Software Engineering Courses

I. Engineering Methods

EE 382C Introduction to Software Engineering Processes
This course is an introduction to software engineering with an emphasis on the methods, techniques and technology to build and evolve software systems. The emphasis is on software engineering principles.

EE 379K Requirements Engineering: Acquisition and Modeling
This course will address theoretical and practical methods for acquiring and modeling requirements for various systems stakeholders. Topics will include methods and techniques for managing the acquisition process among distributed team members and distributed stakeholders, eliciting and verifying requirements as a function of the type of stakeholder, the types of requirements, and system development maturity, managing the requirements artifacts, constructing model-based representations of requirements, synthesizing requirements for various stakeholders, and analyzing and evolving model-based requirements.

EE 382C Software Architectures
The course will teach students about software architectures, architectural model specification techniques and analysis techniques offered by the research community as well as those architectures, model specifications and analytical methods commonly used in industry.

EE 382C.3 Verification and Validation
This course covers various traditional and state-of-the-art techniques for software validation, a process that includes reasoning about (the correctness of) programs and testing programs. The course content will include both techniques for dynamic analysis, such as glass-box and black-box testing, equivalence partitioning, test strategy and automation, regression testing and debugging, and techniques for static analysis, such as symbolic execution, and also techniques for software model checking including those that employ artificial intelligence based heuristics. (3 credit hours)

EE 382C Empirical Studies in Software Engineering
Software Engineering is of necessity an empirical enterprise. This course looks at different kinds of empirical evaluation from related fields such as behavioral sciences to see how one can adapt those techniques to the service of software engineering.

EE 382V: Formal Methods in Distributed Systems
This course gives an introduction to the use of formal methods within the software design process. Specifically, this class will cover the application of models to distributed and concurrent systems. Modern software systems are commonly highly distributed, and this added sophistication further complicates software design. The rigor offered by formal methods aims to make the process more precise.

EE 382V: Software Engineering Lifecycle
This course addresses five major topics in the Software Engineering Lifecycle: requirements, design, construction, measurement & evaluation, and maintenance & evolution.

EE 382V Advanced Programming Tools
Programming is difficult - some of the problem developers face include
1. How can a project be structured so that developers can work on it concurrently?
2. How can the building of a project be automated?
3. How can a program be written to make it portable?
4. How can a program be prototyped efficiently?
5. How can a program be tested and debugged efficiently?
6. How can the performance of a program be increased?
Using the right tools can solve these problems. Examples include tools for version control, documentation, program building and configuration, automatic testing, program analysis, and integrated development.

Our approach will be to introduce a specific problem, show how a tool can solve the problem, and then develop the technical principles underlying the tool. We will have written homework problems as well as coding exercises for each concept. The class will have a major design project that will begin at the start of the term. Use of the tools will be a required part of the project. We will use open-source tools to illustrate these concepts. The specific tool stack is described in the lectures section of this document. I selected these tools based on my experience at Google; they also power many state-of-the-art commercial projects.

II. Software System Technology:

EE 382v Data Engineering
Course Description – Data Engineering is concerned with the role of data in the design, development, management, and utilization of complex computing/information systems. Issues of interest include database design; meta knowledge of the data and its processing; languages to describe data, define access, and manipulate databases; strategies and mechanisms for data access, security, and integrity control. (3 credit hours)

EE 382V – Mobile Computing
As mobile computing devices like laptops, PDAs, cellular phones, and even miniature sensors become increasingly pervasive, the demand for applications for this novel environment escalates. This course explores the effects of mobile computing on software design and development. The approach taken uses current research projects in the field of mobile computing to highlight the key aspects that complicate software engineering. We will focus on these concerns in the context of application development. (3 credit hours)

EE 380L Data Mining
Basic concepts of data mining, in parallel with a practical track involving hands-on experience with industrial strength software and a term project will be covered.

EE 382N Communication Networks: Tech/Arch/Protocol
This is an introductory course in Computer Networking. It covers all basic components of modern networks, including: link level technologies such as Ethernet, token rings, and wireless Ethernet; switching technologies such as bridges and ATM; internetworking including IP; the transport layer, including TCP and RPC; and congestion control. Time permitting we will also consider security, quality of service, high-performance networks, and/or multimedia. Although IP and TCP are primary examples used in the course, it is NOT a course on TCP/IP!

EE 382N Distributed Systems
Programming experience, Graduate standing. This will be an introductory graduate level course in distributed systems. It will expose students to theoretical as well as practical aspects of designing such systems. The course assumes that the student has some familiarity with programming. There is no final exam but there will be two exams during the course.

EE382N.4 - Advanced Embedded Microcontroller Systems
Hardware and software design of microcontroller systems; applications, including communication systems; object-oriented and operating systems approaches to interfacing and resource management. (3 credit hours)

EE 382C.16 Distributed Information System Security
Intended to acquaint the student with the analysis and engineering techniques employed in securing today’s networked information system environment. Emphasis is placed on examination of practical security threats, exposures in distributed systems and the technology that is being applied and developed as countermeasures.

EE 382C.8 Methodologies for Hardware/Software Co-design
Techniques used to design complex hardware/software systems; emphasis on specification, modeling, estimation, partitioning, verification/validation, and synthesis.
EE 382C Database Systems
After taking this course students should understand basic database theory: Models/Representations/Views, Relational calculus/SQL, Normalization, and Indexing/transactions/concurrency/recovery; Know fundamentals of one DBMS; Be prepared to learn any DBMS; Be ready to be a programmer or sophisticated user

EE382C: Dependable Real-Time Systems
The spectrum of modeling techniques from synchronous to asynchronous, with emphasis on in-between models such as the timed asynchronous model of Flaviu Cristian’s and the related real-time assumptions. Control structures such as timed state-transition systems, constraints in temporal and real-time logics. Analysis techniques such as model checking of timed systems, extended Presburger arithmetic. Basic building blocks and protocols such as clock synchronization, synchronous atomic broadcast, time-bounded membership protocols, basic real-time scheduling theory, recovery methods by state rollback and others.

Practical implementation issues such as special operating system data structures and algorithms and open system design constraints will be considered

III. Leadership Skills:
EE 382C System Engineering Program Management and Evaluation
Management, engineering, and evaluation approaches applicable to a spectrum of software development programs is taught. General guidelines, metrics, program artifacts, and processes will be discussed in conjunction with case studies.

V. Summer Courses

EE 382V Software Engineering Lifecycle
This overview course addresses five main topics in the Software Engineering Lifecycle: Requirements, Design, Construction, Measurement & Evaluation, and Maintenance & Evolution.

EE 382V Software System Measurement and Metrics
This course uncovers what software metrics are, why they are used, who should develop the metrics, when they should be applied, who should own the measurements, and what should be done with them. It explores the current definitions of useful metrics related to software engineering. It provides (i) the theoretical definition of measure and its application to software engineering; (ii) a comprehensive review of the state of the art in measurement for software engineering; (iii) guidelines on establishing a measurement program; (iv) cost-benefit analyses for selected industrial case studies of software measurement programs.

IV. Master’s Report

EE 398R – Master’s Report
Completion of report in the last semester enrolled in the program to fulfill the requirement for the master’s degree. Offered on the credit/no credit basis only. Prerequisite: Graduate standing in electrical engineering and consent of the graduate adviser. (3 credit hours)