knowledge of Associated Press style was required. This course was an intermediate option or elective for communications students.

As part of the course, journalism students had a number of assignments and deliverables that took them through the game creation process. They were required to keep a weekly game journal that described the features of various games that they researched and the game mechanics involved in creating the game. They developed directions for a board game as well as explored basic game-creation concepts. The final project proposal included a storyboard, directions, and sprite characters, as well as Associated Press rules in a multiple-choice format. They also developed a prototype for the final game.

Other student outcomes for the course included: being able to describe fundamental principles of game design; being able to discuss the ethical challenges related to the development of news games; and being able to develop a demonstrated ability to collaborate within a small, interdisciplinary team to design, implement, test, and evaluate small computer games to a degree appropriate to the student's academic background.

On the computer science team, participating students were part of a larger course called Game Engine Programming, which was a technical elective course emphasizing the coding constructs, framework, and assets necessary to create an original game. Course prerequisites for students included Computer Science III and junior standing. The students who enrolled were expected to be familiar with Java, C++, C#, Visual Studio, and Eclipse. The course curriculum contained ABET-accredited computer science core and advanced content for algorithms, data structures, programming languages, and software design.

Typically, Howard University computer science students graduate without ever enrolling in a course requiring software development as a member of a team with non-computer science students. This exclusively computer science perspective does not prepare students for the cross-disciplinary approach adopted by many successful, creative organizations. It is especially unrepresentative of the computer and video-game industry where programmers are just one element of a creative team.

In today's matrixed work environment, an employee may work on multiple projects at the same time as part of different teams that are managed by different project managers. Wolz et. al. (2011) said it is crucial that computer science students get out of their silos to gain experience working as part of a team to develop the soft skills needed for the complex organizational structure that requires employees who can effectively communicate with other members of the team.

Recognizing this reality, collaborative game projects were established with journalism students to develop scripted, story-driven games and interactive media. The class met for 90 minutes twice per week with typically one class per week conducted jointly with computer science and journalism students. There were six Game Engine Programming students. Four of whom were able programmers that had participated in programming competitions, while the other two were less capable. One was a struggling computer-science major, and the other was a senior mechanical-engineering student with limited programming skill. All the students were motivated to take the course because of their love of video games. The computer science students were responsible for three course assignments: game play review, an original game created for competition submission, and Kinect Sensor routine.

Students in both classes were required to participate in the game review event, the HU Roast and Toast, a scripted live event for students, faculty, and interested gamers conducted in the School of Communications screening room. The students reviewed selected console games (PS3, Wii, Xbox) for their story, graphics/audio, controller interface, and future extensions. Among games reviewed were Batman: Arkham City, Soul Caliber 5, Call of Duty: Modern Warfare 3, Elder Scrolls V: Skyrim, Dance Central 2 and Assassin's Creed: Revelations, Just Dance 3, and Dance Central 2. These game reviews were designed to help the computer science students to develop the skills of critique.

Game Engine students were assigned to create a game for submission to the Microsoft Imagine Cup or the ESA National STEM Video Game Challenge. They also were assigned to create an original routine using the Microsoft Kinect Sensor API. The routines were a sequence of motion-captured Kinect Sensors with associated scores, but not complete games or game concepts. Game Engine students were introduced to the Kinect Sensor API through Visual Studio using examples discussed in detail during class sessions. These examples were extended to become the basis for their Kinect Sensor routines.

Student outcomes for the computer science students in Game Engine Programming included:

- Recognizing the value in collaboration with journalism students.
- Working efficiently and creatively with students from other majors.
- Seeking students from relevant non-computer science majors to participate in game projects and competitions.
- Collaborating across disciplines to develop game and interactive products.

In order to begin planning for our Cooperative Expertise pilot project, the faculty members conducted two telephone meetings over winter break to define the goals of the collaboration. Admittedly, this pilot project was very ambitious. Not only did it seek to impart computational thinking and design skills in journalism students with no technology background, but it also sought to test the theory that the game created for the course would encourage more students to become proficient in AP style. In addition, it would seek collaboration between two groups of student with ordinarily little in common. Our over-arching objective was to examine how a journalism course and a computer science course could be coordinated to enhance learning and support faculty expertise in an interdisciplinary manner through the development of deliverables — a Scratch game that teaches Associated Press style and development of the scripted live event HU Roast and Toast.

As part of the planning, the faculty members collaborated on course activities for two separate classes. The Game Engine Programming course for computer science students and the News Game for journalism students met jointly each week during the spring 2012 semester. Course activities included computer science students mentoring and teaching Scratch to the journalism students. The journalism students researched and created the storyboard and script for the game. The four-person team worked collaboratively with Scratch programming language to develop a game. The journalism students were also charged with promoting the Roast and Toast event as well as editing videotape from the event.

At the end of the semester, we were anxious to synthesize our experiences in the pilot project. In post-mortem discussions and retrospective analysis, we discussed what we thought worked and what did not. We both emphasized the value of the journalism students experiencing the type of thinking computer science requires — very specific, step-by-step problem solving. The computer science students likewise had an opportunity to enhance their creativity and soft skills as well as gain experience working in a matrixed environment. Both groups of students benefited from learning more about how computational thinking has been infused in today's newsroom and in other areas of professional work.

In our assessment, a number of lessons were learned that could improve subsequent experiences in using the Cooperative Expertise model. What follows is what we feel are the most important lessons about team teaching, expectations,

skill development, and student collaboration to deliver a successful cooperative expertise project.

Team Teaching and Expectations

The project course was planned over winter break and instructors communicated by phone but not in person. Both instructors conceded that a closer face-to-face collaboration to define parameters of the course, opportunities for enrichment, and articulation of combined goals would have resulted in better outcomes for student collaboration. Because neither instructor was familiar with the other's teaching style, nor each other's disciplines, it was difficult to anticipate ambiguous patches, and sometimes expectations were unmet. Our goals may have been too ambitious for the journalism students and the projects too complex for students who did not have a technology background. Similar to what Wolz, Ault, and Nakra discovered in their Cooperative Expertise project, perhaps our expectation that the students complete a fully functioning AP style game was off the mark. Instead, the game in our project was the vehicle to teach the computational concepts. According to that consideration, our project was successful.

But as Eisen and Tisdell (2002) noted, our interaction with the students and each other created an environment that offered multiple meaningful perspectives that could lead to new knowledge being created. In fact, we have decided to continue to participate in developing interdisciplinary courses. We are also working to develop research opportunities based on our experience in the Cooperative Expertise model. Computer Science and Journalism faculty will continue to develop curriculum to prepare effective programmer-journalists. Imbedded in the computer science and journalism curriculums will be joint class sessions, projects, and competition submissions. In the future, in addition to improving the student collaboration process, we will seek to develop improved assessment tools to foster collaboration among reluctant students.

Skill Building

Although News Game's syllabus was modeled on an existing gaming course, adjustments had to be made for a smaller class size and for journalism students. Faculty also reviewed several Scratch games to demonstrate the attributes of the software to journalism students. It was apparent that previous knowledge of basic programming or an intense Scratch boot camp would have been more helpful for the journalism students. In looking back on the project, the students did deliver a game based on the AP style that was created in Scratch, but the students reverted to their silos with the journalism students producing the creative content and the computer science students building the game incorporating the creative content. In the end, both groups of students were able

to deliver on the project, but it was not clear that they benefited as much as they could from the collaborative efforts.

However, it was clear that the students from both groups were reluctant to collaborate inside and outside the classroom environment. Journalism students were encouraged to seek one-on-one Scratch assistance from the computer science students, yet they hesitated to do so. Similarly the computer science students were expected to engage in peer-to-peer Scratch interaction with journalism students. Neither group attempted to collaborate outside class sessions. Male-female differences may have had some impact on collaboration, as the two female journalism students seemed hesitant about working with the all-male cohort of computer science students with programming expertise. In addition, because game development was novel to the journalism students, they may have been unsure what questions they needed answered in order to implement in their game concepts in Scratch.

Team-building Exercises

The two classes convening once per week for joint sessions would have benefited from ice-breaking exercises to get to know one another on a more personal level prior to starting course projects. Such exercises might have resulted in improved communication and collaboration. Cross-disciplinary teamwork was not a component of the Game Engine programming course grade. As such, even though teamwork was promoted and encouraged, it did not a have meaningful impact on the computer science students' games. Computer science students didn't collaborate with the journalism students for any project. The computer science students didn't seek the journalism student's input about their XNA or even producing the video with the journalism students who had documented expertise. The "go it alone" approach resulted in less functionality in their projects.

Conclusion

The Cooperative Expertise model is a valuable tool in helping to infuse computational thinking among college journalism students. The opportunity to work with computer science colleagues who view the communications profession from another perspective has been helpful in understanding the technological changes happening in society. As professors coming from different disciplines, we were able to see new opportunities for collaboration as well as gain another perspective on the range of skills that students need to be successful in today's work environment. Some potential limitations included difficulties in scheduling classes that met at the same time and having them cross-listed in the school catalog.

We, however, made some assumptions about collaboration by the students that did not bear out. A key concern was the balancing individual responsibility

with group collaboration that required the computer science students to tutor the journalism students to learn Scratch and the journalism students to deliver the content and other assets to build the game with the computer science students. We did not foresee the cultural differences between the journalism and computer science students. Also, while we assumed students were receiving an immersive experience by dint of their working together in the same room, that may not have been true in practice. Student efforts did not meet our pedagogical goals of student cross-discipline collaboration nor sharing of subject-matter expertise. In the end, the two teams reinforced typical undergraduate tendency toward individualistic behavior.

References

Anderson, C.W. (October 26, 2011). *Notes Towards an Analysis of Computational Journalism*. HIIG Discussion Paper Series No. 2012-1. Available at SSRN:http://ssrn.com/abstract=2009292 or http://dx.doi.org/10.2139/ssrn.2009292

Buskirk, E. V. (2010, April 7). Will Columbia-trained, code-savvy journalists bridge the media/tech divide? Wired.com. Retrieved from http://www.wired.com/business/2010/04/will-columbia-trained-code-savvy-journalists-bridge-the-mediatech-divide

Cohen, S., Hamilton, J. T., & Turner, F. (2011). Computational journalism *Communications of the ACM*, 54(10), 66-71.

Cronk, M. (2012). Using gamification to increase student engagement and participation in class discussion. *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, Vol. 2012 (1), pp. 311-315.

Diakopoulos, N. (2010, January). A functional roadmap for computational journalism. [Online]. Retrieved from http://www.nickdiakopoulos.com/2011/04/22/a-functional-roadmap-for-innovation-in-computational-journalism/

Eisen, M. J., & Tisdell, E. J. (2002). Team teaching: the learning side of the teaching-learning equation. *Essays on Teaching Excellence*, 2002-03, 14. Retrieved from http://cte.udel.edu/sites/cte.udel.edu/files/u7/v14n6.htm

Flew, T., Spurgeon, C., Daniel, A., & Swift, A. (2012). The promise of computational journalism. *Journalism Practice*, 6(2), 157-171.

Francisco-Revilla, L. (2012, June). Digital libraries for computational journalism. In Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries (pp. 365-366). ACM.

Heiberger, G. & Junco, R. (2011). Thriving in academe: Reflections on helping students learn. *NEA Higher Education Advocate*. Retrieved from http://blog.reyjunco.com/pdf/ HeibergerJuncoNEA.pdf

Howard University. (2010). *Presidential Commission on Academic Renewal, Defining the Future: Enriching the Ground on Which We Stand*. Retrieved from http://pcar.howard.edu/PCAR/Reports/PCAR-Executive-Report-6-4-10-final.pdf Washington, D.C.: Thornton, Alvin.

Kirchhoff, S. M. (2009). The US newspaper industry in transition: CRS report for congress. *Congressional Research Service*, report, 7-5700.

Pearson, Kim (2010, January 6). *Distributed Expertise in Enhancing Computing Education With Connections to the Arts*. [blog post]. Retrieved from: http://pearson.net/?p=552

Pulimood, S. M., Shaw, D., & Lounsberry, E. (2011, March). Gumshoe: A model for undergraduate computational journalism education. In Proceedings of the 42nd ACM technical symposium on Computer science education (pp. 529-534). New York, NY: ACM.

Rosenstiel, T., & Kovach, B. (2007). *The Elements of Journalism: What Newspeople Should Know and the Public Should Expect, Completely Updated and Revised*. New York, NY: Three Rivers Press.

Sandford, R., Ulicsak, M., & Facer, K. (2006). Teaching with Games: using computer games in formal education. *Futurelab, Bristol*. Retrieved from http://media.futurelab.org.uk//resources/documents/lit_reviews/Serious-Games_Review.pdf

Way, T., Cassel, L., Pearson, K., Wolz, U., Tatar, D., & Harrison, S. (2010). A Distributed Expertise Model for Teaching Computing Across Disciplines and Institutions. Unpublished paper presented at the International Conference on Frontiers in Education: Computer Science and Computer Engineering, Las Vegas, NV.

Winer, D. (2011, January 24). [blog post]. Retrieved from http://www.niemanlab.org/2011/01/dave-winer-how-can-universities-educate-journo-programmers/

Wolz, U., Ault, C., & Nakra, T. M. (2007). Teaching game design through cross-disciplinary content and individualized student deliverables. *Message from the Program Committee Chair*, 8.

Wolz, U., Cassel, L. B., Way, T., & Pearson, K. (2011). Cooperative expertise for multidisciplinary computing. In proceedings of the 42nd ACM technical symposium on computer science education, Dallas, Texas, 9-12 March, (pp. 329-334). New York, NY: ACM.