Research

Laws, regulations, and organizational policies codify societal values that challenge software engineers building regulated systems. My research examines how software engineers can ensure software requirements comply with relevant privacy and security regulations, laws, and policies. Transparently complying with laws and regulations creates an opportunity to build trust between end-users and the software systems that manage their information; and an ethical responsibility for any software engineering professional. Furthermore, governments are increasingly regulating software systems due, in large part, to privacy and security concerns. Methods, tools, and techniques for evaluating, establishing, or demonstrating legal compliance in software systems will only become more important for the foreseeable future. This relatively young area of research is known as Regulatory Compliance Software Engineering (RCSE).

I am seeking an academic position at an institution that will enable me to continue to advance the science of RCSE. My research shows that software engineers are currently ill-equipped to reason about legal compliance and suggests that software engineering methods, tools, and techniques can mitigate this deficiency. Moreover, I look forward to educating and inspiring my future students to consciously focus on regulatory compliance as a professional and ethical responsibility. By discussing research focused on real-world problems in the classroom students can be engaged in and inspired by scientific discovery and foster a desire to engineer new products that improve the world. I believe an academic position is a natural fit for my personality, my creative and analytical mind, and my sincere desire to improve the lives of others.

Dissertation Research

In his 1986 talk, entitled "You and Your Research," Richard Hamming posits that great scientists should ask themselves three guiding questions about their research:

1. What are the most important problems in your field?
2. Are you working on one of them?
3. Why not?

Hamming's essay greatly affected my decision to return to graduate school and focus my career on truly important problems. Legal compliance in software systems is one of the most important problems in the field of software engineering. The main objective of my work is to ensure that software engineers can confidently build systems that comply with law. In my dissertation, I developed and empirically validated: (a) techniques for determining which software requirements are legally implementation ready (LIR); (b) metrics to estimate which software requirements are LIR; and (c) tool support for identifying LIR software requirements using legal requirements metrics.

My method for evaluating existing software requirements for legal compliance takes as an input a legal text and a set of requirements that must comply with that legal text. It produces as an output a set of documented legal compliance concerns that should be addressed to ensure the system meets its legal obligations. It also produces a traceability mapping of requirements to specific subsections within the legal text, which is then used by the legal requirements metrics I've developed to estimate which requirements are LIR. This traceability mapping also improves organizational ability to demonstrate due diligence.

These LIR metrics use attributes of a legal text, such as the percentage of sections within a legal text that are traceable to some requirement, to estimate which requirements are LIR. These metrics were validated in a case study against expert legal opinions and analyzed statistically using the Goal / Question / Metric paradigm. I also developed a legal requirements management tool, called Red Wolf, to support the use of legal requirements metrics as a part of the software engineering process. Legal requirements metrics are a promising approach to aiding software engineers as they make legal compliance decisions.

Ethics and Privacy Research

Software engineering professionals must operate ethically and respect societal values like privacy, particularly since software is becoming ubiquitous. As a Wilkinson Research Ethics Fellow, I examined the ethics of behavioral advertising. Behavioral advertising is a method for targeting advertisements to individuals based on behavior profiles, which are created by tracking user behavior over a period of time. Individually targeted advertising can significantly improve the effectiveness of advertising, but it may have serious implications for civil liberties, including privacy. My research examined the ethics of behavioral advertising in the context of recent advances in technology, changes in the political and legal climate, and the history of advertising. Understanding this context helps engineers develop successful technologies in a competitive marketplace and helps lawmakers specify clear, technology-aware laws to protect citizens. My interest in ethical software engineering is also evidenced by my focus on privacy- and security-related federal regulations that govern software systems.

Privacy is at the leading edge of regulation for software systems. Promises organizations make in their public privacy policies can be enforced by law in most jurisdictions around the world. As a part of The Privacy Place research group, I redesigned and implemented a tool to support the analysis of privacy policies called the Privacy Goals Management Tool. Researchers can use this tool to extract high-level privacy goals as stated in online privacy policies, classify those goals, and ensure that these goals are accurately represented in the software system.

Posdoctoral Research

As a Postdoctoral Fellow in the School of Interactive Computing at Georgia Tech, I helped found a RCSE research group. I also continued research begun as a PhD student, including a project on integrating regulatory compliance into acceptance testing that won the Best Paper award at Agile2013. I also wrote a book chapter (to be published in 2014) with Dr. Travis Breaux for the International Association of Privacy Professionals (IAPP) detailing how information technology professionals prevent undue interference with user privacy. This was an extension of my prior work in engineering ethics and privacy. The main thrust of my research at Georgia Tech was an investigation of natural language processing techniques for improving RCSE.

Legal texts are structured, hierarchical documents that lend themselves well to analysis using natural language processing (NLP) techniques. For example, topic modeling is a text mining technique that can discover the themes in massive document collections. I applied topic modeling to the problem of discovering privacy requirements in a collection of over 2,000 policy documents and found that topic models can indicate whether a document contains software requirements expressed as privacy protections or vulnerabilities. Partially automating the analysis of legal texts through topic models and other NLP tools may prove to be essential for both regulators and software engineers building systems that must comply with numerous laws and regulations with Dr. Jacob Eisenstein.

Teaching

One of the most important lessons I learned in my time at NC State and Georgia Tech is an appreciation for excellence in teaching. Standing in front of a classroom and professing to be an expert capable of preparing students to be professionals is an immense responsibility. It is not easy, but it is clearly one of the most important tasks our society faces today. In fact, as an educator, I believe education is one of the most powerful forces to impact positive change in society and the world at large.

Educators are responsible for preparing students for success in whatever field they may choose to follow. Prior to attending graduate school, I was a full-time software engineer for Advanced Micro Devices (AMD) working on MIPS-based embedded Linux systems. That experience taught me crucial lessons of life as a professional software engineer that I endeavor to pass along to my students. I believe my collective experiences to date as a practitioner, researcher, and public policy expert uniquely position me to provide guidance based on personal experience about any potential future a computer science student may wish to pursue.

At Georgia Tech, I was privileged to participate in faculty planning meetings leading up to the launch of their new Online Masters in Computer Science degree program in partnership with Udacity and AT&T. These discussions forced me to think deeply about many fundamental questions for academic institutions in the 21st century, including questions about the role should Massive Open Online Courses (MOOCs) play in higher education. Clearly, students benefit from in-person instruction, but they may also benefit from judicious use of MOOC technologies. Academics should not dismiss these technologies out of hand, and I look forward to the challenges of incorporating the best pedagogical techniques available in my future classes.

I am confident that I could teach any course in an undergraduate computer science curriculum. However, I have direct experience and interest in teaching courses related to software engineering, computer security, digital privacy, and requirements engineering. Moreover, my work experience in systems programming qualifies me to teach courses on open source software development, embedded systems development, and operating systems.

I strongly believe in incorporating and highlighting the following strategies in my teaching:

Active Learning. Students must be engaged in hands-on, active exercises while in the classroom. These exercises are critical to ensuring that students retain the information presented and begin to see how it might be applied to their homework, projects, and ultimately to solve real-world problems.

Team-based projects. Computer science in general, and software engineering in particular, are team-based endeavors. Students must learn how to work in groups to build something that simply cannot be built by one person. In addition, collaborative projects often result in students learning more from one another than they could ever learn from any individual instructor.

Real-world relevance. Students can see the motivation to learn material that is clearly centered on real-world engineering challenges. People connect to material when they can see how it impacts the lives of others. This is why I choose relevant, real-world project domains, such as healthcare, mobile computing, or social networking, when designing in-class examples, homework material, and projects.

Feedback. Students learn best when they receive thoughtful feedback in a timely fashion on the work that they have done. When teaching CSC 216—*Programming Concepts in Java*, I worked with my TA to ensure that we provided grades as quickly as possible, particularly for the weekly quizzes. Although the quizzes were not a large portion of the grade for the course, they were a weekly signpost to the students, the TA, and myself regarding the areas in which students were excelling and those in which they were struggling.

Reflection. Numerous important subjects fall through the cracks of the curriculum at almost all universities. As a result, I start each class with a short, interesting topic students may wish to explore further. The purpose of this is to inform students about an important part of computer science or software engineering that they may be absent from the curriculum and encouraging them to explore it further. Examples of these topics from my CSC 216 class include everything from high-level discussions on the differences between IDEs and programmatic text editors to the rationale for using version control software.

Clarity and Humor. I have been lucky to learn from an expert in clear communication punctuated with humor to ensure that ideas "sticks" with students. My advisor excels at using humor to clarify a difficult point. This is particularly powerful for those broader points about software engineering and society with which many college students struggle.

As a student at North Carolina State University, I sought teaching opportunities to prepare myself to become a professional educator. I served as a teaching assistant on three separate occasions, and on one of those occasions, I was directly involved with designing a new graduate-level course on Privacy Technology, Policy, and Law—a new and important area of study for graduate students focusing on software engineering. I was also invited as a guest lecturer on more than 10 separate occasions for courses ranging from an introductory undergraduate course on Java programming to several graduate courses on software engineering. Lectures included the following topics: an Introduction to Software Engineering, Software Processes, Software Planning, Risk Management, Project Scheduling, Project Estimation, Requirements Engineering, and Formal Methods.

During the summer of 2011, I taught CSC 216 – Programming Concepts in Java as the instructor of record. CSC 216 is the second course in computing, intended for majors and students in the Computer Science Certificate Program. Emphasis is placed on encapsulation; methods and types; polymorphism; inheritance; interfaces; testing strategies; linked structures; and specification and implementation of finite-state machines. As the instructor of record, my responsibilities covered the breadth and depth of teaching: hiring, managing, and mentoring a teaching assistant; creating, implementing, and grading compelling and challenging course quizzes, projects, and exams; curriculum planning; lecturing; and assigning course grades.

At Georgia Tech, I co-taught a mixed undergraduate- and graduate-level course titled Privacy Technology, Policy, and Law. This course examines challenge of constructing and complying with privacy policies, technical requirements, and legislation. Students in the course participated in extensive in-class debates of privacy challenges, including the ethical and legal foundations of privacy, development of privacy-related technologies, and the comparative differences between American and European approaches to privacy legislation. Course topics included social networks, surveillance, wiretapping, encryption, and web-based advertising. This course also featured semester-long, team-based projects with an Atlanta-based healthcare company seeking to engage students in real-world privacy challenges including: compliance with federal privacy regulations, assessment of security and privacy policies, and development of new, privacy-sensitive approaches to business analytics. By the end of the semester, each team created a formal privacy impact analysis report and presented their findings in person to executives from the company.

Georgia Tech also afforded me the opportunity to mentor several students as they started a career in research. I supervised two independent study graduate students, both of whom completed semester-long research projects in requirements engineering. I also recruited and supervised an incoming PhD student interested in regulatory compliance software engineering.

I approach my responsibilities as an educator humbly recognizing that my knowledge has the power to dramatically improve the futures not only of the students that I teach, but also their families and by extension, our society as a whole. This is not a responsibility to be taken lightly. Educators must embrace it. They must recognize that they cannot blame earnest students for failing to learn in the same way authors cannot blame their readers for failing to grasp their message—that responsibility belongs to the communicator alone.