Education Updates

6.170 focuses on software design

6.170, also known as Software Studio, is a class that Professor Daniel Jackson built and has been teaching since Fall 2011. Jackson relates the history. "When we designed the new curriculum, we always planned a more advanced course on programming and software engineering." He and Prof. Rob Miller had discussed how 6.005, Elements of Software Construction, had turned out to be more challenging than they had anticipated. "So we decided it [6.005] needed to be reshaped into a more conventional programming course — from the more advanced course I'd originally devised," he explained.

The need for a more advanced course resulted in Jackson's creating the current 6.170 and Miller's reshaping 6.005. Jackson chose the number 6.170 in tribute to the original software engineering course developed by Professors Barbara Liskov and John Guttag in the 1980s. The new 6.170 is very different, with more focus on application design and less on programming, but some of its key features, notably the half-term-long team project, were inspired by the original one.

"I decided to make 6.170 a class focused on design aspects of software," Jackson says, "and chose the web as the platform — since it was already becoming the dominant software platform, and we had no course on building web apps."

Jackson's rationale has paid off. At this point 6.170, only offered in the fall term, is a very popular class now up to 180 students (compared with just 42 three years ago). Besides the fact that students want the practical skill of building a web app, he is excited by the challenge of teaching design, something he says "is underemphasized in general at MIT."

Design of software, however, has its specific issues. Jackson notes that programming courses often skip the most important and hardest part of design: getting from a vague sense of the problem to a specification of what you want to build. In other words, understanding how the software should behave.

In teaching 6.170, Jackson says he's considered several issues. What do you need to think about to cross this gap—from defining the problem to specifying what to build, and, what should be written down to help crystallize and record the ideas (for critique purposes). He has developed a simple method. "It consists of formulating the purposes that you want to achieve," he notes, "and then devising concepts that fulfill these purposes."

EECS senior Josh Haimson says about 6.170 "I liked the focus on design and the practical nature of precisely narrowing and defining a real project of our own design."

From left, EECS graduate students Erica Du, and Jonathan Wang and TA Viikas Velagapudi hash out an app design in the new 6.170.

Haimson appreciates the focus on understanding the problem. "In any engineering setting, it's important to truly understand the problem that you are solving and refine your vision for the simplest way to solve that problem," he says. Haimson is interested in pursuing his interests in AI and entrepreneurship when he graduates in 2016. He wants to work in a company that applies state of the art Artificial Intelligence (AI) and machine learning algorithms to influence industries in a meaningful way.

"From the point of view of alums and companies," Jackson notes, "the most valuable thing about 6.170 is that students are taught how to conceive what the app is about—what it's purpose is and what the concepts are."

6.170 TA and EECS graduate student, Michael Maddox says that one reason the class is very popular is that it covers web programming — something that is extremely useful and applicable, he notes. But he adds that web programming is a difficult area with a lot of moving parts, difficult pitfalls and is generally poorly documented compared with other programming environments.

Another more recent feature in the setup of 6.170, Jakson notes, is the use of a newer (server-side) framework called Node.js, which is based on JavaScript. Using a conceptually simpler and smaller core means that the students don't need to learn a new language such as the previous more well established Ruby on Rails language, which has a lot of hidden "black magic" — therefore requiring a higher learning curve

(next page)

TAs for 6.170 seated left to right: Rebecca Krosnick, Cynthia Jing and Kathryn Siegel; standing left to right: Daniel Jackson, Evan Wang, Vivek Desari, Emily Zhang, Kimberly Toy, Charles Liu, Bryan Collazo, Michael Maddox and Mark Day.

and not allowing a clear vision of what is going on under the hood. "[With Node.js], it's possible to create small pedagogical examples," he says and "...it looks good on a resume as it's perceived as cutting edge." He also adds that it can be installed in a few minutes and tends to be used for more interactive systems.

Data structuring is yet another aspect of design that Jackson originally taught in 6.005 and now incorporates in 6.170. He notes, "Getting from what data you need to how the data is structured requires a simple but potent representation called a data model, which is then transformed into a database structure."

Maddox notes that Prof. Jackson's professional research in data and conceptual modeling has built in a take-home message for 6.170: "The up-front design pays dividends throughout the life-cycle of software," he says. "Through the conceptual and data modeling ideas that are introduced in the course, programmers should in principle be able to save themselves a lot of time and effort later on, and potentially avoid critical and irreversible infrastructure errors altogether."

Charles Liu, 6.170 TA and EECS graduate student explains that this data model is a "snapshot" of what data exists in an application and how the various pieces of data relate to one another. He notes that this is a huge concern not just in 6.170 and web applications, but in any program that deals with creating and updating data fields. "After an app is in use," he says, "changing the structure of the data is much harder than getting it right the first time."

Liu also notes that other prevalent models in computer science are discussed in 6.170 as well, such as "model-view-controller", which governs the relations between the data, the user's actions and the presentation of the data, and also "client-server", which governs the separations and communications between two machines – one the "client" (ie., the browser) and the "server."

EECS Junior Jessica Andersen says "Being a computer scientist isn't only about making things — it's about sharing the things you make." She thinks the general design and communication through design skills that were taught were valuable for her and any computer science student, especially "...the ability to share the things you've created."

From TA and EECS graduate student Kimberly Toy's perspective, the most appealing aspect of 6.170 is a set of learning skills that are directly applicable to industry. "Many of the software giants are consumer web companies, and when hiring, they definitely value the practical experience that students gain through 6.170," she notes, adding that making web applications is fun anyway. "We use our computers and surf the web all the time, so thinking of ideas for new applications can come quite naturally," she says. "The idea of implementing that idea and making it a reality is very exciting and is what 6.170 is all about — empowering you to bring those ideas to life."

Photo left: From left, Pasin Manurangsi, Eric Chang, Minshu Zhan, Weihua James Li, and Prof. Daniel Jackson with TA Hassan Mousaid. Minshu Zhan holds the winning team's ice cream gift certificate.

6.035: Computer Language Engineering

Winning 6.035 team, named 0xD06E, from left: Jenny Ramseyer, Eli Davis, Tom Boning, and Michael Behr with TA Sasa Misailovic.

What it takes to win 6.035, a semester-long project competition

For over a decade, 6.035 has been taught at MIT with the goal of students building a software system from scratch, says Prof. Martin Rinard, who joined Prof. Saman Amarasinghe in 1999 to teach the class. Teams for the term-long project are formed at the beginning of term since building a compiler is a significant engineering effort, Rinard notes. "The end of term competition [known as the compiler derby] is designed to provide a fun and motivating experience for committed students, who welcome a challenge, enjoy competition and strive for excellence."

Over the past few years, the course has moved towards giving the students more choices and freedom with less structure. In previous years, students were required to use Java and given code skeletons to support coding the compiler in this language. Students now start with a clean slate and the freedom to use other languages such as Scala and Haskell, which can be more productive for writing compilers. And, "the class has become more uniformly successful"—to the point that this fall, "...everyone's compiler worked with no outright disasters," Rinard says.

6.035 TA and EECS graduate student Sasa Misailovic notes that building the compiler visits all the major phases of compiler development, from beginning with specification of the programming language to ending with the techniques that make the produced executable fast. "Most of the student compilers have from several thousand to over ten thousand lines of code," he says.

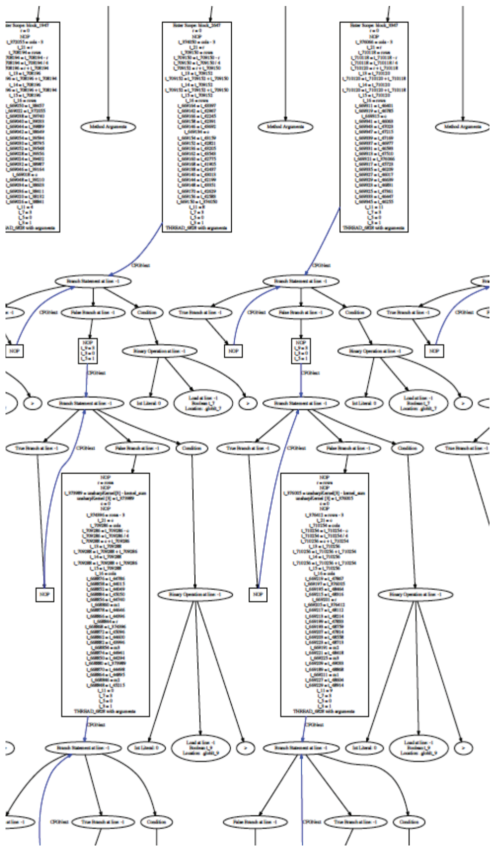
"Unlike most classes in CS, 6.035 is structured around having a semester-long project — actually writing from the ground up your own compiler [a system that takes a program's source code and produces an executable file that can run on a computer's processor]. It's a rare and valuable experience to do it, though you want to take this class with friends you can rely on." says Tom Boning, EECS senior and one of four on the winning team. His role on the team was to focus on the hardware and what was needed to make it listen.

Team member Michael Behr, a senior in Brain and Cognitive Science and MEng student in EECS — the one who took on team manager role plus implementing the team's parallelizing code — explained the class structure: "The first half semester we wrote a compiler that could generate code for a computer to run and in the second half, we improved the compiler to optimize that code so that it could run as quickly as possible."

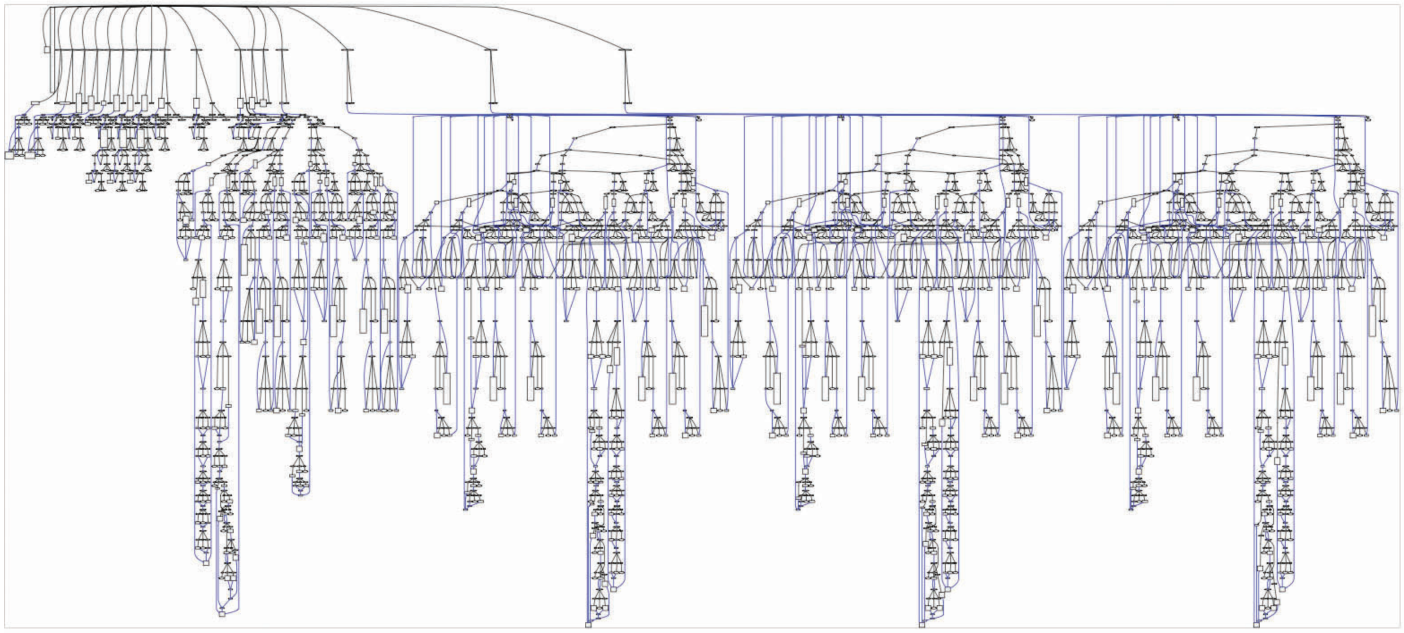
"The group's compiler focused on generating parallel code," he continued, "splitting most of the work into multiple tasks that the computer can run at the

(next page)

Below, a cropped section of the team's parallel output. See the full parallel output diagram next page.



Winning and learning in 6.035, *continued*


The parallel output diagram, courtesy of the members of the the 6.035 winning team Jenny Ramseyer, Michael Behr, Tom Boning and Eli Davis.

same time." The group was able to split most of the work into four tasks, allowing the program to run three and four times as fast. He says "This kind of work is becoming increasingly important as processor manufacturers push their chips to the limits of physics. This parallelization is what is making it possible to take advantage of the past decade's advances in processors."

EECS Senior Jenny Ramseyer another member of the winning team described 6.035 as a mainly lecture-based class about compilers sprinkled with "funny stories from industry". She noted about her team, "We chose the name 0xD06E, in honor of the Internet "doge" phenomenon. It's doge in hexadecimal."

Ramseyer says the toughest times in the class were getting the compiler to output machine code known as codegen and the final round of optimizations. She and fourth team member Eli Davis, also a senior in EECS, pulled two all-nighters in a row for the last round of optimizations. "It was bad. But fun!" Ramseyer says. "This class will take up all of your free time, ...but it worked out. Debugging was definitely the worst. You're looking at these giant messy graphs." [See the code diagram she prepared, above.] Ramseyer, who is looking forward to her MEng for which she has already lined up a project in reconstructing origami tessellations, notes about the class, "You really should know your code inside out, and understand how everything works together. In other classes, it seems more like you're filling in little functions that magically work together to do something, but in 6.035 you write everything."

"The hardest part of the process for me was the end of that first half, when we went from representing a program to actually running it," Behr said. "We spent the first few weeks building representations of computer programs and checking by hand that they looked right, but the processor was much more demanding than we had been." When the team realized how the program needed to run — in comparison with their ideas about how it should look — they discovered their mistakes. Behr noted, "Any students reading this and planning to take the class: be ready for the difficulty to ramp up enormously once you're generating assembly code!"

As Team 0xD06E was reaching a winning conclusion, Boning described the experience. "It's really rewarding to see something you built yourself take shape and actually work. You're not filling in a box inside something someone else built—you're coding the entire thing yourself." He says that he can see applications of this class in reverse engineering, which tries to decompile, or at least figure out the behavior of programs based on compiled binaries. His goal is to head for industry after he completes his degree in June 2015.

Behr, who is aiming for scientific research in the cognitive sciences and neuroscience, admits that it's unlikely he will be writing many compilers again, but the skill sets he has gained include much better understanding of what exactly happens when he writes a computer program — something that he anticipates will be needed in the computationally rich areas of neuroscience. Even more important for him was the experience of being team manager.

"We were thrown into the deep end of the project in a way that I don't expect to happen for most of my career," he notes. "We had no prior experience and no management but what we provided," Behr said. "It was a crash course in a lot of the ways that organizing a project can go wrong: misunderstanding each other's work, leaving functionality unwritten because everyone thought someone else would do it, miss-prioritizing important tasks, and many more failure modes. Fortunately we all kept our heads on and stayed friends by the end of the project!"

Jenny Shen, left, EECS senior and 6.UAT undergraduate TA, describes a word she has drawn from a deck of Taboo cards — developed by a number of TAs, students and 6.UAT creator and EECS Senior Lecturer Tony Eng (center). Anthony Adams '15, second from left, assists his teammate while Benoit Landry (far right), EECS grad student, and Francis Chen '15 attempt to identify the mystery word.

6.UAT gives engineers tools to convey ideas

by Kathryn O'Neill

If you're playing improvisational games or Taboo in class, chances are you're in 6.UAT Oral Communication. This is not your average engineering class—yet instructors and students agree that 6.UAT is invaluable to success in engineering.

"All of us [alumni] looking back think that 6.UAT might have been the most important class in our curriculum," says David Thomas '13, a quantitative researcher for Teza Technologies. "I can't imagine a more quantitative and technical job than the one I have, but a big part of the job is convincing people a project is worth pursuing and getting partners on board."

Designed to teach students to speak confidently and give effective technical presentations, 6.UAT is a required subject for all undergraduates in Electrical Engineering and Computer Science (EECS). "Engineers have great ideas, brilliant ideas, but ... part of being good at what you do is being able to explain what it is that you do to audiences with different backgrounds," says Senior Lecturer Tony Eng, who launched the subject in 2004 and has taught it ever since.

"You can go though MIT and just focus on big algorithms and the best way to write code, but if you didn't learn the communication piece, that would be a shame," says Sean Liu '10, M.Eng. '10, who works as a product manager at Facebook. "So much [of a job] is getting a project approved and communicating results. I'm very thankful that the department folded this class in."

Practice, practice, practice

A one-semester subject that is usually best taken in spring of the sophomore year or during the junior year, 6.UAT typically requires students to attend one large-group meeting and two small-group recitations a week. The class features many opportunities to practice public speaking and centers on three main assignments:

- A short, structured talk of four to five minutes in which students present a technical project they've worked on to a general MIT audience;
- An eight-to-10-minute talk explaining the intuition behind a technical topic to a non-technical audience—namely, a live high school student audience during an outreach event at MIT; and
- A 15-17-minute talk proposing a technical project to an audience of peers.

"The MIT culture puts a great deal of emphasis on coming up with great ideas, and our students (and faculty) do an impressive job of it. Our culture does not, it seems to me, put sufficient value on the ability to communicate those ideas," says Professor Randall Davis, one of a number of EECS faculty members who have taught recitations for the subject. "One of the important things 6.UAT does is try to change this, teaching our students how to be effective communicators and showing them the value in that skill."

Eng agrees. "Students think, 'Why do I have to take a class on talking? I've been talking since I was 3,'" he says. "I hope they discover there are all these tools and ideas they can use."

For example, Eng says he incorporates improvisation to help students gain a level of comfort dealing with the unexpected and uses a form of the game Taboo to train students to avoid jargon. (In Taboo, players have to describe a word while avoiding certain banned terms; Eng has created his own cards featuring technical terms and a list of outlawed jargon.) Students in 6.UAT also gain experience designing technical presentations, presenting to different audiences, and giving and receiving constructive feedback.

6.UAT students give a three- minute presentation for their final class in 26-100. (photo, above) Earlier in the term, 6.UAT students give teaching lectures to a group of high school students. (photo, page top)

Engineering a presentation

Liu says he was pleasantly surprised to discover he could improve his speaking skills though 6.UAT, because he had always been fearful of communicating to audiences. "You can think of it almost as an engineering problem," he says, noting that key talking points can be blocked out and developed piece by piece. "It becomes almost a formula where I can solve each block independently and construct the presentation."

Students say that it's helpful that Eng provides a great role model for the class—illustrating in every lecture that it is possible to be both a skilled engineer (Eng has five technical degrees from MIT, including a PhD in computer science) and a compelling speaker. "He tells these magical stories and tries to convey the story behind each lesson. It's pretty cool and inspiring," Liu says. "He is like the Jedi master of public speaking," Thomas says, adding that Eng finds a seemingly endless number of ways to make 6.UAT entertaining. "You don't know what to expect, but it'll be fun and different from other engineering courses," Thomas says.

Along the way, students hopefully change their views about oral communication, Eng says. "When a student approaches presenting, they think of it as a hurdle to get across to get the grade. In reality, it's more about the audience and getting a message across," he says.

That's a lesson that Liu remembers well. "Even something technical and dry if presented with the right examples and analogies can tell the story of what the project was trying to do," Liu says. "That was a big aha moment."

Extraordinary Educators in EECS

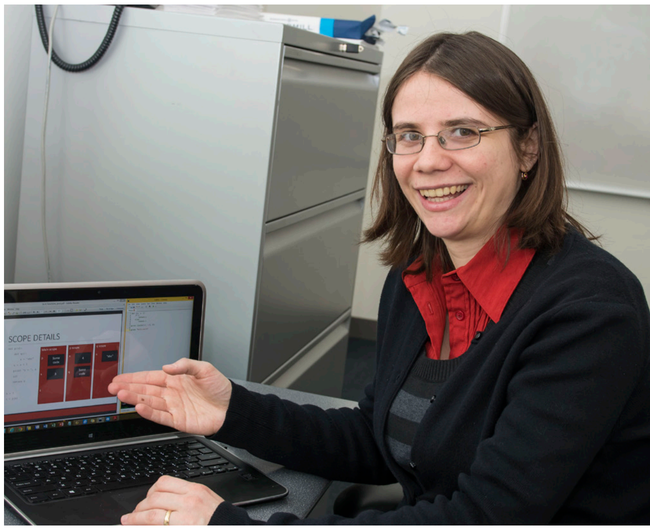
Managing the enrollment overload in EECS

With the expectation that in September 2014 there would be more than 6000 occupied seats in EECS classes, 4000 of which would be occupied by undergraduates, the EECS Department has taken on some extraordinary help.

Three years ago, the EECS Department began experimenting with adding longer-term lecturer support. The department focused on finding lecturers who could be faculty collaborators in resource intensive classes. As of fall 2014, EECS has had six such lecturers, each with a three-year contract, and each an excellent classroom instructor and educational innovator who also has a strong record of collaborating with faculty on course development and delivery.

These "Extraordinary Educators" in EECS (EE-EECS) include Adam Hartz (6.01), Katrina LaCurts (6.033 and 6.02), Joe Steinmeyer (6.01), Max Goldman (6.005), Ana Bell (6.00 and 6.00x), and Silvina Hanono Wachman (6.004 and 6.004x). Our EE-EECS lecturers are dedicated award-winning technical educators, and creative curriculum developers and leaders. They co-lead classes during the term, and spend summers collaborating with faculty on education-related research. And even though the EE-EECS program is in its infancy, it has already dramatically improved both student evaluations and faculty quality of life.

Read about these six Extraordinary Educators below. One thing they all have in common is a love for teaching.



When **Ana Bell** graduated with her PhD from Princeton in 2013, she was hired by MIT EECS to be a lecturer for 6.00.1x , the first offering of a split 6.00x. While at Princeton, Ana was a TA in undergrad for one year, teaching intro computer science to freshman and during the summers, she taught the basics of computer science to high school students. "I enjoy introducing computer science to students who have never programmed before. It's satisfying to see students excited to see their first program run for the first time," she says. Ana plans to continue teaching computer science and hopes to continue exposing students to the basics of computer science and programming.

She has noticed during her tenure over the past almost two years that with MOOCs gaining traction, the department is transitioning to have the course be more interactive and hopefully more efficient at teaching computer science. "We used MITx to provide students with practice programming exercises and held exams online rather than on paper to better simulate the process of programming that students are used to from problem sets," she notes.

Max Goldman has been at MIT a while — earning his BS in Course 18C (Math with Computer Science) and his PhD studying with Prof. Rob Miller in Human Computer Interaction in CSAIL in 2012. He became excited about teaching through a number of experiences. First, he says he has Prof. (now emeritus) Paul Penfield to thank for his first teaching opportunity in Course VI when he was an undergraduate assistant for Information and Entropy (then 6.095 and now 6.050) in 2002. His research interest in better ways to learn and teach software engineering grew out of his experience with MEET, the educational initiative aimed at building a common professional language between young Israeli and Palestinian leaders. He taught MEET students, designing and building the program's CS curriculum and managing teams of MIT instructors. As an extraordinary educator, Max is teaching 6.005, a very popular class — the 10th largest at MIT, according to *The Tech*. He finds it very rewarding to work with so many students. "I think the ideas and tools in 6.005 are absolutely necessary for anyone who wants to write larger programs, or collaborate with others, or anyone who simply wants their code to work reliably." He says that the class has been reorganized in favor of active learning with as much time as possible devoted to working problems and writing code. He loved crafting fun lectures, but enjoys working on the new challenges of an active learning classroom.

Extraordinary Educators, *continued*

Adam Hartz knows MIT EECS well, having earned his SB (6-3) in 2011 and his MEng also in Electrical Engineering and Computer Science in 2012. He has been an EECS extraordinary educator for just about three years. How did he prepare? During his undergrad and MEng years, he worked as a lab assistant and teaching assistant for 6.01— with some brief experience with 6.02 and 6.003 as well. He started as a lab assistant for 6.01 in the Fall 2008 term and has been working his way up through the ranks for a while. "By the time I was hired as a lecturer, I had a lot of experience under my belt, ...so I was ready to jump in," he says. Working mostly with freshmen and sophomores, Adam finds it exciting to be a part of their academic transition from high school students to MIT students. "In particular," he notes, "6.01 has a strong engineering focus, and for many students, it is the first time they engage with an authentic engineering problem; it is very gratifying to help students make the transition from solving to creating, from analysis to design." He also finds it gratifying to work on course development, on improving teaching and learning on a broader scale. "I can think of nothing I would rather be doing than teaching," Adam says, "and I hope to continue doing so in the long term." He says it's an added bonus to be able "to work with so many wonderful people who all care about teaching and learning just as much as I do."

Katrina LaCurts, SM '10, PhD '14, the daughter of two teaching parents, coached high school programming teams while she was in college, and in graduate school (in EECS at MIT) she TAed for 6.02 and taught recitations for 6.02 and 6.033. She also spent two summers as the discrete math instructor for the Women's Technology Program (WTP) for which she designed her own curriculum and managed her own staff. She has also taken classes with the Teaching and Learning Lab at MIT and keeps track of the literature in higher education. The most rewarding aspect of her teaching as an extraordinary lecturer she says is seeing the students get excited about the material—especially if they were initially unenthusiastic. "My favorite course reviews to receive are ones along lines of 'This class was way more interesting than I thought it would be!'" she says. "There are so many amazing things in the world, especially in science and engineering, and it's easy to lose sight of that." she adds. "So I try to instill that knowledge in my students, and it's very rewarding when I am successful."

Katrina is hoping to remain in teaching. In fact, ideally, she'd like to remain in a similar position for the rest of her career.

Joe Steinmeyer's first taste of teaching was as an undergrad TA at the University of Michigan, and he liked it. Then at MIT as a TA and grad instructor in EECS classes for the last four years of his graduate work, he got to know how things work in the department. In addition, he has worked for the last six years in some of the Office of Engineering Outreach Programs (OEOP) summer and school-year programs teaching really smart high school students. In some cases, he has enjoyed working with them again in Course 6 classes as they have ended up coming to MIT. "I think just seeing students *get* concepts is rewarding," he says. "By that I mean I'm a big nerd and really love EECS material and when somebody else understands something and then proceeds to 'nerd out' about it... I think that is really fun and rewarding," he says.

Joe finds the extraordinary educator position is really helpful towards his goal of teaching long-term. He enjoys the freedom in curriculum development and experimentation. "MIT isn't the sort of place that wants or encourages people to teach stale curricula," he notes, "so having both the feedom and encouragement to experiment with teaching techniques and content is really nice." He is excited by the way that MIT and EECS are embracing the MOOC movement and by the fact that education is a dynamic field. "So it's neat to be in this environment," he says.

As an undergraduate, **Silvina Hanono Wachman** studied electrical engineering with a focus on computer engineering at Cornell University. Then she came to MIT for her master's and PhD, focused on programmable hardware and code generation that could be retargeted for different VLIW architectures. As a graduate student she TAed 6.004, the class she is teaching now. After a 12-year stint working in industry, she decided she wanted to return to teaching and came back to MIT — teaching 6.004 again. This has led to her being hired as a Lecturer for the residential MIT course as well as preparing for the 6.004x version of the course for edX. "I love explaining difficult concepts to students in a way that simplifies the material and helps them understand it," she says. She hopes to continue teaching at MIT as well as on edX courses. In addition to working with 6.004, she looks forward to expanding her teaching repertoire with other classes as well. She notes that the development of the edX version of 6.004 presents many challenges. While the material taught is pretty much the same," she says, "adapting it all to work on edX and be able to self assess is a daunting task for the amount of material involved. Nevertheless, she says it is extremely rewarding to be involved in the process during its infancy.

Taking the EECS Tour

Since 2000, the EECS Undergraduate Office has trained its students to lead tours

Anne Hunter remembers when the EECS Department started doing tours 20 years ago. She was the tour guide. Five years later, the tours became popular enough that Anne offered to "meet and greet" and students were hired to run the tours. She says it is not so much an admissions kind of thing. "Having a bunch of smart people come learn about all the great things that go on here is the best kind of PR," she says. In fact, several of her student tour guides were originally wowed by the department when they went on a tour years before.

The tours are definitely seasonal, Anne says, with very large groups coming in spring and summer. She says that big groups are good because then there is a chance for the prospective students in the group to meet and ask their down-to-earth questions like "What is it like to be a student here?"

Tour guide Alex Hsu, '13 and currently an MEng student in the department, remembers large tours where big groups take longer to move from one spot to the next and younger ones in the group actually become friends exchanging emails.

"As I leave Anne's office," Alex says about her tours, "I usually introduce myself and try to find out what people are most interested in (whether it is EE or CS or they have no idea). I also explain the differences between 6-1, 6-2, 6-3 and 6-7 and discuss the MEng program. I usually also try to mention that at MIT we speak in numbers so if I say a number that doesn't make sense, people should definitely ask because chances are it is MIT speak."

Anne Hunter surrounded by her EECS undergraduate tour guides, spring 2012. Seated from left: Dylan Sherry (6-2, '12), Kevin Fischer (6-1, '12), standing from left: Louis Lania (6-2, '14), Irena Huang (6-1, '11), Anne Hunter, and Elaina Chai (6-1, '12).

On the EECS Tour, *continued*

Cody Coleman, '13, MEng '15, describes his EECS experiences on a tour. Tour guides Ben Greenberg, MEng '15, on his right and Alex Hsu, '13, 'MEng '15, center join the listening guests.

The first stop on the tour is the **6.01 lab**. "I like to show people one of the robots and discuss how hands-on the class is, giving students a chance to get real experience with coding and with building circuits. This helps people figure out which track they are the most interested in." This also gives her a chance to talk about class requirements and sizes.

Next Alex moves the group on to the **6.02/6.002 lab space**, where she shows off the tabletop MRI machines. "A lot of times people (especially parents) are concerned about degree programs and not having time to take extra classes," Alex says. So she makes sure to explain that MIT students have wiggle room in most majors and even double majors are possible. At the 6.02 end of the lab she also likes to talk about 6.02 check-offs with TAs, where a TA will go over each student's code individually.

In the lab she also mentions Athena clusters (and usually gets asked about macs vs. pcs, so she explains that students use our his or her own operating system. Then she typically moves down to the **6.002** end of the lab where she shows off some of the EE equipment and talks about how this is one example of an elective class that isn't required by everyone in the Course 6 major. Here she also discusses how this class is very theoretical, but also has a fun lab component, which takes less time than 6.01 lab per week. She describes how the class builds throughout the semester, how students learn how to make various components, like a digital to analog converter, and an amplifier circuit, and ultimately an mp3 player — with one still conveniently in the lab .

Next stop is the (new) **Engineering and Design Studio**. Sometimes the lab techs talk while the tour is there, but if not, Alex points out the various machines that students have access to and shows off some of the tools and projects that are on display. Next they walk to CSAIL, Alex usually stops by the elevators in building 36 to talk about UROPs and how that is an amazing opportunity for MIT undergrads. Then as they walk through **RLE** she mentions that RLE is the biggest EE lab on campus and points to the informational wall art. At this point, she also talks about SuperUROPs and how they are a Course 6 creation and an unparalleled opportunity. She also talks about how a lot of SuperUROP projects turn into MEng theses for students, allowing them to have a more substantial research project.

Next in **CSAIL** outside of the **PR2 robots lab**, Alex shows the robot and talks about the research being done that is easy for them to visualize. She tells them about the goal of getting robots into homes, but how a "simple" task like "do the laundry" varies from home to home, so the robot has to learn how to learn. From there they go down to the first floor, where by far one of the most popular stops is the mechanical calculator (the **DIGI-COMP II**) that can do bitwise math using pool balls as the 'bits'. "People are fascinated by this!" Alex says. "I once had a group that stayed here for 15 minutes ...they thought this was the most interesting part of the tour." From here they see 32-123 if it is open and talk about MITx and then head back to chat more with Anne Hunter in the EECS Undergraduate Office.

Cody Coleman, '13, MEng '15 has enjoyed giving tours since 2013. He says: "Between my bachelors and masters degree in the EECS department, I have traveled to over 15 different countries, worked at Google, and done research to define the future of learning, accomplishing more than I had ever imagined. What is even more impressive is that I am not an anomaly but the norm. There is a strong network and support system, making MIT a place where you can have an impact and make the world a better place. As a tour guide, I get to impart that to people all over the country and the world. "