Hello and welcome to our 2011 edition of the CS program newsletter!

An exciting year has passed. Many of you will wonder how the budget cuts affect our programs. In the newsletter last year I wrote that NAU has done an amazing job of weathering the crisis. Until lately, we had been cautiously optimistic, as the cuts were at the lower end of what was expected. New developments put a damper on our confidence, however, as NAU’s budget is going to be reduced by another $4 million, in addition to the original cut of $25.8 million. On the other hand, we have no doubt that our help as engineers and engineering educators is absolutely indispensable to get Arizona’s economy back on its feet.

And here comes the really good news: Our enrollments have continued to grow spectacularly. Our spring 2011 enrollment is up 19% since 2010 and 44% since 2009, which is exceeded by only Mathematics and Statistics (48%) among Flagstaff departments with over 100 students. Our fall 2010 new student (freshman and transfers) enrollment was up 24% in our department compared to last fall (up 85% since 2005). Most departments in our college were above the 9% university average.

In the meantime we have considerably streamlined and improved our two Computer Science degrees, i.e. the Bachelor of Science in Computer Science (BS in CS) and the Bachelor of Science in Applied Computer Science (BS in ACS), to better serve the needs of the industry and our students. The BS in CS is an ABET accredited program. As such, it provides more career flexibility and a broader range of job opportunities. The BS in ACS, on the other hand, is a less rigorous, non-accredited program. Career opportunities for this degree typically are in the areas of smaller-scale, hands-on software development, software testing, quality assurance, software technical support, to name a few.

In the BS in CS program, we dropped two liberal studies courses along with one science elective. This allows us to
significantly strengthen the CS core of the program by adding CS 212 (Web Programming), CS 476 (Requirements Engineering) and one more CS elective. We have revamped our BS in ACS program from the ground up. We dropped the foreign language requirement and added CS 112 (Intro to WWW/Internet), CS 122 & 122L (Programming for Engineering and Science) and CS 212 (Web Programming). Also, one fewer technical and one fewer CS elective are required than before.

One other area we made significant progress in is our graduate programs. We implemented the “Integrated Program”, which is a fast track program offered to our best and brightest undergraduate students to get a graduate degree.

I hope you enjoy our newsletter! If you are ever in the area or passing through Flagstaff, please drop by. We are always happy to reconnect with our friends and alums and our ACM student club is always looking for speakers to share insights.

All the Best,
Wolf-Dieter Otte, Associate Chair and CS programs coordinator

CS CAREERS

by Steven M. Jacobs, Adjunct Professor

Computer Science majors are again heading into an excellent career. The “Best Careers in 2011” lists from U.S. News and World Report, CNN, and others point to computer science, software engineering, and related careers (such as IT) as still being in demand. Often these careers appear on lists of well-paid entry-level positions.

I encourage students to create and maintain a current resume. Make sure you take advantage of NAU-provided resources to aid you in your job search, such as seminars (resume writing, interview techniques, etc.) and campus recruiting events. Get a hold of the “Job Search Guide” from the NAU Gateway Student Success Center. Attend the CENFS Career Fairs, even if you are not graduating until much later. Often, employer representatives can give insights as to courses they need you to complete, while they share any internship opportunities.

And, be flexible. Keep in mind many of the jobs may not be precisely what you hoped for – for example, you may be asked to develop software in C++ when you really wanted to do Java, or you may be asked to upgrade a web site using web development tools when you really wanted to build database applications. You may be asked to maintain (i.e. enhance or improve) a large portion of existing, operational code. The key to success in the marketplace is flexibility both in assignments and often geographical preference. It is important to be willing to learn new things and take on new challenges.

Your learning does not stop at NAU. You will be learning new skills and development processes on the job.

STUDENT CHAPTER OF ACM

NAU ACM is the NAU chapter of the Association for Computing Machinery. Computer professionals and enthusiasts are welcome to join, where they will learn about fascinating new technologies and gain a deeper understanding of existing technologies. Members of ACM have easy access to tutoring and study materials, and have a quiet area conducive to studying and completing assignments. ACM also provides unique opportunities for members, such as building and programming a robot, creating video games for the ACM game cabinet, and competing in national programming competitions.

Weekly ACM meetings are open to everyone, and are held in room 105 of the engineering building at 6:00 every Thursday. See http://www.nau.edu/acm for more details!

The ACM room.
A place for CS majors to relax, meet, and work on projects.
Exploring Social Aspects of Software Development

By Dr. John C. Georgas

Software development is a highly social activity, with important design and implementation decisions being made within the context of developer communications and collaborations. The end goal, however, involves the production of technical artifacts, such as requirements specifications, architectural designs, development documentation and running software systems. As a result, software development exhibits a complex interplay of technical and social concerns that are highly interdependent on one another. One of the most critical of these interplays is the one between software architecture and the communication structures of the developers charged with transforming it into a complete, executable software system.

The study of software architecture is concerned with how to best capture a software system’s modular decomposition into components and their interfaces, which makes architecture a key factor in dictating the communication needs of developers who are assigned to the system. This influence, however, is not one-way: architectures are also strongly affected by the existing organizational and communication structures of the developers before a particular development effort and its corresponding architectural design is initiated. Divisions based on technical expertise, the co-location or geographical distribution of developers, and existing communication practices may all influence how a particular architecture is designed. As a matter of fact, organizations tend to develop architectural designs that mirror their existing structure and communication structures – a phenomenon referred to as Conway’s Law.

Research efforts in the scholarly field known as socio-technical congruence have discovered that software projects with a high degree of match between their architecture and the social structures of their developers result in products with higher quality that are more efficiently developed. So, while we know that these socio-technical dependencies are important, it is still challenging to apply insights from their study in everyday development efforts.

In collaboration with Dr. Anita Sarma from the University of Nebraska, Lincoln, I am working toward addressing this challenge by developing tool support for allowing developers to leverage socio-technical congruence information during ongoing development processes. Under NSF support, these investigators are developing the Architecture-centric Social Network Awareness Environment (ArSoNAE) as an extension of the Eclipse integrated development environment, in order to provide socio-technical insights via an architectural perspective.

CS Course Redesign

By Dr. Kefei Wang

CS122 (Programming for Engineering and Scientists) is a large enrollment, lower-division course. The course is required for majors in both Electrical Engineering (EE) and Mechanical Engineering (ME). It covers introductory computer science concepts including algorithms, and gives the student hands-on experience with MATLAB and C++ programming languages.

Last Newsletter, I mentioned that we received a grant from NAU, funded by the Arizona Board of Regents, to redesign courses for increased enrollment while at the same time, improving student performance. Here are some of the highlights of my work accomplished to date:

- The “Question Guided” lecturing format, which integrated Clickers usage, was developed. Students prepared with the reading assignments, and then discussed more in-depth questions in class. Instructors could use Clickers to monitor and guide the learning process. More group discussions and team projects were integrated into the course to enhance communication skills.

- The “Virtual Lab” environment was investigated and developed. Ideally, students were able to participate in an online session to conduct lab work. The lab instructor was able to interact, help students individually by audio/video, and sharing programs.

- The “online office hour”, which was based on the Elluminate® system, was adopted to broadcast the assignment discussions to all students through the Internet. It was interactive and effective just as the traditional office hours.

- More on-line resources and activities to help students better understand concepts. Twitter and Facebook social networks were used in the course to make it more attractive to today’s students.

- Open source programming environments were deployed to dramatically reduce the cost for both students and the college.

These methods have been proved to better fit today’s students who perform better and learn more with varied instructional media and presentation formats.
A FOND FAREWELL

BY ABE PRAILLE, CS LECTURER

The 2010-2011 academic year is longtime instructor Abe Pralle’s final year of teaching Computer Science at NAU. Abe announced his intent to resign at the beginning of the school year and intends to focus his attentions on Plasmaworks, the digital design company he co-founded two years ago, and its game brand Castle Arcadium.

“I’ve always been passionate about game development,” Abe says. “And in fact I’d never planned on being a teacher. After I graduated with my Bachelor’s in 1995, the department asked for my help teaching a lab. The lab turned into a lab plus an intro lecture, and we all thought it’d be just for a semester or maybe a year. And here I am 16 years later!

“I’ve really appreciated my time teaching as it lead to a great deal of personal development. Firstly I became a much better programmer from having to be ready to explain things. Secondly I became a much better manager. Lastly, the summer and winter breaks afforded me the time to develop a custom computer language and game programming framework - a long-term dream of mine that probably wouldn’t have been possible had I headed into the game industry right away.”

Abe’s language is named “Slag”. “I call it the language of ergonomic programming,” he says. “The focus is on identifying common, repetitive programming patterns and eliminating the need for them.” Abe then used Slag in creating the “Plasmacore” cross-platform 2D game programming framework, which supports Windows, Mac, Linux, iPhone/iPad, Android, and Windows Phone 7. Both Slag and Plasmacore are open source (http://plasmaworks.com/plasmacore) and Abe uses Plasmacore to teach NAU’s Virtual Worlds programming class. “Students pick it up very quickly,” Abe reports. “It’s high-level, general purpose, object-oriented programming with the help of commands that are basically ‘load image’ to load an image and ‘draw image’ to draw that image - and then you just compile it for iPhone, say. It’s simple, easy, and fast.”

While Abe’s game business hasn’t yet become self-sufficient, he says that now is definitely the right time to switch careers. “I’ve had side game projects going since I was in high school, and ever since I graduated college I’ve been waiting for the ‘right time’ to pursue game development exclusively. What I finally realized is that the right time is when I am ready, not when there’s some perfect combination of circumstances. I’ve loved my time at NAU, but now I’m ready to head on down the road.”

REFLECTIONS ON A SABBATICAL PROJECT IN DRESDEN, GERMANY

BY DR. ECK DOERRY

The word is out: Dr. D is back from Germany! Whether this is good or bad news has undoubtedly varied from student to student (!!), but I’m certainly glad to be back. The contrast between NAU, the University of Applied Sciences in Dresden (HTWD), and then (especially) coming back has been a real eye-opener, allowing me to gain a whole new perspective on our higher education system here in the United States.

Before I talk about that, however, I promised to report on how the sabbatical project panned out. As you may recall from the last newsletter, my sabbatical involved an experimental pilot of an “entrepreneurial design” concept, centered around mixing business (marketing, management, and finance) and engineering students in a project-oriented course based on real-world product development. The iCubed project team eventually grew to 31 students spanning 7 distinct disciplines, working to develop a marketable, fully-automated biodiesel production unit for the home or small-business market. The project ended successfully, albeit with some concessions due to lack of time: we ended up having to abort the prototype build-phase, and settle for a lab prototype, full engineering plans for the production unit, and a business plan for pursuing venture capital.

We learned an enormous amount from this pilot, and a paper on the project got strong reviews and will be presented at the ASEE annual conference in June. The German students loved it as well, and actually begged local faculty to find a way to continue the project in the fall semester; our colleagues at the HTWD are working on offering similar courses on their own in the future. Back at NAU, we plan to incorporate the insights we gained into an ongoing discussion with the Franke School of Business to develop an “engineering entrepreneurship” track for advanced students. What a great Capstone experience a project like this would make!

As for contrasts in higher education, I have to say that I’m glad to be back at NAU. I had incredibly committed and competent collaborators and colleagues on the HTWD faculty, but being a professor in Europe is still very much an “old world” experience: in general, everything is more stiff and formal, and professors maintain incredible prerogatives over their teaching (e.g., there are no syllabi), while at the same time teaching many large classes. What this adds up to is a strong divide between professors and students, and even between professors, their colleagues, and college staff. Everyone has their job, the boundaries are rigid, and woe to anyone who doesn’t observe them. The system of evaluation is quite different as well: there is generally no homework, no midterms, nothing but lecture until a final exam that counts for everything not used to this (and complained frequently), but generally liked it, admitting that it required “not so much self-discipline” and allowed them to learn more from the feedback along the way.
In summary, I found the new perspective interesting, but am glad to return to NAU: I missed the relaxed atmosphere, the personal banter with students... and, of course, the sun in Flagstaff!

And while the landscape of technology has changed fundamentally, educators have been slower to adapt to these changes. While several universities have mandated new freshmen have iPads or similar smart mobile devices it’s less clear that we are using the technology effectively. We are moving toward a new era of pervasive computing but are we effectively using these resources to create pervasive education?

In the fall of 2010 Abe Pralle and I began a project sponsored by a Presidential Innovation sub-award from Paul J. to develop an iPad version of our Computer Science I class (CS126). The danger in developing a class like this is to take canned material and just view the platform as a new way to view the content.

Our goal in this work was to create a class that was fundamentally different - utilizing unique features of the iPad and building on the ways that people normally use the device. The result is an iPad based workspace for developing simple code that is different than anything that currently exists for the iPad. Students can compile code, get feedback, do homework and take tests on the device. The “textbook” for the class was designed specifically for the device and incorporates interactive examples, text-to-speech reading and vivid imagery not possible in traditional texts.

CS1 on the iPad utilizes a lot of existing work we have done on JavaGrinder our online CS1 practice system and is forming what we hope is the basis for pedagogical ecosystem of components that can be used to practice the art of computer science. We have recently received funding from Google to explore the extension of this ecosystem into the cloud by using Google App Engine to provide services for managing classes, code, and content used in JavaGrinder and ultimately our iPad platform. Our vision is that the future of computing education isn’t in a lab but that the world around is a lab of pervasive resources that clients for laptops, touch devices, or smart phones can use to deliver content, tools and social support.
**Practice-Oriented Approach to Teaching Undergraduate Data Mining Course**

**By Dr. Dan Li**

Data mining is a fast-growing field of study in Computer Science. Many schools have developed data mining course for undergraduate students. At NAU, data mining is an elective course (CS 445) taken by juniors and seniors in Computer Science (CS) and Applied Computer Science (ACS). The overall course outcome is outlined in the syllabus as “Successful completion of this course will provide a student with the necessary skills to design basic data mining algorithms to solve a variety of real-world applications.” The course content for data mining has been well defined and streamlined because of the availability of outstanding data mining textbooks and the guidance and recommendations provided by the ACM SIGKDD Curriculum Committee. However, the focus on theoretical contents of data mining makes it hard for undergraduate students to digest, and thus, compromises the overall learning outcome. In Fall 2010, we offered data mining course for the third time after we first created this course in Spring 2006. In the first two offerings, we mainly focused on the theoretical contents of data mining with only one team project as a practical experiment. The assessment results were not as good as expected in the first two offerings. Given the applied nature of data mining, we introduced a practice-oriented approach in the third offering to better serve the course outcome. Based on the recommendations from the ACM SIGKDD Curriculum Committee, the objectives of integrating the hands-on components into the theoretical knowledge delivery are five-fold.

1. Learn to use data mining systems by using some data mining software.
2. Implement some data mining functions including association mining, classification, clustering, text mining, and visualization.
3. Implement, refine, and compare of several different data mining methods.
4. Propose, implement, and test new data mining solutions.
5. Use real data to implement and test data mining functions.

By introducing the hands-on components, students have their very first opportunity to deal with real-world data and to carry out experiments that illustrate topics in a realistic setting. The open-source data mining tool, RapidMiner, is introduced to assist students to explore and digest various data mining processes and algorithms. To meet course objectives, we integrate practical experiments into almost every aspect of the core data mining concepts. Each course objective is supported by at least three hands-on experiments. This provides students plenty of opportunities to explore, implement, and digest data mining algorithms and functions. To generate the greatest productivity and self-satisfaction, we allow students in the data mining class to propose, implement, and test data mining solutions for any projects they have strong passion for. The comparison of student scores received on core topics from the last two course offerings shows the dramatic improvement on students’ performance. The D&F rate has dropped from 50% to 21%. With the limited number of samples, we cannot simply conclude that the practice-oriented approach is the mere contributor to this improvement, but without any doubt, the hands-on experiments do help students get a better insight into data mining core concepts.

**Outstanding Faculty Who Care**

The Computer Science faculty has varying professional specialties, but all are united by a strong commitment to teaching. Among the special interests of the faculty are Groupware Systems and Intelligent Interfaces (Doerry), Data Mining (Li), Optical Networks and Security (Wang), Graphics and Computational Geometry (Palmer), Distributed Systems and Web Technologies (Otte), Software Engineering and Robotics (Georgas), Games and Virtual Worlds (Pralle), and introductions to software development and user interfaces (Jacobs).

Please come see us anytime, make an appointment, ask questions, try our interesting classes, sponsor a project, make a donation, join NAU ACM, tell a friend. There a many ways to get involved.

See more about our program at:

[http://www.cefns.nau.edu/Academic/CS](http://www.cefns.nau.edu/Academic/CS)

---

Another photo of Dr. D when he was in Dresden: