Shai Simonson

A Post-Baccalaureate Undergraduate-Level Program in Computer Science

A one-year pilot education, tuition-free, that could be shared under an open-source license with the rest of the world.

LISA HANEY

What options exist for non-CS college graduates who decide later in life to consider a career in computer science? This question is especially relevant today with the shortage of qualified computer scientists both in industry and academia. Many universities do not allow college graduates to enroll in an undergraduate degree program. Moreover, taking the courses à la carte would take four years or more. Master's degree programs are not an option because a more extensive preparation is required. Java or Oracle certification, or system administration courses abound, but they are vocational training and do not provide a broad view of the discipline.

The concept of ArsDigita University (ADUni), the brainchild of Philip Greenspun, was to offer bright and motivated college graduates a free undergraduate-level education in computer science in just one year with minimal interruption to their lives. All materials, including videotapes of lectures, would be shared under an open-source license with the rest of the world. ADUni was funded by the ArsDigita foundation.

From September 1, 2000, through July 15, 2001, 34 talented and motivated college graduates attended a one-year intensive and complete undergraduate computer science program, tuition-free. ADUni offered a high-quality undergraduate computer science education, preparing its graduates for jobs in the industry or graduate school in computer science. Enrollment was restricted to superior students holding at least a B.A. or the equivalent in any field outside computer science. Certificates of completion were granted but no degrees awarded.

ADUni is described here in some detail, and an ambitious plan sketched out to leverage this pilot year into a broad national program that addresses the question first raised. The plan would need the cooperation of industry, academia, private foundations, the NSF, and professional organizations. The purpose of this "On Site" is to start a dialogue and plant a seed.

Students

Students came from all over the U.S.; a few resided in Europe. They were 28 men and 6 women whose ages ranged from 22 through 68, with the median age 30, mean 33. Their backgrounds included law (3), medicine (3), humanities (6), journalism (2), fine arts (5), engineering (5), mathematics (3), social science (5), and education (2). Some had no exposure to computers, but most had significant experience with using computers. Some were even self-taught system administrators. There were over 350 applications; 40 candidates were accepted and 36 chose to attend. By the end of the year, two students had dropped out, seven finished the program with some deficiencies, 27 completed all of the program requirements, six with honors.

Curriculum and Schedule

The curriculum at ADUni was linear, with one course per month, each month corresponding to a standard semester course both in content and in contact time. Each course met six days a week, for two hours of formal lecture each morning, and two more hours of informal recitations each afternoon. ADUni students spent an average of 10–12 hours a day in lecture, recitation or working on problem sets. The academic plan was roughly in line with

Course Description

The number of weeks (and hence semester equivalent credits) per course is marked in parentheses. For details regarding texts, requirements, content, and faculty, see aduni.org.

#1: Mathematics for Computer Science (4)
A basic introduction to Calculus and Linear Algebra. Teaches mathematical literacy for science and engineering.

#2: Structure and Interpretation of Computer Programs (4)
An introduction to programming and the power of abstraction, using text of the same name.

#3: Discrete Mathematics and Probability (4)
Emphasis was placed on providing a context for the applications of the mathematics within computer science.

#4: How Computers Work (3)
The basics of digital logical design, computer organization and architecture, including assembly language, processor design, memory hierarchies, and pipelining. The distinction between RISC and CISC architectures was emphasized.

#5: Object-Oriented Program Design and Software Engineering (4)
The concepts of the object-oriented paradigm using Java, and the basic principles of software engineering are emphasized. Event-driven programs with GUIs are designed. The large group project was to develop a server/client for Gnutella, a distributed music-sharing system similar to Napster without the central server.

#6: Algorithms (4)
The design and analysis of algorithms and how to handle NP-complete problems. Emphasis was placed on the trade-off between experimental and theoretical results.

#7: Systems (3)
Case studies of working systems provide comparisons and contrasts. This course's group project was to write an NSF systems proposal to fund a middleware product for announcement RFP01-63.

#8: Software Engineering for Web Applications (4)
The basics of designing a dynamic Web site with a database back-end. Included scripting languages, cookies, SQL, HTML, XML, and VoiceXML. Computer-human interface and the graphical display of information are emphasized. This course was built around a month-long project. Crucial to the project was the identification of a client with whom the group must work designing specifications, implementing them, evaluating, and testing.

#9: Theory of Computation (3)
A theoretical treatment of what can be computed and how fast. Applications to compilers, string searching, and control circuit design were discussed.

#10: Artificial Intelligence (1)
An overview of AI from the technical and the philosophical points of view.

#11: Unix Workshop (1)
A self-study workshop to review and/or learn a wide range of Unix tools. This course was pass/fail.

#12: Database Management Systems (3)
Database systems were studied from the physical layer of B-trees and file servers, to the abstract layers of relational design.

Month 12: Applied Probability (2)
This course extended the discrete probability learned in course #2, and focuses on applications rather than proofs. A problem set based on identifying tumors (using magnetic resonance imaging) was done using Matlab.

the ACM CC2001 guidelines; the courses were exclusively computer science and related mathematics. There were 39 total weeks of courses (equivalent to 39 semester credits) distributed roughly into the following categories: Mathematics (10), Software (13), Hardware (3), Systems (6), and Theory (7).

Because every person took the same courses in the same order and at the same time, there were opportunities to build connections between the courses. Hence, the sequence of courses was more unified than otherwise possible at a regular college. For example, projects built in the Java software engineering course were expanded on in "Algorithms." Database projects included the optimization of previously designed Web sites. Discussion of recurrence equations in discrete math built on experience with recursion in "Scheme." A list of courses with brief descriptions are boxed on the preceding page.

Colloquia

A nice feature of ADUni was its monthly colloquium. Every month a prominent speaker was invited from industry, academia, or other parts of the greater computer science community. Our location and mission allowed us to recruit some well-known speakers, including Gerry Sussman, Richard Stallman, and Mike Sipser. A standard university CS department would never have the time, interest or resources to consistently round up such a fine set of speakers for an undergraduate audience. The colloquia were videotaped and are available for free on the ADUni Web site.

Physical Space

The physical space at ADUni was crucial to its success. It created an environment in which the social and academic center of the students' (and staff's) lives was in the lab, rather than the dorm, pub, or library, as it might be at a more traditional school. This not only encouraged collaboration, but also allowed for more flexible pedagogy. The contrast with a typical college where classmates see each other twice a week for 90 minutes, and perhaps at an arranged study date, was significant. For example, at ADUni an exam was never given in the lecture hall. Each student simply worked at his or her own workstation in the lab. Online access was available during lab and exams, without special arrangements. The distinctions between exam, assignment, problem set, and project, sometimes got blurry. Ad-hoc group review sessions were easily organized by the teaching staff and by students.

The lab was a large open room with 40 stations. Each station was equipped with a 3'-by-6' table, a small file cabinet, and a Linux workstation with Internet access. Students had their own textbooks and space. Tables up in front of the room were reserved for teaching assistants; bookshelves contained a small library. In the center of the room, was a movie theatre-style popcorn machine, a watercooler and a portable whiteboard.

The lab was the nerve center of ADUni. It was rarely empty at any time of the day or night. For many students it was their home for a year; they left only to sleep.

Faculty and Staff

ADUni had one full-time faculty member (doubling as the director) and three full-time teaching assistants (TAs). Many part-time faculty and TAs were hired from MIT and nearby high-tech companies. Each course had one lead faculty member in charge of lectures, lecture notes, problem sets, and overall course design. The TAs, in addition to serving as resources for students, assisted with grading, tutorials, problem set design, recitation sections, and Web site maintenance. A full-time administrator was assisted by two volunteers to handle the day-to-day logistics, including guest speakers, admissions, faculty recruitment publicity, Web postings, payroll, social events, and evaluations. All lectures and many recitations were taped, most by a professional videographer, the rest by students and staff volunteers.

Evaluation

ADUni was a huge success. The students learned. They completed a rigorous curriculum (readings, exams, problem sets, and projects) and performed better overall, according to their professors, than the typical undergraduate class. Graduation was an

emotional event, as many of the 34 graduates prepared to start new careers. Within the next five years, I estimate 2–5 students will enroll in Ph.D. programs, 20–25 will become software engineers, 4-6 will apply the knowledge to their established careers, and 2–3 will move on to new careers related indirectly to computer science.

All lectures and recitations were videotaped and are available free to the public under an open-source license on the Web site. The problem sets, solutions, exams, and lecture notes are also available, as are student evaluations and suggestions for every course. Many people have made use of the videos and other materials provided online. The site gets bandwidth traffic of 12G per day and plenty of good feedback.

The ADUni concept has some advantages over the standard undergraduate model:

- The curriculum was designed as a symphony rather than independent pieces.
- The environment encouraged students to work more seriously and focus.
- A natural collaborative environment developed among the students.
- Work was intense and focused on one topic at a time, a highly effective style.

Future Plans

ADUni was the pilot study for an ambitious but exciting model of education in computer science. This new model addresses the need for more qualified technical people in the workforce as well as the shortage of graduate students in computer science. It efficiently taps into the reservoir of talented college graduates that dream of computer science as a career, but are faced with either vocational programs, or multiyear academic commitments.

An independent national program can leverage our first year's startup effort and experience. The program, based on the ADUni pilot tested curriculum, would not grant a degree but would be accredited by ACM or CSAB/ABET. Colleges and universities would sponsor a network of satellite sites, and provide lab space and full-time mentors. The continuity and community would be provided by these local sites and mentors. Instructors would be experts in academia or industry brought in on a per-course basis. The Web could be used for weaving connectivity between sites. A conference once a year would allow all sites to meet and share results.

This would not be a distance learning program. What distance learner can simulate the ADUni lab? What distance learner dedicates 12 hours a day?

This would *not* be a distance-learning program. What distance learner can simulate the ADUni lab? What distance learner dedicates 12 hours a day? The national program should strive to mimic the success of ADUni, which was an on-site phenomenon, hence the need for local mentors.

THE EDUCATIONAL DETAILS HAVE BEEN DESIGNED and tested. The pilot year's graduating class will provide the reputation. The idea needs peoplepower, money, organization, and support from the computer science community. Costs can be covered by tuition and joint sponsorship with the NSF, private foundations, and academic institutions.

This educational effort can bridge the communication gap between academics and industry. It can offer computer science to the largest untapped market in the world—bright professionals with a new career dream.

See aduni.org for a more detailed description of the program, links to all our course materials, and the current status of our graduates.

SHAI SIMONSON (shai@stonehill.edu) is a professor in the Department of Computer Science at Stonehill College, North Easton, MA, and the director of ADUni.