School of Computer Science Courses

Note on Course Numbers
Each Carnegie Mellon course number begins with a two-digit prefix which designates the department offering the course (76-xxx courses are offered by the Department of English, etc.). Although each department maintains its own course numbering practices, typically the first digit after the prefix indicates the class level: xx-1xx courses are freshmen-level, xx-2xx courses are sophomore level, etc. xx-6xx courses may be either undergraduate senior-level or graduate-level, depending on the department. xx-7xx courses and higher are graduate-level. Please consult the Schedule of Classes (https://enr-apps.as.cmu.edu/open/SOC/SOCServlet) each semester for course offerings and for any necessary pre-requisites or co-requisites.

Computational Biology Courses

02-201 Programming for Scientists
Fall and Summer: 12 units
Provides a practical introduction to programming for students with little or no prior programming experience. Extensive programming assignments will illustrate programming concepts, languages, and tools. Programming assignments will be based on analytical tasks that might be faced by scientists and will typically include parsing, statistical analysis, simulation, and optimization. Principles of good software engineering will also be stressed. Most programming assignments will be done in the Go programming language, an industry-supported, modern programming language, the syntax of which will be covered in depth. Several other assignments will be given in other programming languages such as Python and Java to highlight the commonalities and differences between languages. No prior programming experience is assumed, and no biology background is needed. Analytical skills and mathematical maturity are required. Course not open to CS majors.

02-223 Personalized Medicine: Understanding Your Own Genome
Fall: 9 units
Do you want to know how to discover the tendencies hidden in your genome? Since the first draft of a human genome sequence became available about a decade ago, the cost of genome sequencing has decreased dramatically. Personal genome sequencing will likely become a routine part of medical exams for patients for prognostic and diagnostic purposes. Personal genome information will also play an increasing role in lifestyle choices, as people take into account their own genetic tendencies. Commercial services such as 23andMe have already taken first steps in this direction. Computational methods for mining large-scale genome data are being developed to unravel the genetic basis of diseases and assist doctors in clinics. This course introduces students to biological, computational, and ethical issues concerning use of personal genome information in health maintenance, medical practice, biomedical research, and policymaking. We focus on practical issues, using individual genome sequences (such as that of Nobel prize winner James Watson) and other population-level genome data. Without requiring any background in biology or CS, we begin with an overview of topics from genetics, molecular biology, stats, and machine learning relevant to the modern personal genome era. We then cover scientific issues such as how to discover your genetic ancestry and how to learn from genomes about migration and evolution of human populations. We discuss medical aspects such as how to predict whether you will develop diseases such as diabetes based on your own genome, how to discover disease-causing genetic mutations, and how genetic information can be used to recommend clinical treatments.

02-250 Introduction to Computational Biology
Spring: 12 units
This 12-unit class provides a general introduction to computational tools for biology. The course is divided into two modules, which may be taken individually as courses 02-251 and 02-252. Module 1 covers computational molecular biology/genomics. It examines important sources of biological data, how they are archived and made available to researchers, and what computational tools are available to use them effectively in research. In the process, it covers basic concepts in statistics, mathematics, and computer science needed to effectively use these resources and understand their results. Specific topics covered include sequence data, searching and alignment, structural data, genome sequencing, genome analysis, genetic variation, gene and protein expression, and biological networks and pathways. Module 2 covers computational cell biology, including biological modeling and image analysis. It includes homeworks requiring modification of scripts to perform computational analyses. The modeling component includes computer models of population dynamics, biochemical kinetics, cell pathways, neuron behavior, and stochastic simulations. The imaging component includes basics of machine vision, morphological image analysis, image classification and image-derived models. Lectures and examinations are joint with 03-250 but recitations are separate. Recitations for this course are intended primarily for computer science, statistics or engineering majors at the undergraduate or graduate level who have had significant prior experience with computer science or programming. Students may not take both 02-250/03-250 and either 02-251/03-251 or 02-252/03-252 for credit. Prerequisite: 03-121

02-251 Introduction to Computational Molecular Biology
Spring: 6 units
This 6-unit minicourse provides a general introduction to computational tools for biology with specific emphasis on molecular biology and genomics. Along with 02-252, it makes up one half of the full Introduction to Computational Biology, 02-250, although either half can be taken individually. 02-251 examines important sources of biological data, how they are archived and made available to researchers, and what computational tools are available to use them effectively in research. In the process, it covers basic concepts in statistics, mathematics, and computer science needed to effectively use these resources and understand their results. Specific topics to be covered include sequence data, searching and alignment, structural data, genome sequencing, genome analysis, genetic variation, gene and protein expression, and biological networks and pathways. Lectures and examinations are joint with 03-251 but recitations are separate. Recitations for this course are intended primarily for computer science, statistics or engineering majors at the undergraduate or graduate level who have had significant prior experience with computer science or programming. Students may not take both 02-251/03-251 and 02-250/03-250 for credit. Prerequisite: 03-121

02-252 Introduction to Computational Cell Biology
Spring: 6 units
This 6-unit minicourse provides an overview of important modeling and image analysis applications of computers to solve problems in cell biology. Along with 02-251, it makes up one half of the full Introduction to Computational Biology, 02-250, although either half can be taken individually. Major topics covered are computer models of population dynamics, biochemical kinetics, cell pathways, neuron behavior, and stochastic simulations. The imaging component includes basics of machine vision, morphological image analysis, image classification and image-derived models. It includes homeworks requiring modification of scripts to perform computational analyses. Lectures and examinations are joint with 03-252 but recitations are separate. Recitations for this course are intended primarily for computer science, statistics or engineering majors at the undergraduate or graduate level who have had significant prior experience with computer science or programming. Students may not take both 02-252/03-252 and 02-250/03-250 for credit. Prerequisite: 03-121
02-261 Quantitative Cell and Molecular Biology Laboratory
Fall: 9 units
This is an introductory laboratory-based course designed to teach basic biological laboratory skills used in exploring the quantitative nature of biological systems and the reasoning required for performing research in computational biology. Over the course of the semester, students will perform many experiments and quantitatively analyze the results of these experiments. Students will also have the opportunity to design experiments based on the data they collect. During this course students will be using traditional, well-developed techniques to answer open questions. What microbes are found in the food we eat? What changes do cells undergo during apoptosis? Understanding the results of these experiments will require students to think critically about the results they generate, the appropriate controls required to confirm results, and the biological context within which these results were obtained. During this course students will gain experience in many aspects of scientific research, including: Sequencing and analyzing a large and diverse population of DNA, Designing and performing PCR for a variety of analyses, Maintaining cell cultures, Taking brightfield and fluorescent microscopy images, Developing methods for automated analysis of cell images, Communicating results to peers and colleagues. As space is limited, laboratory sections will be small. Additional sections will be added to accommodate all students on the waitlist. Course Outline: (1) 3-hour lab per week (1) 2-hour lecture per week. 9 units.

02-317 Algorithms in Nature
Fall: 9 units
Computer systems and biological processes often rely on networks of interacting entities to reach joint decisions, coordinate and respond to inputs. There are many similarities in the goals and strategies of biological and computational systems which suggest that each can learn from the other. These include the distributed nature of the networks (in biology molecules, cells, or organisms often operate without central control), the ability to successfully handle failures and attacks on a subset of the nodes, modularity and the ability to reuse certain components or sub-networks in multiple applications and the use of stochasticity in biology and randomized algorithms in computer science. In this course we will start by discussing classic biologically motivated algorithms including neural networks (inspired by the brain), genetic algorithms (sequence evolution), non-negative matrix factorization (signal processing in the brain), and search optimization (ant colony formation). We will then continue to discuss more recent bidirectional studies that have relied on biological processes to solve routing and synchronization problems, discover Maximal Independent Sets (MIS), and design robust and fault tolerant networks. In the second part of the class students will read and present new research in this area. Students will also work in groups on a final project in which they develop and test a new biologically inspired algorithm. See also: www.algorithmsinnature.org Pre-requisite: 15-210, no prior biological knowledge required. Prerequisites: 15-251 and 15-210

02-402 Computational Biology Seminar
Fall and Spring: 3 units
This course consists of weekly invited presentations on current computational biology research topics by leading scientists.

02-403 Special Topics in Bioinformatics and Computational Biology
Intermittent: 6 units
A decade ago, mass spectrometry (MS) was merely a qualitative research technique allowing the analysis of samples regardless of the presence of specific biomolecules. However, as MS has turned quantitative, more sophisticated experiments can be performed, such as the recording of signal transduction kinetics and the analysis of the composition of protein complexes and organelles. This makes MS-based proteomics a powerful method to study spatiotemporal protein dynamics. The development of relative quantification approaches, which generally use 2H, 13C or 15N isotope labels, has especially led to an increase in quantification accuracy and set off numerous new experimental approaches to study protein regulation. In this mini-course, we will cover mass spectrometry principles, discuss classical as well as current primary literature addressing method development and quantitative analysis, and highlight state-of-the-art biological studies that employ MS. A combination of lectures, student presentations, and written exercises will establish a thorough knowledge of current bioanalytical MS approaches. Prerequisites: (03-250 or 02-250) and 03-121

02-404 Special Topics in Bioinformatics and Computational Biology: Adv. Phage Genomics
Intermittent
This course is offered occasionally and focuses on a different topic each time. It typically consists of a combination of lectures introducing an important area of computational biology research followed by critical reading and presentation of papers from the literature in the area by students. Grading is based on class participation and occasional written assignments. Prerequisites: 03-121 and 02-250/03-250 or permission of instructor.

02-421 Algorithms for Computational Structural Biology
Intermittent: 12 units
Some of the most interesting and difficult challenges in computational biology and bioinformatics arise from the determination, manipulation, or exploitation of molecular structures. This course will survey these challenges and present a variety of computational methods for addressing them. Topics will include: molecular dynamics simulations, computer-aided drug design, and computer-aided protein design. The course is appropriate for both students with backgrounds in computer science and those in the life sciences.

02-422 Advanced Algorithms for Computational Structural Biology
Fall: 9 units
Missing Course Description - please contact the teaching department.

02-450 Automation of Biological Research
Fall: 9 units
Biology is increasingly becoming a “big data” science, as biomedical research has been revolutionized by automated methods for generating large amounts of data on diverse biological processes. Integration of data from many types of experiments is required to construct detailed, predictive models of cell, tissue or organism behaviors, and the complexity of the systems suggests that these models need to be constructed automatically. This requires iterative cycles of acquisition, analysis, modeling, and experimental design, since it is not feasible to do all possible biological experiments. This course will cover a range of automated biological research methods and a range of computational methods for automating the acquisition and interpretation of the data (especially active learning, proactive learning, compressed sensing and model structure learning). Grading will be based on class participation, homeworks, and a final project. The course is designed for graduate and upper-level undergraduate students with a wide variety of backgrounds. The course is intended to be self-contained but students may need to do some additional work to gain fluency in core concepts. Students should have a basic knowledge of biology, statistics, and programming. Experience with Machine Learning is useful but not mandatory. Prerequisite: 15-122
Course Website: http://jane.compbio.cmu.edu/courses/automationbioresearch

02-500 Undergraduate Research in Computational Biology
Fall and Spring
This course is for undergraduate students who wish to do supervised research for academic credit with a Computational Biology faculty member. Interested students should first contact the Professor with whom they would like to work. If there is mutual interest, the Professor will direct you to the Academic Programs Coordinator who will enroll you in the course.

02-510 Computational Genomics
Spring: 12 units
Dramatic advances in experimental technology and computational analysis are fundamentally transforming the basic nature and goal of biological research. The emergence of new frontiers in biology, such as evolutionary genomics and systems biology is demanding new methodologies that can continuously quantitatively address issues of both the computational and mathematical sophistication. In this course we will discuss classical approaches and latest methodological advances in the context of the following biological problems: 1) Computational genomics, focusing on gene finding, motifs detection and sequence evolution. 2) Analysis of high throughput biological data, such as gene expression data, focusing on issues ranging from data acquisition to pattern recognition and classification. 3) Molecular and regulatory evolution, focusing on phylogeny, inference and regulatory network evolution, and 4) Systems biology, concerning how to combine sequence, expression and other biological data sources to infer the structure and function of different systems in the cell. From the computational side this course focuses on modern machine learning methodologies for computational problems in molecular biology and genetics, including probabilistic modeling, inference and learning algorithms, pattern recognition, data integration, time series analysis, active learning, etc.
02-512 Computational Methods for Biological Modeling and Simulation
Fall: 9 units
This course covers a variety of computational methods important for modeling and simulation of biological systems. It is intended for graduates and advanced undergraduates with either biological or computational backgrounds who are interested in developing computer models and simulations of biological systems. The course will emphasize practical algorithms and algorithm design methods drawn from various disciplines of computer science and applied mathematics that are useful in biological applications. The general topics covered will be models for optimization problems, simulation and sampling, and parameter tuning. Course work will include problems sets with significant programming components and independent or group final projects. Prerequisites: 03-121 and (02-201 or 15-122 or 02-601)

02-513 Algorithms and Data Structures for Scientists
Spring: 12 units
Introduction to design and analysis of algorithms and data structures. Emphasis placed on techniques that are useful for the analysis of scientific data. Topics include dynamic programming, linear programming, network flows, local and heuristic search, and randomization. NP-completeness and approximation algorithms will also be covered. Data structures discussed will include balanced trees, priority queues, trees for geometric data, string data structures, and hashing. Minimal previous algorithmic knowledge is assumed. Classwork may include programming assignments, but strong programming skills not required. This course is not open to CS majors.

02-514 String Algorithms
Fall: 12 units
Provides an in-depth look at modern algorithms used to process string data, particularly those relevant to genomics. The course will cover the design and analysis of efficient algorithms for processing enormous collections of strings. Topics will include string search; inexact matching; string compression; string data structures such as suffix trees, suffix arrays, and searchable compressed indices; and the Burrows-Wheeler transform. Applications of these techniques in biology will be presented, including genome assembly, transcript assembly, whole-genome alignment, gene expression quantification, read mapping, and search of large sequence databases. No knowledge of biology is assumed, and the topics covered will be of use in other fields involving large collections of strings. Programming proficiency is required.

Course Website: http://www.cs.cmu.edu/~ckingsf/class/02-714

02-530 Cell and Systems Modeling
Spring: 12 units
This course will introduce students to the theory and practice of modeling biological systems from the molecular to the organism level with an emphasis on intracellular processes. Topics covered include kinetic and equilibrium descriptions of biological processes, systematic approaches to model building and parameter estimation, analysis of biochemical circuits modeled as differential equations, modeling the effects of noise using stochastic methods, modeling spatial effects, and modeling at higher levels of abstraction or scale using logical or agent-based approaches. A range of biological models and applications will be considered including gene regulatory networks, cell signaling, and cell cycle regulation. Weekly lab sessions will provide students hands-on experience with methods and models presented in class. Course requirements include regular class participation, bi-weekly homework assignments, a take-home exam, and a final project. Prerequisites: The course is designed for graduate and upper-level undergraduate students with a wide variety of backgrounds. The course is intended to be self-contained but students may need to do some additional work to gain fluency in core concepts. Students should have a basic knowledge of calculus, differential equations, and chemistry as well as some previous exposure to molecular biology and biochemistry. Experience with programming and numerical computation is useful but not mandatory. Laboratory exercises will use Matlab as the primary modeling and computational tool augmented by additional software as needed.

02-601 Programming for Scientists
Fall and Summer: 12 units
Provides a practical introduction to programming for students with little or no prior programming experience. Extensive programming assignments will illustrate programming concepts, languages, and tools. Programming assignments will be based on analytical tasks that might be faced by scientists and will typically include parsing, statistical analysis, simulation, and optimization. Principles of good software engineering will also be stressed. Most programming assignments will be done in the Go programming language, an industry-supported, modern programming language, the syntax of which will be covered in depth. Several other assignments will be given in other programming languages such as Python and Java to highlight the commonalities and differences between languages. No prior programming experience is assumed, and no biology background is needed. Analytical skills and mathematical maturity are required. Course not open to CS majors.

02-703 Special Topics in Bioinformatics and Computational Biology
Intermittent: 6 units
A decade ago, mass spectrometry (MS) was merely a qualitative research technique allowing the analysis of samples regarding the presence of specific biomolecules. However, as MS has turned quantitative, more sophisticated experiments can be performed, such as the recording of signal transduction kinetics and the analysis of the composition of protein complexes and organelles. This makes MS-based proteomics a powerful method to study spatiotemporal protein dynamics. The development of relative quantification approaches, which generally use 2H, 13C or 15N isotope labels, has especially led to an increase in quantification accuracy and set off numerous new experimental approaches to study protein regulation. In this mini-course, we will cover mass spectrometry principles, discuss classical as well as current primary literature addressing method development and quantitative analysis, and highlight state-of-the-art biological studies that employ MS. A combination of lectures, student presentations, and written exercises will establish a thorough knowledge of current bio-analytical MS approaches.

02-721 Algorithms for Computational Structural Biology
Intermittent: 12 units
Some of the most interesting and difficult challenges in computational biology and bioinformatics arise from the determination, manipulation, or exploitation of molecular structures. This course will survey these challenges and present a variety of computational methods for addressing them. Topics will include: molecular dynamics simulations, computer-aided drug design, and computer-aided protein design. The course is appropriate for both students with backgrounds in computer science and those in the life sciences.

02-740 Bioimage Informatics
Spring: 12 units
The goals of this course are to provide students with the following: the ability to use mathematical techniques such as linear algebra, Fourier theory and sampling in more advanced signal processing settings; fundamentals of multiresolution and wavelet techniques; and in-depth coverage of some bioimaging applications such as compression and denoising. Upon successful completion of this course, the student will be able to explain the importance and use of signal representations in building more sophisticated signal processing tools, such as wavelets; think in basic time-frequency terms; describe how Fourier theory fits in a bigger picture of signal representations; use basic multirate building blocks, such as a two-channel filter bank; characterize the discrete wavelet transform and its variations; construct a time-frequency decomposition to fit a given signal; explain how these tools are used in various applications; and apply these concepts to solve a practical bioimaging problem through an independent project. Pre-requisite: 18-791, or permission of instructor. (Also known as 18-799) Prerequisite: 18-791
Human-Computer Interaction Courses

05-291 Learning Media Design
Fall: 9 units
[DeaTe collaborative course] Learning is a complex human phenomenon with cognitive, social and personal dimensions that need to be accounted for in the design of technology enhanced learning experiences. In this studio course students will apply learning science concepts to critique existing forms of learning media, establish a set of design precedents to guide project work and produce a series of design concepts that support learning interactions in a real-world context. Collaborating in small interdisciplinary teams, students will partner with a local informal learning organization (e.g. museum, after school program provider, maker space) to conduct learning design research studies, synthesize findings, establish learning goals and iteratively prototype and assess design concepts. As final deliverables, students will present their design research findings, design concepts, and prototypes to stakeholders, and draft a media-rich proposal for their learning media concept to pitch to a local funder. Please note that there may be usage/materials fees associated with this course. Please note that there may be usage/materials fees associated with this course.
Prerequisite: 05-292

05-292 Learning Media Methods
Spring: 6 units
Learning Media Methods brings together students from across the disciplines to consider the design of mediated learning experiences through a project-based inquiry course. Students will be introduced to a range of design research methods and associated frameworks that explore the cognitive, social and affective dimensions of learning in everyday contexts through readings, invited lectures, in-class activities and assignments. Students will conduct a series of short design research studies to define learning goals and develop supporting design concepts that improve learning outcomes for diverse participants in informal learning settings (e.g. museums, afterschool programs, maker spaces or online). In concept development, we will look at how to position technology and question its role in the setting to engage and foster positive learning interactions. The course will culminate in a media-rich presentation of design concepts to a stakeholder audience, and include an evaluation plan describing how learning outcomes for the project would be assessed.
Corequisite: 80-292

05-320 Social Web
12 units
With the growth of online environments like MySpace, Second Life, World of Warcraft, Wikipedia, blogs, online support groups, and open source development communities, the web is no longer just about information. This course, jointly taught by a computer scientist and a behavioral scientist, will examine a sampling of the social, technical and business challenges social web sites must solve to be successful, teach students how to use high-level tools to analyze, design or build online communities, and help them understand the social impact of spending at least part of their lives online. This class is open to advanced undergraduates and graduate students with either technical or non-technical backgrounds. Course work will include lectures and class discussion, homework, class presentations, and a group research or design project.
Course Website: http://sweb12.hciresearch.org/

05-331 Building Virtual Worlds
Fall: 24 units
This is a project course, where interdisciplinary teams build desktop and immersive (helmet-based) interactive virtual worlds. The course will cover world building, environmental design, non-linear story telling, and related topics. Students will use 3D Studio Max (CAD modeler), paint tools, such as Adobe PhotoShop and DeepPaint, sound processing tools, and the Alice authoring system (www.alice.org/bvw.htm). Each year, we hold an exhibition in McConomy auditorium to show class projects to the Carnegie Mellon community. The goal of the course is to take students with varying talents, backgrounds, and perspectives and put them together to do what they couldn't do alone. The course is targeted at undergraduates, but grad students may also enroll. To enroll, students must have ONE of the following skills: Modeling with 3D Studio Max Painting using shadow/shape/ light in a realistic style; Programming, as evidenced by using the Alice system (www.alice.org); Ability to compose and record original music; Storyboarding Production tracking. The key is that there are no "idea people" in the course; everyone must share in the mechanical creation of the worlds. This is a hands-on course and it takes a lot of time, but most students find it very fulfilling and fun. Note that we don't try to teach artists to program, or engineers to paint; we form teams where everyone does what they're already skilled at to attack a joint project. Class time is roughly split between regular lectures, display/critique of group projects, and guest lectures.
Course Website: http://www.alice.org/bvw.htm

05-391 Designing Human Centered Software
Spring: 12 units
Why are things so hard to use these days? Why doesn’t this thing I just bought work? Why is this web site so hard to use? These are frustrations that we have all faced from systems not designed with people in mind. The question this course will focus on is: how can we design human-centered systems that people find useful and usable? This course is an introduction to designing, prototyping, and evaluating user interfaces. If you take only one course in Human-Computer Interaction, this is the course for you. This class is open to all undergrads and grad students, with either technical or non-technical backgrounds. We will cover theory as well as practical application of ideas from Human-Computer Interaction. Course work includes lectures, class discussion, homework, class presentations, and group project.
Course Website: http://www.hcii.cmu.edu/courses/designing-human-centered-software

05-392 Interaction Design Overview
Fall: 9 units
This studio course offers a broad overview of communication and interaction design. Students will learn design methodologies such as brainstorming, sketching, storyboarding, wire framing, and prototyping. Students learn to take a human-centered design approach to their work. Assignments include short in-class exercises as well as individual and team-based projects. Students take part in studio critiques, engaging in critical discussions about the strengths and weaknesses of their own work and the work of others. No coding is required.

05-395 Applications of Cognitive Science
Spring: 9 units
The goal of this course is to examine cases where basic research on cognitive science, including cognitive neuroscience, has made its way into application, in order to understand how science gets applied more generally. The course focuses on applications that are sufficiently advanced as to have made an impact outside of the research field per se; for example, as a product, a change in practice, or a legal statute. Examples are virtual reality (in vision, hearing, and touch), cognitive tutors, phonologically based reading programs, latent semantic analysis applications to writing assessment, and measures of consumers’ implicit attitudes. The course will use a case-study approach that considers a set of applications in detail, while building a general understanding of what it means to move research into the applied setting. The questions to be considered include: What makes a body of theoretically based research applicable? What is the pathway from laboratory to practice? What are the barriers - economic, legal, entrenched belief or practice? The format will emphasize analysis and discussion by students. They should bring to the course an interest in applications; extensive prior experience in cognitive science is not necessary. The course will include tutorials on basic topics in cognitive science such as perception, memory, and spatial cognition. These should provide sufficient grounding to discuss the applications.
Course Website: http://www.hcii.cmu.edu/courses/applications-cognitive-science
05-410 User-Centered Research and Evaluation  
Fall: 12 units  
This course provides an overview and introduction to the field of human-computer interaction (HCI). It introduces students to tools, techniques, and sources of information about HCI and provides a systematic approach to design. The course increases awareness of good and bad design through observation of existing technology, and teaches the basic skills of task analysis, and analytic and empirical evaluation methods. This is a companion course to courses in visual design (15-421) and software implementation (05-430, 05-431). When registering for this course, undergraduate students are automatically placed the wait list. Students will be then moved into the class, based on if they are in the BHCI second major and year in school. This course is NOT open to students outside the HCI major. When registering for this course, undergraduate students are automatically placed the wait list. Students will be then moved into the class, based on if they are in the BHCI second major and year in school.

05-411 Cognitive Crash Dummies  
6 units  
Crash dummies in the auto industry save lives by testing the physical safety of automobiles before they are brought to market. "Cognitive crash dummies" save time, money, and potentially even lives, by allowing computer-based system designers to test their design ideas before implementing them in products and processes. This mini course will review the state of the art of perceptual, cognitive and motor modeling for assessing designs before building working systems. This course will introduce reading breaking research in predicting different aspects of human performance and building models in established modeling frameworks. No prior experience in human performance modeling is assumed; students from all disciplines are welcome.

05-413 Human Factors  
Fall: 9 units  
This course uses theory and research from human factors, cognitive science, and social science to understand and design the interactions of humans with the built world, tools, and technology. The course emphasizes current work in applied domains such as automotive design, house construction, medical human factors, and design of information devices. The course also will emphasize not only individual human factors (e.g., visual response, anthropometry) but also the organizational arrangements that can amplify or correct human factors problems. Through reading, discussion, and projects, you will learn about human perceptual, cognitive, and physical processes that affect how people interact with, and use, technology and tools. You will learn why we have so many automobile accidents, voting irregularities, and injuries from prescription medication. You will learn some tried and true solutions for human factors problems, and some of the many problems in human factors that remain. You will also have gained experience in research in this field.

Course Website: http://www.hcii.cs.cmu.edu

05-417 Computer-mediated Communication  
Spring: 6 units  
This course examines fundamental aspects of interpersonal communication and considers how different types of computer-mediated communications (CMC) technologies affect communication processes. Among the topics we will consider are: conversational structure and CMC, tools to support nonverbal and paralinguistic aspects of communication such as gesture and eye gaze, and social and cultural dimensions of CMC. Students will be expected to post to weekly discussion lists, to write a paper on a specific aspect of CMC, and to present a talk on their final project to the class. The course should be appropriate for graduate students in all areas and for advanced undergraduates.

05-418 Design Educational Games  
Spring: 12 units  
The potential of digital games to improve education is enormous. However, it is a significant challenge to create a game that is both fun and educational. In this course, students will learn to meet this challenge by combining processes and principles from game design and instructional design. Students will also learn to evaluate their games for fun, learning, and the integration of the two. They will be guided by the EDGE framework for the analysis and design educational games. The course will involve a significant hands-on portion, in which students learn a design process to create educational games? digital or non-digital. They will also read about existing educational games and discuss game design, instructional design, learning and transfer, and the educational effectiveness of digital games. They will analyze an educational game and present their analysis to the class.

Course Website: http://www.hcii.cmu.edu/courses/design-educational-games

05-430 Programming Usable Interfaces  
Spring: 15 units  
This course is combines lecture, and an intensive programming lab and design studio. It is for those who want to express their interactive ideas in working prototypes. It will cover the importance of human-computer interaction/interface design, iterative design, input/output techniques, how to design and evaluate interfaces, and research topics that will impact user interfaces in the future. In lab, you will learn how to design and program effective graphical user interfaces, and how to perform user tests. We will cover a number of prototyping tools and require prototypes to be constructed in each, ranging from animated mock-ups to fully functional programs. Assignments will require implementing UIs, testing that interface with users, and then modifying the interface based on findings. Some class sessions will feature design reviews of student work. This course is for HCII Masters students and HCI dual majors with a minimal programming background. Students will often not be professional programmers, but will need to interact with programmers. REGISTRATION SELECTION: Students taking this course can sign up for either Prototyping Lab recitation. PREREQUISITES: Proficiency in a programming language, program structure, algorithm analysis, and data abstraction. Normally met through an introductory programming course using C, C++, Pascal or java, such as 15100, 15112, 15127 or equivalent. Students entering this course should be able to independently write a 300-line program in 48 hours. Prerequisites: 15-112 or 15-100 or 15-127

05-431 Software Structures for User Interfaces  
Fall: 15 units  
(15 credit, combined lecture and lab) This course considers the basic and detailed concepts that go into building software to implement user interfaces. It considers factors of input, output, application interface, and related infrastructure as well as the typical patterns used to implement them. It will also consider how these components are organized and managed within a well-structured object oriented system. After considering these fundamental concepts in the first portion of the course, the later part will consider advanced topics related to emerging future concepts in user interface design. The course includes an intensive programming lab, either on the topic of mobile or web interfaces. This course is intended for HCII Master, BHCI dual majors and others who wish to understand the structures needed for professional development of interactive systems, and has a strong programming background. PREREQUISITES: Proficiency in a programming language, program structure, algorithm analysis, and data abstraction. WAITLIST LOGISTICS: Note that ALL students who register for this class will initially be placed on a waitlist. Your position on the waitlist is not an indication of whether you will be accepted into the class. Contacting the instructor will not move you off the waitlist. Priority for getting off the waitlist are MHCII students, BHCI students (more senior students first), and then others.

05-432 Personalized Online Learning  
Fall: 12 units  
Online learning has become widespread (e.g., MOOCs, online and blended courses, and Khan Academy) and many claim it will revolutionize higher education and K-12. How can we make sure online learning is maximally effective? Learners differ along many dimensions and they change over time. Therefore, advanced learning technologies must adapt to learners to provide individualized learning experiences. This course covers a number of proven personalization techniques used in advanced learning technologies. One of the techniques is the use of cognitive modeling to personalize practice of complex cognitive skills in intelligent tutoring systems. This approach, developed at CMU, may well be the most significant application of cognitive science in education and is commercially successful. We will also survey newer techniques, such as personalization based on student metacognition, affect, and motivation. Finally, we will look at customization approaches that are widely believed to be effective but have not proven to be so. The course involves readings and discussion of different ways of personalizing instruction, with an emphasis on cognitive modeling approaches. Students will learn to use the Cognitive Tutor Authoring Tools (CTAT, http://ctat.pact.cs.cmu.edu) to implement tutor prototypes that rely on computer-executable models of human problem solving to personalize instruction. The course is meant for graduate or advanced undergraduate students in Human-Computer Interaction, Psychology, Computer Science, Design, or related fields, who are interested in educational applications. Students should either have some programming skills or experience in the cognitive psychology of human problem solving, or experience with instructional design.

Course Website: http://www.hcii.cmu.edu/courses/personalized-online-learning
05-433 Programming Usable Interfaces OR Software Structures for Usable Interfaces
Fall: 6 units
Section A: Programming Usable Interfaces Section B: Software Structures for Usable Interfaces This is a lecture-only course (see 05-430/05-630 or 05-431/631 for the lecture + lab version of these courses) that is intended for those who want to learn how to design and evaluate user interfaces. We will cover the importance of human-computer interaction and interface design, the iterative design cycle used in HCI, an overview of input and output techniques, how to design and evaluate interaction techniques, and end with a discussion of hot topics in research that will impact user interfaces in the coming years. This course is only intended for HCI Masters students or HCI undergraduate majors who have already taken an associated User Interface lab, or non-MHCI/BHCI students interested in the design of user interfaces. PREREQUISITES: There are no prerequisites for this lecture-only course.

05-434 Machine Learning in Practice
Fall and Spring: 12 units
Machine Learning is concerned with computer programs that enable the behavior of a computer to be learned from examples or experience rather than dictated through rules written by hand. It has practical value in many application areas of computer science such as on-line communities and digital libraries. This class is meant to teach the practical side of machine learning for applications, such as mining newsgroup data or building adaptive user interfaces. The emphasis will be on learning the process of applying machine learning effectively to a variety of problems rather than emphasizing an understanding of the theory behind what makes machine learning work. This course does not assume any prior exposure to machine learning theory or practice. In the first 2/3 of the course, we will cover a wide range of learning algorithms that can be applied to a variety of problems. In particular, we will cover topics such as decision trees, rule based classification, support vector machines, Bayesian networks, and clustering. In the final third of the class, we will go into more depth on one application area, namely the application of machine learning to problems involving text processing, such as information retrieval or text categorization.

Course Website: http://www.hcii.cmu.edu/courses/applied-machine-learning

05-435 Advanced Fabrication Techniques for HCI
Fall: 12 units
This course will consider how new fabrication techniques such as 3D printing, laser cutting, CNC machining and related computer controlled technologies can be applied to problems in Human-Computer Interaction. Each offering will concentrate on a particular application domain for its projects. This year the course will consider assistive technology. This course will be very hands-on and skills-oriented, with the goal of teaching students the skills necessary to apply these technologies to HCI problems such as rapid prototyping of new device concepts. This year, every student in this course will build and take home a 3D printer. (There will be $400-$500 cost associated with this course to make that possible. Details on this are still to be determined.)

05-439 The Big Data Pipeline: Collecting and Using Big Data for Interactive Systems
Spring: 12 units
This course covers techniques and technologies for creating data driven interfaces. You will learn about the entire data pipeline from sensing to cleaning data to different forms of analysis and computation.

Course Website: http://data.cmu.edu

05-499 Special Topics in HCI
Fall and Spring: 12 units
D: Jason Hong - Ubiquitous Personal Smart Agents

05-509 Game Design
Spring: 12 units
05-509 Game Design Spring: 12 units The goal of this course is to prepare students interested in entertainment technology for a career involving design of computer games and other interactive experiences. Students in this course will read and write about game design, and design many games of their own. Do not mistake this for a course in computer game development. This course is focused on the rules and methods of game design, which remain fairly constant regardless of the technology used to develop a game. While technology will play a significant role in our studies, technological details will not be our focus. Students will study and design games of all sorts: card games, dice games, athletic games, story games, and yes, even video games. How to design games, how to design them well, and how to see your designs to completion will be what students master in this course.

05-540 Rapid Prototyping of Computer Systems
Spring: 12 units
This is a project-oriented course, which will deal with all four aspects of project development: the application, the artifact, the computer-aided design environment, and the physical prototyping facilities. The class consists of students from different disciplines who must synthesize and implement a system in a short period of time. Upon completion of this course the student will be able to: generate systems specifications from a perceived need; partition functionality between hardware and software; produce interface specifications for a system composed of numerous subsystems; use computer-aided development tools; fabricate, integrate, and debug a hardware/software system; and evaluate the system in the context of an end user application. The class consists of students from different disciplines who must synthesize and implement a system in a short period of time.

Course Website: http://www.hcii.cmu.edu/courses/rapid-prototyping-computer-systems

05-571 Undergraduate Project in HCI
Spring: 12 units
Experiential learning is a key component of the MHCi program. Through a subdisciplinary team project, students apply classroom knowledge in analysis and evaluation, implementation and design, and develop skills working in multidisciplinary teams. Student teams work with Carnegie Mellon University-based clients or external clients to iteratively design, build and test a software application which people directly use. Prerequisites: 05-431 or 05-430 or 05-410 or 05-433 or 05-633 or 05-631 or 05-630 or 05-610

Course Website: http://www.hcii.cmu.edu/courses/undergraduate-project-hci

05-589 Independent Study in HCI-UG
All Semesters
In collaboration with and with the permission of the professor, undergraduate students may engage in independent project work on any number of research projects sponsored by faculty. Students must complete an Independent Study Proposal, negotiate the number of units to be earned, complete a contract, and present a tangible deliverable. The Undergraduate Program Advisor's signature is required for HCI undergraduate-level Independent Study courses.

Course Website: http://www.hcii.cmu.edu/independent-study

05-600 HCI Pro Seminar
Fall: 6 units
Students will attend weekly HCI Seminar Series of talks given by national leaders in the field of Human-Computer Interaction, attend communication workshops and conflict management workshops.

Course Website: http://www.hcii.cs.cmu.edu

05-610 User-Centered Research and Evaluation
Fall: 12 units
This course provides an overview and introduction to the field of human-computer interaction (HCI). It introduces students to tools, techniques, and sources of information about HCI and provides a systematic approach to design. The course increases awareness of good and bad design through observation of existing technology, and teaches the basic skills of task analysis, and analytic and empirical evaluation methods. This is a companion course to courses in visual design (05-650) and software implementation (05-630, 05-631). This course is NOT open to students outside of the HCI major.

Course Website: http://www.hcii.cs.cmu.edu
05-650 Interaction Design Studio
Spring: 12 units
This course follows Interaction Design Fundamentals (05-651). Students are expected to apply what they have learned about design thinking and basic practices of interaction design. We follow a human-centered design process that includes research, concept generation, prototyping, and refinement. Students must work effectively as individuals and in small teams to design mobile information systems and other interactive experiences. Assignments approach design on three levels: specific user interactions, contexts of use, and larger systems. Students will become familiar with design methodologies such as sketching, storyboarding, wire framing, prototyping, etc. No coding is required. This course serves as a prerequisite for Interaction Design Studio (05-650).

05-651 Interaction Design Fundamentals
Fall: 12 units
This studio course introduces students to design thinking and the basic practices of interaction design. We follow a human-centered design process that includes research, concept generation, prototyping, and refinement. Students must work effectively as individuals and in small teams to design mobile information systems and other interactive experiences. Assignments approach design on three levels: specific user interactions, contexts of use, and larger systems. Students will become familiar with design methodologies such as sketching, storyboarding, wire framing, prototyping, etc. No coding is required. This course serves as a prerequisite for Interaction Design Studio (05-650).

05-823 E-Learning Design Principles
Fall: 12 units
This course is about e-learning design principles, the evidence and theory behind them, and how to apply these principles to develop effective educational technologies. It is organized around the book "e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning" by Clark & Mayer with further readings drawn from cognitive science, educational psychology, and human-computer interaction. You will learn design principles 1) for combining words, audio, and graphics in multimedia instruction, 2) for combining examples, explanations, practice and feedback in online support for learning by doing, and 3) for balancing learner vs. system control and supporting student metacognition. You will read about the experiments that support these design principles, see examples of how to design such experiments, and practice applying the principles in educational technology development.
Course Website: http://www.learnlab.org/research/wiki/index.php/E-learning_Design_Principles_2013#Course_Details

05-831 Building Virtual Worlds
Fall: 24 units
This is a project course, where interdisciplinary teams build desktop and immersive (helmet-based) interactive virtual worlds. The course will cover world building, environmental design, non-linear storytelling, and related topics. Students will use 3D Studio Max (CAD modeler), paint tools, such as Adobe PhotoShop and DeepPaint, sound processing tools, and the Alice authoring system (www.alice.org/bvw.htm). Each year, we hold an exhibition in McConomy auditorium to show class projects to the Carnegie Mellon community. The goal of the course is to take students with varying talents, backgrounds, and perspectives and put them together to do what they couldn't do alone. The course is targeted at undergraduates, but grad students may also enroll. To enroll, students must have one of the following skills: Modeling with 3D Studio Max Painting using shadow/shape/light in a realistic style Programming, as evidenced by using the Alice system (www.alice.org) Ability to compose and record original music Storyboarding Production tracking Other relevant skills The key thing is that there are no "idea people" in the course; everyone must share in the mechanical creation of the worlds. This is a hands-on course and it takes a lot of time, but most students find it very fulfilling and fun. Note that we don't try to teach artists to program, or engineers to paint; we form teams where everyone does what they're already skilled at to attack a joint project. We are scheduled for 2 weekly meetings, 2.5 hours each. These slots are roughly split between regular lectures, discuss/group projects, and guest lectures.
Course Website: http://www.alice.org/bvw.htm

05-840 Tools for Online Learning
Fall: 12 units
This course is for students in the METALS program.
08-340 Green Computing  
Spring: 9 units  
Energy is a key societal resource. However, our energy usage is rising at an alarming rate and therefore it has become critical to manage its consumption more efficiently for long term sustainability. This course introduces students to the exciting area of “Green Computing”, and is organizationally divided into two tracks. The first track is “Energy-Efficient Computing”, which considers the state of the art techniques for improving the energy efficiency of mobile devices, to laptop and desktop class computers and finally to data centers. We will cover energy efficiency across the hardware/software stack, starting from the individual components like processors and radio interfaces to system level architectures and optimizations. The second track is “Applying Computing towards Sustainability”, covering topics that leverage computing to reduce the energy footprint of our society. In particular, we will focus on Smart Buildings and the Smart Grid, covering topics such as sensing, modeling and controlling the energy usage of buildings, new operating systems or software stacks for the smart infrastructure, as well as the privacy and security issues with the new “internet of things”. The goal of this course is to help students acquire some of the knowledge and the skills needed to do research in this space of “Green Computing”. Although the course is listed within SCS, it should be of interest to students in several departments, including ECE, MechE, CEE, EPP and Architecture.  
Course Website: http://www.cs.cmu.edu/~yuvraja/courses/08-340

08-540 Green Computing  
Spring: 9 units  
Energy is a key societal resource. However, our energy usage is rising at an alarming rate and therefore it has become critical to manage its consumption more efficiently for long term sustainability. This course introduces students to the exciting area of “Green Computing”, and is organizationally divided into two tracks. The first track is “Energy-Efficient Computing”, which considers the state of the art techniques for improving the energy efficiency of mobile devices, to laptop and desktop class computers and finally to data centers. We will cover energy efficiency across the hardware/software stack, starting from the individual components like processors and radio interfaces to system level architectures and optimizations. The second track is “Applying Computing towards Sustainability”, covering topics that leverage computing to reduce the energy footprint of our society. In particular, we will focus on Smart Buildings and the Smart Grid, covering topics such as sensing, modeling and controlling the energy usage of buildings, new operating systems or software stacks for the smart infrastructure, as well as the privacy and security issues with the new “internet of things”. The goal of this course is to help students acquire some of the knowledge and the skills needed to do research in this space of “Green Computing”. Although the course is listed within SCS, it should be of interest to students in several departments, including ECE, MechE, CEE, EPP and Architecture.  
Course Website: http://www.cs.cmu.edu/~yuvraja/courses/08-340

08-463 Service Innovation  
Spring: 6 units  
This course introduces students to the concept of services and their increasing role in the global economy and global employment, and explores services as a part of a goods-services continuum ranging from products or goods to service encapsulation of products to pure services. Service innovation is defined, and contrasts drawn between service and product innovation. Service innovation mechanisms and barriers to innovation are explored, as are organizational outcomes and measurement of service innovation. Course discussion identifies selected issues in service innovation, such as the innovation value chain, co-creation of service innovations, service innovation for sustainability, innovation in public service, changes in employment, globalization of service innovation, service design, or the ethics of service innovation; and assesses the impact of these on future service innovation agendas.  
08-532 Law of Computer Technology  
Fall: 9 units  
The Law of Computer Technology A survey of how legislatures and courts cope with rapidly advancing computer technologies and how scientific information is presented to, and evaluated by, civil authorities. The course is also an introduction to the legal process generally and the interaction between the legal system and technology organizations. Topics include: patents, copyrights in a networked world, law of the Internet, free speech, data security, technology regulation, international law, and transborder crime. Open to juniors, seniors and graduate students in any school. Open to sophomores by permission of the instructor. Prerequisites: none.  
08-533 Privacy, Policy, Law and Technology  
Fall and Spring: 9 units  
This course focuses on policy issues related to privacy from the perspectives of governments, organizations, and individuals. We will begin with a historical and philosophical study of privacy and then explore recent public policy issues. We will examine the privacy protections provided by laws and regulations, as well as the way technology can be used to protect privacy. We will emphasize technology-related privacy concerns and mitigation, for example: social networks, smartphones, behavioral advertising (and tools to prevent targeted advertising and tracking), anonymous communication systems, big data, and drones. This is part of a series of courses offered as part of the MST-Privacy Engineering masters program. These courses may be taken in any order or simultaneously. Foundations of Privacy (Fall semester) offers more in-depth coverage of technologies and algorithms used to reason about and protect privacy. Engineering Privacy in Software (Spring semester) focuses on the methods and tools needed to design systems for privacy. This course is intended primarily for graduate students and advanced undergraduate students with some technical background. Programming skills are not required. 8-733, 19-608, and 95-818 are 12-unit courses for PhD students. Students enrolled under these course numbers will have extra assignments and will be expected to do a project suitable for publication. 8-533 is a 9-unit course for undergraduate students. Masters students may register for any of the course numbers permitted by their program. This course will include a lot of reading, writing, and class discussion. Students will be able to tailor their assignments to their skills and interests. However, all students will be expected to do some writing and some technical work.  
Course Website: http://cups.cs.cmu.edu/courses/privpolawtech.html

08-541 Hardware and Software Systems for Smart Homes and Buildings  
Spring: 9 units  
Smart automation in the home and buildings hold tremendous promise to enhance the quality of life, improving from products or goods to service encapsulation of products to pure services. Service innovation is defined, and contrasts drawn between service and product innovation. Service innovation mechanisms and barriers to innovation are explored, as are organizational outcomes and measurement of service innovation. Course discussion identifies selected issues in service innovation, such as the innovation value chain, co-creation of service innovations, service innovation for sustainability, innovation in public service, changes in employment, globalization of service innovation, service design, or the ethics of service innovation; and assesses the impact of these on future service innovation agendas.  
08-671 Java for Application Programmers  
Fall and Spring: 6 units  
This course provides an intensive exploration of computer programming in the Java Language for Masters students who have had some prior, but perhaps limited, programming experience in Java or in some other programming language. The course starts with a review of the fundamental topics of programming in Java (data types, operators, control structures) along with discussions of object oriented programming (classes, instances, and class hierarchies). After covering the basics we move on using Java to cover simple graphical user interfaces (using Swing), file I/O, concurrency (i.e., threads), network I/O (HTTP), simple data structures (lists and maps). Students are required to have a reasonably modern laptop computer on which install the Java software used for this course.

Course Website: http://cups.cs.cmu.edu/courses/privpolawtech.html
08-672 J2EE Web Application Development  
Fall: 6 units  
This course will introduce concepts in programming web application servers. We will study of the fundamental architectural elements of programming web sites that produce content dynamically. We will be demonstrating the course using Java Servlets and Java Server Pages. We will also cover the related topics as necessary so that students may build significant applications. Such topics are expected to include: HTML, CSS, HTTP, Relational and Non-Relational Databases, Object-Relational Mapping Tools, Security Issues, AJAX, and Cloud Deployment. Students are required to be familiar with Java Programming before taking this course. Those who are not are encouraged to take 08-671 in mini 1 before taking this course. Students are required to have a reasonably modern laptop computer on which to install the Java software used for this course.

08-733 Privacy, Policy, Law and Technology  
Fall: 12 units  
This course focuses on policy issues related to privacy from the perspectives of governments, organizations, and individuals. We will begin with a historical and philosophical study of privacy and then explore recent public policy issues. We will examine the privacy protections provided by laws and regulations, as well as the way technology can be used to protect privacy. We will emphasize technology-related privacy concerns and mitigation, for example: social networks, smartphones, behavioral advertising (and tools to prevent targeted advertising and tracking), anonymous communication systems, big data, and drones. This is part of a series of courses offered as part of the MSIT-Privacy Engineering masters program. These courses may be taken in any order or simultaneously. Foundations of Privacy (Fall semester) offers more in-depth coverage of technologies and algorithms used to reason about and protect privacy. Engineering Privacy in Software (Spring semester) focuses on the methods and tools needed to design systems for privacy. This course is intended primarily for graduate students and advanced undergraduate students with some technical background. Programming skills are not required. 8-733, 19-608, and 95-818 are 12-unit courses for PhD students. Students enrolled under these course numbers will have extra assignments and will be expected to do a project suitable for publication. 8-533 is a 9-unit course for undergraduate students. Masters students may register for any of the course numbers permitted by their program. This course will include a lot of reading, writing, and class discussion. Students will be able to tailor their assignments to their skills and interests. However, all students will be expected to do some writing and some technical work.

Course Website: http://cups.cs.cmu.edu/courses/privpolawtech.html

08-781 Mobile and Pervasive Computing Services  
Spring: 9 units  
With over 6 billion mobile phone users worldwide, including well over a billion smart phone users, new wireless and pervasive computing applications and services are changing the way enterprises interact with consumers and employees. The explosion in smart phone ownership along with the deployment of 4G networks is leading to a slew of new mobile apps and services. They range from mobile commerce services to wireless enterprise apps and mobile social networking apps, all the way to more futuristic pervasive computing scenarios where phones interact with a number of other everyday objects (e.g. smart homes, smart cars, smart glasses, health/fitness sensors). These apps and services are emerging as part of an increasingly rich ecosystem where context awareness and intelligent predictive technologies are used to offer increasingly personalized experiences to users. This same ecosystem has emerged as the engine behind increasingly targeted marketing and advertising scenarios that also raise challenging privacy issues. The course objective is to introduce participants to the technologies, services and business models associated with Mobile and Pervasive Commerce. It also provides an overview of future trends and ongoing research. You will learn to evaluate critical design tradeoffs associated with different mobile technologies, architectures, interfaces and business models and how they impact the usability, security, privacy and commercial viability of mobile and pervasive computing services and apps. Topics include Mobile Communications, Mobile OS, Mobile Web technologies including app development, Mobile Security, Mobile Payments, Mobile Web Apps and Services (e.g. Mobile Entertainment, Mobile Banking, Mobile, Mobile Social Networking, Mobile Health, etc.), Location-Based Services, RFID, Mobile Enterprise Apps, Pervasive Computing Applications, Context awareness, intelligent assistant technologies, and privacy.

Course Website: http://www.normsadeh.com/ms-course-overview

08-801 Dynamic Network Analysis  
Spring: 12 units  
Who knows who? Who knows what? Who communicates with whom? Who is influential? How do ideas, diseases, and technologies propagate through groups? How do social media, social, knowledge, and technology networks differ? How do these networks evolve? How do network constrain and enable behavior? How can a network be compromised or made resilient? Such questions can be addressed using Network Science. Network Science, a.k.a. social network analysis and link analysis, is a fast-growing interdisciplinary field aimed at understanding simple & high dimensional networks, from both a static and a dynamic perspective. Across an unlimited application space, graph theoretic, statistical, & simulation methodologies are used. An interdisciplinary perspective on network science is provided, with an emphasis on high-dimensional dynamic data. The fundamentals of network science, methods, theories, metrics & confidence estimation, constraints on data collection & bias, and key research findings & challenges are examined. Illustrative networks discussed include social media based (e.g., twitter), disaster response, organizational, semantic, political elite, crises, terror, & P2P networks. Critical procedures covered include: basic centralities and metrics, group and community detection, link inference, network change detection, comparative analytics, and big data techniques. Applications from business, science, art, medicine, forensics, social media & numerous other areas are explored. Key issues addressed: Conceptualization, measurement, comparison & evaluation of networks. Identification of influential nodes and hidden groups. Network emergence, evolution, change & destabilization. Graduate course taught every other year. Prerequisite: Undergraduate-level statistics course or instructor permission. Linear algebra is recommended. Students are encouraged to bring & use their own data, or to use provided data.

08-805 Engineering Privacy in Software  
Spring: 12 units  
This section is for Ph.D. students; all other students should enroll in 80-605. Privacy harms that involve personal data can often be traced back to software design failures, which can be prevented through sound engineering practices. In this course, students will learn how to identify privacy threats due to surveillance activities that enhance modern information systems, including location tracking, behavioral profiling, recommender systems, and social networking. Students will learn to analyze systems to identify the core operating principles and technical means that introduce privacy threats, and they will learn to evaluate and mitigate privacy risks to individuals by investigating system design alternatives. Strategies to mitigating privacy risk will be based on emerging standards and reliable privacy preference data. Students will have the opportunity to study web, mobile- and cyber-physical systems across a range of domains, including advertising, healthcare, law enforcement and social networking. In addition, students will know how, and when, to interface with relevant stakeholders, including legal, marketing and other developers in order to align software design with privacy policy and law.

08-810 Computational Modeling of Complex Socio-Technical Systems  
Spring: 12 units  
Social and cultural systems are complex. Whether considering world transforming events such as the Arab Spring or the impact of health care reforms, the interactions among people, technology, and organizations can generate unanticipated outcomes. Computer simulation is a critical methodology for explaining and predicting these events. In this course, the basics of simulation modeling, design, testing and validation are covered. Different simulation approaches are contrasted such as agent-based modeling and system dynamics.
Language Technologies Institute Courses

11-344 Machine Learning in Practice
Fall: 12 units
Course Website: http://www.lti.cs.cmu.edu/Courses/11-717-desc.htm
Machine Learning is concerned with computer programs that enable the behavior of a computer to be learned from examples or experience rather than dictated through rules written by hand. It has practical value in many application areas of computer science such as on-line communities and digital libraries. This class is meant to teach the practical side of machine learning for applications, such as mining and organizing data or building adaptive user interfaces. The emphasis will be on learning the process of applying machine learning effectively to a variety of problems rather than emphasizing an understanding of the theory behind what makes machine learning work. This course does not assume any prior exposure to machine learning theory or practice. In the first 2/3 of the course, we will cover a wide range of learning algorithms that can be applied to a variety of problems. In particular, we will cover topics such as decision trees, rule-based classification, support vector machines, Bayesian networks, and clustering. In the final third of the class, we will go into more depth on one application area, namely the application of machine learning to problems involving text processing, such as information retrieval or text categorization.

11-411 Natural Language Processing
Spring: 12 units
This course will introduce students to the highly interdisciplinary area of Artificial Intelligence known alternately as Natural Language Processing (NLP) and Computational Linguistics. The course aims to cover the techniques used today in software that does useful things with text in human languages like English and Chinese. Applications of NLP include automatic translation between languages, extraction and summarization of information in documents, question answering and dialog systems, and conversational agents. This course will focus on core representations and algorithms, with some time spent on real-world applications. Because modern NLP relies so heavily on Machine Learning, we’ll cover the basics of discriminative classification and probabilistic modeling as we go. Good computational linguists also know about Linguistics, so topics in linguistics (phonology, morphology, and syntax) will be covered when fitting. From a software engineering perspective, there will be an emphasis on rapid prototyping, a useful skill in many other areas of Computer Science. In particular, we will introduce some high-level languages (e.g., regular expressions and Dyna) and some scripting languages (e.g., Python and Perl) that can greatly simplify prototype implementation. Prerequisite: 15-122

11-411 Machine Learning for Text Mining
Spring: 9 units
This course provides a comprehensive introduction to the theory and implementation of algorithms for organizing and searching large text collections. The first part of the course studies text search engine technology for enterprise and Web environments; the open-source Indri search engine is used as a working example. The second half studies text mining techniques such as clustering, categorization, and information extraction. Programming assignments give hands-on experience with document ranking algorithms, categorizing documents into browsing hierarchies, and related topics.

11-492 Speech Processing
Fall: 12 units
to be determined by the department.

11-617 Language Technologies for Computer Assisted Language Learning
Spring: 12 units
This course studies the design and implementation of CALL systems that use Language Technologies such as Speech Synthesis and Recognition, Machine Translation, and Information Retrieval. After a short history of CALL/LT, students will learn where language technologies (LT) can be used to aid in language learning. From there, the course will explore the specifics of designing software that must interface with a language technology. For each LT, we will explore: what information does the LT require, what type of output does the LT send to the CALL interface, what are the limits of the LT that the CALL designer must deal with, what are the real constraints, what type of training does the LT require. The goal of the course is to familiarize the student with existing systems that use LT and to provide an overview of CALL/LT software the limitations imposed by the LT designing CALL/LT software Grading criteria: several short quizzes term project: production of a small CALL/LT system, verbal presentation and written documentation of design of the software
Course Website: http://www.lti.cs.cmu.edu/Courses/11-717-desc.htm

11-630 MCDS Practicum Internship
Fall
The MCDS Practicum course is used for recording CDS students summer internship units that are in the Analytics track of the program. Section A is used for 7-month internship opportunities Section B is used for Returning Fall Analytic students who DO NOT attain a 7-mo Internship. Section R is used to record MCDS students Internship Requirements

11-696 MIIS Capstone Planning Seminar
Spring: 6 units
The MIIS Capstone Planning Seminar prepares students to complete the MIIS Capstone Project in the following semester. Students are organized into teams that will work together to complete the capstone project. They define project goals, requirements, success metrics, and deliverables; and they identify and acquires data, software, and other resources required for successful completion of the project. The planning seminar must be completed in the semester prior to taking the capstone project.

11-711 Algorithms for NLP
All Semesters: 12 units
Algorithms for NLP is an introductory graduate-level course on the computational properties of natural languages and the fundamental algorithms for processing natural languages. The course will provide an in-depth presentation of the major algorithms used in NLP, including Lexical, Morphological, Syntactic and Semantic analysis, with the primary focus on parsing algorithms and their analysis.

11-716 Graduate Seminar on Dialog Processing
All Semesters: 6 units
Dialog systems and processes are becoming an increasingly vital area of interest both in research and in practical applications. The purpose of this course will be to examine, in a structured way, the literature in this area as well as learn about ongoing work. The course will cover traditional approaches to the problem, as exemplified by the work of Grosz and Sidner, as well as more recent work in dialog, discourse and evaluation, including statistical approaches to problems in the field. We will select several papers on a particular topic to read each week. While everyone will do all readings, a presenter will be assigned to overview the paper and lead the discussion. On occasion, a researcher may be invited to present their own work in detail and discuss it with the group. A student or researcher taking part in the seminar will come away with a solid knowledge of classic work on dialog, as well as familiarity with ongoing trends.

11-721 Grammars and Lexicons
All Semesters: 12 units
Grammars and Lexicons is an introductory graduate course on linguistic data analysis and theory, focusing on methodologies that are suitable for computational implementation. The course will cover a wide range of learning algorithms that can be applied to a variety of problem sets and take-home exams.

11-722 Grammar Formalisms
Intermittent: 12 units
The goal of this course is to familiarize students with grammar formalisms that are commonly used for research in computational linguistics, language technologies, and linguistics. We hope to have students from a variety disciplines (linguistics, computer science, psychology, modern languages, philosophy) in order to cover a broad perspective in class discussions. Comparison of formalisms will lead to a deeper understanding of human language and natural language processing. The course formalisms will include: Head Driven Phrase Structure Grammar, Lexical Functional Grammar, Tree Adjoining Grammar and Categorial Grammar. If time permits, we will cover Penn Treebank, dependency grammar, and Construction Grammar. We will cover the treatment of basic syntactic and semantic phenomena in each formalism, and will also discuss algorithms for parsing and generating sentences for each formalism. If time permits, we may discuss formal language theory and generative capacity. The course is taught jointly by the following faculty of the Language Technologies Institute: Alan Black Alon Lavie Lori Levin (main coordinator)
11-731 Machine Translation
Spring: 12 units
Instructors: Chris Dyer (leader), Alon Lavie. Prerequisites: 11-711 "Algorithms for NLP" or equivalent background is recommended. Course Description: Machine Translation is an introductory graduate-level course surveying the primary approaches and methods for developing modern state-of-the-art automated language translation systems. The main objectives of the course are: Obtain a basic understanding of modern MT systems and MT-related issues. Learn about theory and approaches in Machine Translation and implement the main components of statistical MT systems.

Course Website: http://www.lti.cs.cmu.edu/Courses/11-731-desc.html

11-741 Machine Learning for Text Mining
Spring: 12 units
This course studies the theory, design, and implementation of text-based information systems. The Information Retrieval core components of the course include statistical characteristics of text, representation of information needs and documents, several important retrieval models (Boolean, vector space, probabilistic, inference net, language modeling), clustering algorithms, automatic text categorization, and experimental evaluation. The software architecture components include design and implementation of high-capacity text retrieval and text filtering systems. A variety of current research topics are also covered, including cross-lingual retrieval, document summarization, machine learning, topic detection and tracking, and multi-media retrieval. Prerequisites: Programming and data-structures at the level of 15-212 or higher. Algorithms comparable to the undergraduate CS algorithms course (15-451) or higher. Basic linear algebra (21-241 or 21-341). Basic statistics (36-202) or higher.

Course Website: http://www.lti.cs.cmu.edu/Courses/11-741-desc.html

11-751 Speech Recognition and Understanding
All Semesters: 12 units
The technology to allow humans to communicate by speech with machines or by which machines can understand when humans communicate with each other is rapidly maturing. This course provides an introduction to the theoretical tools as well as the experimental practice that has made the field what it is today. We will cover theoretical foundations, essential algorithms, major approaches, experimental strategies and current state-of-the-art systems and will introduce the participants to ongoing work in representation, algorithms and interface design. This course is suitable for graduate students with some background in computer science and electrical engineering, as well as for advanced undergraduates. Prerequisites: Sound mathematical background, knowledge of basic statistics, good computing skills. No prior experience with speech recognition is necessary. This course is primarily for graduate students in LTI, CS, Robotics, ECE, Psychology, or Computational Linguistics. Others by prior permission of instructor.

11-752 Speech II: Phonetics, Prosody, Perception and Synthesis
Spring: 12 units
The goal of the course is to give the student basic knowledge from several fields that is necessary in order to pursue research in automatic speech processing. The course will begin with a study of the acoustic content of the speech signal. The students will use the spectrographic display to examine the signal and discover its variable properties. Phones in increasingly larger contexts will be studied with the goal of understanding coarticulation. Phonomical rules will be studied as a contextual aid in understanding the spectrographic display. The spectrogram will then serve as a first introduction to the basic elements of prosody. Other displays will then be used to study the three parts of prosody: amplitude, duration, and pitch. Building on these three elements, the student will then examine how the three interact in careful and spontaneous speech. Next, the students will explore perception. Topics covered will be: physical aspects of perception, psychological aspects of perception, testing perception processes, practical applications of knowledge about perception. The second part of this course will cover all aspects of speech synthesis. Students need only have a basic knowledge of speech and language processing. Some degree of programming and statistical modeling will be beneficial, but not required. Taught every other year

Course Website: http://www.lti.cs.cmu.edu/Courses/11-752-desc.html

11-761 Language and Statistics
Fall: 12 units
Language technologies (search, text mining, information retrieval, speech recognition, machine translation, question answering, biological sequence analysis,...) are at the forefront of this century's information revolution. In addition to their use in machine learning, these technologies rely centrally on classic statistical estimation techniques. Yet most CS and engineering undergraduate programs do not prepare students in this area beyond an introductory probability course. This course is designed to plug this hole. The goal of “Language and Statistics” is to ground the data-driven techniques used in language technologies in sound statistical methodology. We start by formulating various language technology problems in both an information theoretic framework (the source-channel paradigm) and a Bayesian framework (the Bayes classifier). We then discuss the statistical properties of words, sentences, documents and whole languages, and the computational formalisms used to represent language. These discussions naturally lead to specific models. This course stress how these different models may be applied to the field what it is today. We will cover theoretical foundations, essential algorithms, major approaches, experimental strategies and current state-of-the-art systems and will introduce the participants to ongoing work in representation, algorithms and interface design. This course is suitable for graduate students with some background in computer science and electrical engineering, as well as for advanced undergraduates. Prerequisites: Sound mathematical background, knowledge of basic statistics, good computing skills. No prior experience with speech recognition is necessary. This course is primarily for graduate students in LTI, CS, Robotics, ECE, Psychology, or Computational Linguistics. Others by prior permission of instructor.

Course Website: http://www.cs.cmu.edu/~roni/11761/

11-762 Language and Statistics II
Spring: 12 units
This course will cover modern empirical methods in natural language processing. It is designed for language technologies students who want to understand statistical methodology in the language domain, and for machine learning students who want to know about current problems and solutions in text processing. Students will, upon completion, understand how statistical modeling and learning can be applied to text, be able to develop and apply new statistical models for problems in their own research, and be able to critically read papers from the major related conferences (EMNLP and ACL). A recurring theme will be the tradeoffs between computational cost, mathematical elegance, and applicability to real problems. The course will be organized around methods, with concrete tasks introduced throughout. The course is designed for SCS graduate students. Prerequisite: Language and Statistics (11-761) or permission of the instructor. Recommended: Algorithms for Natural Language Processing (11-711), Machine Learning (15-681, 15-781, or 11-746). Prerequisite: 11-761

11-791 Intelligent Information Systems Project
Fall: 36 units
This course is designed to plug this hole. The goal of “Language and Statistics” is to ground the data-driven techniques used in language technologies in sound statistical methodology. We start by formulating various language technology problems in both an information theoretic framework (the source-channel paradigm) and a Bayesian framework (the Bayes classifier). We then discuss the statistical properties of words, sentences, documents and whole languages, and the computational formalisms used to represent language. These discussions naturally lead to specific models. This course stress how these different models may be applied to

Course Website: http://www.cs.cmu.edu/~roni/11761/

11-792 Intelligent Information Systems Project
Spring: 12 units
The Software Engineering for IS sequence combines classroom material and assignments in the fundamentals of software engineering (11-791) with a self-paced, faculty-supervised directed project (11-792). The two courses cover all elements of project design, implementation, evaluation, and documentation. Students may elect to take only 11-791; however, if both parts are taken, they should be taken in proper sequence. Prerequisite: 11-791. The course is required for VLSI students.

Prerequisites: 11-791 or 15-393

Course Website: http://www.lti.cs.cmu.edu/Courses/11-791-desc.html

11-927 MLIS Capstone Project
Fall: 36 units
The capstone project course is a group-oriented demonstration of student skill in one or more areas covered by the degree. Typically the result of the capstone project is a major software application. The capstone project course consists of two components. The classroom component guides students in project planning, team management, development of requirements and design specifications, and software tools for managing group-oriented projects. The lab component provides project-specific technical guidance and expertise, for example in the development of a question answering system, dialog, or sentiment analysis application. Thus, each project receives two types of supervision, often from two separate members of the faculty.
Machine Learning Courses

10-401 Introduction to Machine Learning
Fall and Spring: 12 units
TBA

10-601 Introduction to Machine Learning
Fall and Spring: 12 units
Machine Learning (ML) develops computer programs that automatically improve their performance through experience. This includes learning many types of tasks based on many types of experience, e.g., spotting high-risk medical patients, recognizing speech, classifying text documents, detecting credit card fraud, or driving autonomous vehicles. 10601 covers all or most of: concept learning, decision trees, neural networks, linear learning, active learning, estimation & the bias-variance tradeoff, hypothesis testing, Bayesian learning, the MDL principle, the Gibbs classifier, Naive Bayes, Bayes Nets & Graphical Models, the EM algorithm, Hidden Markov Models, K-Nearest-Neighbors and nonparametric learning, reinforcement learning, bagging, boosting and discriminative training. Grading will be based on weekly or biweekly assignments (written and/or programming), a midterm, a final exam, and possibly a project (details may vary depending on the section). 10601 is recommended for CS Seniors & Juniors, quantitative Masters students, & non-MLD PhD students. Prerequisites (strictly enforced): strong quantitative aptitude, college prob/stats course, and programming proficiency. For learning to apply ML practically & effectively, without the above prerequisites, consider 11344/05834 instead. You can evaluate your ability to take the course via a self-assessment exam at: http://www.cs.cmu.edu/~aarli/Class/10701_Spring14/Intro_ML_Self_Evaluation.pdf For section-specific information, see: Section 10601A: https://sites.google.com/site/10601a14spring/about-the-class Section 10601B: http://www.cs.cmu.edu/~10601b/ Prerequisites: 15-122 and (15-151 or 21-127) Corequisites: 36-225 or 36-217 or 21-325 or 15-359
Course Website: https://sites.google.com/site/10601a14spring/about-the-class

10-605 Machine Learning with Large Datasets
Spring: 12 units
Large datasets are difficult to work with for several reasons. They are difficult to visualize, and it is difficult to understand what sort of errors and biases are present in them. They are computationally expensive to process, and often the cost of learning is hard to predict - for instance, and algorithm that runs quickly in a dataset that fits in memory may be exorbitantly expensive when the dataset is too large for memory. Large datasets may also display qualitatively different behavior in terms of which learning methods produce the most accurate predictions. This course is intended to provide a student practical knowledge of, and experience with, the issues involving large datasets. Among the issues considered are: scalable learning techniques, such as streaming machine learning techniques; parallel infrastructures such as map-reduce; practical techniques for reducing the memory requirements for learning methods, such as feature hashing and Bloom filters; and techniques for analysis of programs in terms of memory, disk usage, and (for parallel methods) communication complexity. The class will include programming assignments, and a one-month short project chosen by the student. The project will be designed to compare the scalability of variant learning algorithms on datasets. An introductory course in machine learning, like 10-601 or 10-701, is a prerequisite or a co-requisite. If you plan to take this course and 10-601 concurrently please tell the instructor. The course will include several substantial programming assignments, so an additional prerequisite is 15-211, or 15-214, or comparable familiarity with Java and good programming skills. Undergraduates need permission of the instructor to enroll. Prerequisites: 15-210 or 15-214 Corequisites: 10-701 or 10-601
Course Website: http://curtis.ml.cmu.edu/w/courses/index.php/10-605_in_Spring_2015

10-701 Introduction to Machine Learning
Fall and Spring: 12 units
Machine learning studies the question “How can we build computer programs that automatically improve their performance through experience?” This includes learning to perform many types of tasks based on many types of experience. For example, it includes robots learning to better navigate based on experience gained by roaming their environments, medical decision aids that learn to predict which therapies work best for which diseases based on data mining of historical health records, and speech recognition systems that learn to better understand your speech based on experience listening to you. This course is designed to give PhD students a thorough grounding in the methods, mathematics and algorithms needed to do research and applications in machine learning. Students entering the class with a pre-existing working knowledge of probability, statistics and algorithms will be at an advantage, but the class has been designed so that anyone with a strong numeric background can catch up and fully participate. You can evaluate your ability to take the course via a self-assessment exam that will be made available to you after you register. If you are interested in this topic, but are not a PhD student, or are a PhD student not specializing in machine learning, you might consider the masters-level course on Machine Learning, 10-601. This class may be appropriate for MS and undergrad students who are interested in the theory and algorithms behind ML. You can evaluate your ability to take the course via a self-assessment exam at: http://www.cs.cmu.edu/~aarli/Class/10701_Spring14/Intro_ML_Self_Evaluation.pdf Course Website: http://www.cs.cmu.edu/~epxing/Class/10701-11f/

10-715 Advanced Introduction to Machine Learning
Fall: 12 units
The rapid improvement of sensory techniques and processor speed, and the availability of inexpensive massive digital storage, have led to a growing demand for systems that can automatically comprehend and mine massive and complex data from diverse sources. Machine Learning is becoming the primary mechanism by which information is extracted from Big Data, and a primary pillar that Artificial Intelligence is built upon. This course is designed for Ph.D. students whose primary field of study is machine learning, or who intend to make machine learning methodological research a main focus of their thesis. It will give students a thorough grounding in the algorithms, mathematics, theories, and insights needed to do in-depth research and applications in machine learning. The topics of this course will in part parallel those covered in the general graduate machine learning course (10-701), but with a greater emphasis on depth in theory and algorithms. The course will also include additional advanced topics such as RKHS and representer theory, Bayesian nonparametrics, additional material on graphical models, manifolds and spectral graph theory, reinforcement learning and online learning, etc. Students entering the class are expected to have a pre-existing strong working knowledge of algorithms, linear algebra, probability, and statistics. If you are interested in this topic, but do not have the required background or are not planning to work on a PhD thesis with machine learning as the main focus, you might consider the general graduate Machine Learning course (10-701) or the Masters-level Machine Learning course (10-601).
Course Website: http://www.cs.cmu.edu/~epxing/Class/10715/
10-806 Foundations of Machine Learning and Data Science
Fall: 12 units
This course will cover fundamental topics in Machine Learning and Data Science, including powerful algorithms with provable guarantees for making sense of and generalizing from large amounts of data. The course will start by providing a basic arsenal of useful statistical and computational tools, including generalization guarantees, core algorithmic methods, and fundamental analysis models. We will examine questions such as: Under what conditions can we hope to meaningfully generalize from limited data? How can we best combine different kinds of information such as labeled and unlabeled data, leverage multiple related learning tasks, or leverage multiple types of features? What can we prove about algorithms for summarizing and making sense of massive datasets, especially under limited memory? We will also examine other important constraints and resources in data science including privacy, communication, and taking advantage of limited interaction. In addressing these and related questions we will make connections to statistics, algorithms, linear algebra, complexity theory, information theory, optimization, game theory, and empirical machine learning research. Topics to be covered will include:
- Fundamental measures of complexity for generalization, including VC-dimension and Rademacher complexity.
- Core algorithmic tools including boosting, regularization, and online optimization with connections to game theory.
- Spectral methods, streaming algorithms and other approaches for handling massive data.
- Foundations and algorithms for addressing important constraints or externalities such as privacy, limited memory, and communication constraints.
- Foundations for modern learning paradigms including semi-supervised learning, never-ending learning, interactive learning, and deep learning.
Course Website: http://www.cs.cmu.edu/~ninamf/courses/806/10-806-index.htm

Robots Courses

16-223 Introduction to Physical Computing
Fall: 10 units
Physical computing refers to the design and construction of physical systems that use a mix of software and hardware to sense and respond to the surrounding world. Such systems blend digital and physical processes into toys and gadgets, kinetic sculpture, functional sensing and assessment tools, mobile instruments, interactive wearables, and more. This is a project-based course that deals with all aspects of conceiving, designing and developing projects with physical computing: the application, the artifact, the computer-aided design environment, and the physical prototyping facilities. The course is organized around a series of practical hands-on exercises which introduce the fundamentals of circuits, embedded programming, sensor signal processing, simple mechanisms, actuation, and targeted behavior. The key objective is gaining an intuitive understanding of how information and energy move between the physical, electronic, and computational domains to create a desired behavior. The exercises provide building blocks for collaborative projects which utilize the essential skills and challenge students to not only consider how to make things, but also for whom we design, and why the making is worthwhile. This course is an IDEATe Portal Course for entry into either of the IDEATe Intelligent Environments or Physical Computing programs. CFA/DCT/TPK students can enroll under 16-223; CIT/MCS/SCS students can enroll in the 60-223 version of the course. Please note that there will be lab usage and materials fees associated with this course.

16-233 TBA
Intermittent: 10 units
TBA

16-264 Humanoids
Spring: 12 units
This course surveys perception, cognition, and movement in humans, humanoid robots, and humanoid graphical characters. Application areas include more human-like robots, video game characters, and interactive movie characters.
Course Website: http://www.cs.cmu.edu/~cga/humanoids-ugrad/

16-299 Introduction to Feedback Control Systems
Spring: 12 units
This course is designed as a first course in feedback control systems for computer science majors. Course topics include classical linear control theory (differential equations, Laplace transforms, feedback control), linear state-space methods (controllability/observability, pole placement, LQR), nonlinear systems theory, and an introduction to the role of control using computer learning techniques. Priorities will be given to computer science majors with robotics minor.
Prerequisites: 21-122 and 15-122

16-311 Introduction to Robotics
Spring: 12 units
This course presents an overview of robotics in practice and research with topics including vision, motion planning, mobile mechanisms, kinematics, inverse kinematics, and sensors. In course projects, students construct robots which are driven by a microcontroller, with each project reinforcing the basic principles developed in lectures. Students nominally work in teams of three: an electrical engineer, a mechanical engineer, and a computer scientist. This course will also expose students to some of the contemporary happenings in robotics, which includes current robot lab research, applications, robot contests and robots in the news.
Prerequisites: 24-311 or 21-260 or 21-241 or 21-240 or 16-202
Course Website: http://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/16311/www/current/

16-362 Mobile Robot Programming Laboratory
Fall: 12 units
This course is a comprehensive hands-on introduction to the concepts and basic algorithms needed to make a mobile robot function reliably and effectively. We will work in small groups with small robots that are controlled over wireless from your laptop computers. The robots are Neato household vacuum robots that have been converted to mini forklifts that can move pallets from place to place just like commercial automated guided vehicles do today. The robots are programmed in the modern MATLAB programming environment. It is a pretty easy language to learn, and a very powerful one for prototyping robotics algorithms. You will get a lot of experience in this course in addition to some theory. Lectures are focused on the content of the next lab. There is a lab every week and they build on each other so that a complete robot software system results. The course will culminate with a class-wide robot competition that tests the performance of all of your code implemented in the semester. In order to succeed in the course, students must have a 2nd year science/engineering level background in mathematics (matrices, vectors, coordinate systems) and have already mastered at least one procedural programming language like C or Java. When the course is over, you will have written a single software system that has been incrementally extended in functionality and regularly debugged throughout the semester.
Course Website: http://www.frc.ri.cmu.edu/~alonzo/teaching/16x52/16x62.html http://www.youtube.com/watch?v=skk0eGW5

16-371 Personalized Responsive Environments
Spring: 9 units
[IDEATe collaborative course]. Environmental factors have a significant impact on mood and productivity. Creating responsive environments necessitates the design of surroundings that are able to metamorphose in order to optimize user strengths and available resources and evolve in stride with user needs. This course will investigate the development of spaces that adapt to users' preferences, moods, and task specific needs, and the creation of adaptive environments that learn user preferences over time. Please note that there may be usage/materials fees associated with this course.
Prerequisites: 15-104 or 18-090 or 62-150 or 60-223

16-375 Robotics for Creative Practice
Fall: 10 units
[IDEATe collaborative course]. This project-oriented course brings art and engineering together into making machines which are surprisingly animate. Students will iterate their concepts through several small projects focused on using embodied behavior as a creative medium for storytelling, performance, and human interaction. Students will learn skills for designing, constructing and programming simple robot systems, and exploring their results through exhibition and performance. Technical topics include systems thinking, dynamic physical and computational behavior, autonomy, embedded programming, and fabrication and deployment. Discussion topics include both contemporary kinetic sculpture and robotics research. Please note that there may be usage/materials fees associated with this course.
Prerequisites: 60-223 or 15-104 or 16-222 or 62-150

16-384 Robot Kinematics and Dynamics
Fall: 12 units
Foundations and principles of robotic kinematics. Topics include transformations, forward kinematics, inverse kinematics, differential kinematics (jacobians), singularities, and dynamic equations of motion. Course also include programming on robot arms.
Prerequisites: 15-122 or 24-311 or 21-241 or 18-202 or 16-311

16-399 Introduction to Feedback Control Systems
Spring: 12 units
This course is designed as a first course in feedback control systems for computer science majors. Course topics include classical linear control theory (differential equations, Laplace transforms, feedback control), linear state-space methods (controllability/observability, pole placement, LQR), nonlinear systems theory, and an introduction to the role of control using computer learning techniques. Priorities will be given to computer science majors with robotics minor.
Prerequisites: 21-122 and 15-122
16-385 Computer Vision
Spring: 9 units
Basic concepts in machine vision, including sensing and perception, 2D image analysis, pattern classification, physics-based vision, stereo and motion, and object recognition.
Prerequisites: (15-122 and 18-202) or (21-259 and 15-122 and 21-241)

16-397 Art, Conflict and Technology in Northern Ireland
Spring: 12 units
Art, Conflict and Technology in Northern Ireland is a 12-unit course cross-listed between the School of Art, the Department of English, and the Robotics Institute. Throughout the term students will be introduced to a history of social strife in the North of Ireland from the 1960s to the present, and efforts to reconcile such differences in the contemporary period. We will consider the influence of advancing technology on how narratives are shared within a community and worldwide. We will reflect upon and analyze a variety of literary and visual art sources from the chosen time period, while also learning how to create mixed-media projects using Gigapan and Hear Me systems from Carnegie Mellon’s CREATE Lab in the Robotics Institute. If you have ever considered how artists explore societal strife through their writing or visual arts practice, if you are interested in the social and political influences of evolving technology, or if you are a practicing artist who uses advancing technology as a tool for individual expression, this integrative course is for you. Throughout the semester we will examine the practice of a range of visual artists that include Rita Duffy, John Kindness and Willie Doherty and writers and dramatists like Dermot Healey, Patrick McCabe, and Christina Reid. Students will learn how to use CREATE Lab’s Gigapan and Hear Me systems as platforms for exploring the content presented in the class for the development of final projects. We will travel to Belfast for spring break 2015, to meet a variety of writers and artists whose work we will study, and stakeholders in the reconciliation efforts throughout the region. In addition to weekly lectures on Thursdays throughout the term, students will have a six-week lab on Tuesdays. Lab sessions begin in the second week of classes (January 20).

16-421 Vision Sensors
Spring: 12 units
This course covers the fundamentals of vision cameras and other sensors - how they function, how they are built, and how to use them effectively. The course presents a journey through the fascinating five hundred year history of "camera-making" from the early 1500’s "camera obscura" through the advent of film and lenses, to today’s mirror-based and solid state devices (CCD, CMOS). The course includes a significant hands-on component where students learn how to use the sensors and understand, model and deal with the uncertainty (noise) in their measurements. While the first half of the course deals with conventional “single viewpoint” or “perspective” cameras, the second half of the course covers much more recent “multi-viewpoint” or “multi-perspective” cameras that includes a host of lenses and mirrors.
Prerequisites: 21-111 and 21-241
Course Website: http://www.cs.cmu.edu/~SLUCEY/Design.CV/

16-423 Designing Computer Vision Apps
Fall: 12 units
Computer vision is a discipline that attempts to extract information from images and videos. Nearly every smart device on the planet has a camera, and people are increasingly interested in how to develop apps that use computer vision to perform an ever expanding list of things including: 3D mapping, photo/image search, people/object tracking, augmented reality etc. This course is intended for students who are not familiar with computer vision, but want to come up to speed rapidly with the latest in environments, software tools and best practices for developing computer vision apps. No prior knowledge of computer vision or machine learning is required although a strong programming background is a must (at a minimum good knowledge of C/C++). Topics will include using conventional computer vision software tools (OpenCV, MATLAB toolboxes, VLFeat, CAFFE), and development on iOS devices using mobile vision libraries such as GPUImage and fast math libraries like Armadillo and Eigen. For consistency, all app development will be in iOS and it is expected that all students participating in the Intel-based MAC running OS X Mavericks or later. Although the coursework will be focussed on a single operating system, the knowledge gained from this class is intended to generalize to other mobile platforms such as Android etc.
Prerequisites: (15-213 and 21-240) or (15-213 and 21-241) or (15-213 and 18-202) or (21-259 and 15-213 and 18-202)

16-450 Robotics Systems Engineering
Fall: 12 units
Systems engineering examines methods of specifying, designing, analyzing and testing complex systems. In this course, principles and processes of systems engineering are introduced and applied to the development of robotic devices. The focus is on robotic system engineered to perform complex behavior. Such systems embed computing elements, integrate sensors and actuators, operate in a reliable and robust fashion, and demand rigorous engineering from conception through production. The course is organized as a progression through the systems engineering process of conceptualization, specification, design, and prototyping with consideration of verification and validation. Students completing this course will engineer a robotic system through its compete design and initial prototype. The project concept and teams can continue into the Spring-semester (16-474 Robotics Capstone) for system refinement, testing and demonstration.
Prerequisites: 16-311 and (16-299 or 24-451 or 18-370)

16-455 Human-Machine Virtuosity
Spring: 12 units
Human dexterous skill embodies a wealth of physical understanding which complements computer-based design and machine fabrication. This project-oriented course explores the duality between hand and machine through the practical development of innovative design and fabrication systems. These systems fluidly combine the expressivity and intuition of physical tools with the scalability and precision of the digital realm. Students will develop novel hybrid design and production workflows combining analog and digital processes to support the design and fabrication of their chosen projects. Specific skills covered include 3D scanning, 3D modeling (CAD), 3D printing (additive manufacturing), computer based sensing, and human-robot interaction design. Areas of interest include architecture, art, and product design.
Prerequisites: 16-223 or 60-223

16-456 Advanced Topics in Reality Computing: The Adaptive Home
Fall: 12 units
[IDEAtE collaborative course]. The Adaptive House is the focus of an advanced design studio based around the collaborative development of reality computing applications within a residential prototype. Reality computing encompasses a constellation of technologies focused around capturing reality (laser scanning, photogrammetry), working with spatial data (CAD, physical modeling, simulation), and using data to interact with and influence the physical world (augmented / virtual reality, projector systems, 3d printing, robotics). This studio will use reality computing to understand existing homes, define modes of augmentation, and influence the design of houses yet to be built through full scale prototyping. The objective of the course will be the production of a house that moves beyond the notion of being "smart", but is actively adapted towards its inhabitants? needs and abilities. Topics of special focus within the course are residential design (John Folan), augmented reality and robotics (Pyry Matikainen), and indoor flying robots (Manuela Veloso and Nina Barbuto). Please note that there may be usage/materials fees associated with this course.

16-461 Experimental Capture
Fall: 9 units
Performance capture is used in applications as varied as special effects in movies, animation, sports training, physical rehabilitation, and human-robot/human-computer interaction. This course will survey state-of-the-art techniques and emerging ideas, in the industry and in academia, to capture, model, and render human performances. The course will be a mix between lectures and discussion of recent progress in human motion capture and analysis. The course evaluation will be project-based, in which students will capture their own body and face motion, and build projects around the data they collect individually and as a group. We will cover: 1. Capture Techniques: We will describe and use various systems including motion capture, video-based capture, depth sensors, scanners, and eye-gaze trackers; 2. Modeling and Representation: We will cover classic and contemporary representations of face and body pose and motion, including statistical and physics-based techniques; 3. Rendering Applications: As new rendering paradigms emerge, new applications continue to develop. We will study recent progress in animation, synthesis, classification, and rehabilitation on new forms of displays. Please note that there may be usage/materials fees associated with this course.
Prerequisites: 15-365 or 60-422

Course Website: http://www.cs.cmu.edu/~SLUCEY/Design.CV/
16-465 Game Engine Programming  
Spring: 10 units  
This course is designed to help students understand, modify, and develop game engines. Game engines consist of reusable runtime and asset pipeline code. They provide game-relevant abstractions of low-level system services and libraries, making it easier to write bug-free games that work across multiple platforms. Game engines also handle artistic content, providing or integrating with authoring tools to ease the process of creating high-fidelity games. In this course, we will discuss the problems game engines attempt to solve, examine how current state-of-the-art engines address these problems, and create our own engines based on what we learn. We will cover both the content authoring and runtime aspects of engines. Coursework will consist of frequent, tightly-scoped programming and system design assignments; experiments through game engine source code; and two group projects — one in an engine created from scratch, and one that requires modification of an existing engine. Prerequisites: Students will be expected to be fluent in at least one programming language. We will be working with C++, Javascript, and a smattering of Python. We will be using git for version control and code sharing. The assignments in the course will be designed to be completed on an OS X or Linux workstation (e.g. the IDEaTe “virtual cluster”). Working with Windows will be possible, but might require extra effort. We will be building a 3D model pipeline around Blender, but no prior knowledge of the tool will be assumed.  
Prerequisites: 15-104 or 15-112 or 15-213 or 62-150

16-467 Human Robot Interaction  
Spring: 12 units  
The field of human-robot interaction (HRI) is fast becoming a significant area of research in robotics. The basic objective is to create and investigate interfaces that enable natural and effective modes of interaction with robotic technologies. HRI is highly interdisciplinary, bringing together methodologies and techniques from robotics, artificial intelligence, human-computer interaction, human factors, interaction design, psychology, anthropology, education, drama, and other fields. This course is primarily lecture-based, with in-class participatory mini-projects, group homework assignments, and a group term project that will enable students to put theory to practice using state-of-the-art interactive robots. The topics covered will include man-machine coupling, underlying robotic technologies, as they relate to human-robot interaction, interaction methodologies and techniques, the singularity, and will include significant discussion of application domains that feature HRI. This course has no prerequisites, but some basic familiarity with robots is recommended (programming knowledge is not necessary, but is useful for the term project).  
Course Website: http://www.cs.cmu.edu/~reids/16-467/

16-474 Robotics Capstone  
Spring: 12 units  
The Robotics Capstone course is intended to enhance the student’s capability of professional problem-solving and engineering design skills in robotics. Students taking this course will work in teams to complete a design project to resolve scientific and engineering issues relating to robots. Each team is required to design and build one autonomous robotic system and develop its control algorithm throughout the semester. The class will be composed of lectures and small group meetings. The lectures will discuss the design process and review several subjects of mechatronics (e.g. sensors, actuators, micro-controllers etc.) that will be useful to complete the project. During the weekly small group meetings, students will update their progress and receive feedback on the project. Team participation and communication is important and the presentation and reports must be technical and professional in structure. At the end of the semester, each team is required to make a public presentation of its robot and submit a report that provides clear definition of the problem being addressed, a methodology for the research, literature review, experimental results, conclusions based on findings, and recommendations for future work.  
16-597 Undergraduate Reading and Research  
Fall and Spring  
Missing Course Description - please contact the teaching department.

16-623 Designing Computer Vision Apps  
Fall: 12 units  
Computer vision is a discipline that attempts to extract information from images and videos. Nearly every smart device on the planet has a camera, and people are increasingly interested in how to develop apps that use computer vision to perform an ever expanding list of things including: 3D mapping, photo/image search, people/object tracking, augmented reality etc. This course is intended for students who are not familiar with computer vision, but want to come up to speed rapidly with the latest in environments, software tools and best practices for developing computer vision apps. No prior knowledge of computer vision or machine learning is required although a strong programming background is a must (at a minimum good knowledge of C/C++). Topics will include using conventional computer vision software tools (OpenCV, MATLAB toolboxes, VLFee, CAFFE), and development on iOS devices using mobile vision libraries such as GPUImage and fast math libraries like Armadillo and Eigen. For consistency, all app development will be in iOS and it is expected that all students participating in the class have access to an Intel-based MAC running OS X Mavericks or later. Although the coursework will be focussed on a single operating system, the knowledge gained from this class is intended to generalize to other mobile platforms such as Android etc. Strong programming background required.  
Course Website: http://www.cs.cmu.edu/~slucely/Design.CV

16-627 MSCV Seminar  
Fall  
(Open only to MSCV students.) MSCV students will be required to participate in this one-semester seminar course which will prepare them for the MSCV project starting in the Spring Semester. The first part of this course will cover talks by computer vision and related faculty about the ongoing research, development projects related to Computer Vision at CMU. The second part of this course will include student/faculty tutorial on topics such as OpenCV, Dataset Creation, Mechanical Turk etc. The goal of this series is to get students acquainted with practical knowledge for a successful project. In the last month of the course, each lecture will cover up to four possible MSCV projects pitched by faculty or industrial sponsors. At the end of the course students will turn in their choices, and a faculty committee will assign them the final projects.

16-720 Computer Vision  
Fall and Spring: 12 units  
This course introduces the fundamental techniques used in computer vision, that is, the analysis of patterns in visual images to reconstruct and understand the objects and scenes that generated them. Topics covered include image formation and representation, camera geometry, and calibration, computational imaging, multi-view geometry, stereo, 3D reconstruction from images, motion analysis, physics-based vision, image segmentation and object recognition. The material is based on graduate-level texts augmented with research papers, as appropriate. Evaluation is based on homeworks and a final project. The homeworks involve considerable Matlab programming exercises. Texts recommended but not required: Title: “Computer Vision Algorithms and Applications” Author: Richard Szeliski Series: Texts in Computer Science Publisher: Springer ISBN: 978-1-84996-913-1 Title: “Computer Vision: A Modern Approach” Authors: David Forsyth and Jean Ponce Publisher: Prentice Hall ISBN: 0-13-085198-1  
Course Website: http://www.andrew.cmu.edu/course/16-720/

16-721 Learning-based Methods in Vision  
Spring: 12 units  
A graduate seminar course in Computer Vision with emphasis on using large amounts of real data (images, video, textual annotations, user preferences, etc) to learn the structure of our visual world toward the ultimate goal of Image Understanding. We will be reading an eclectic mix of classic and recent papers on topics including: theories of perception, low-level vision (color, texture), mid-level vision (grouping and segmentation), object and scene recognition, image parsing, words and pictures models, image manifolds, etc. Prereqs: Graduate Computer Vision, 16-720  
Prerequisite: 16-720  
Course Website: http://www.cs.cmu.edu/~efros/courses/LBMV07/
Software Engineering Courses

17-400 Electronic Voting
Fall: 12 units
After the punched-card disaster in Florida in 2000, the U.S. has been rushing to replace old voting equipment with direct-recording electronic (DRE) machines (sometimes incorrectly lumped together as “touchscreens”). Recent examination of these machines by computer security experts has revealed significant security vulnerabilities, leading to a call by some computer scientists to either discontinue use of such machines or equip them with a printing device that would enable the voter to see a paper record of how she had voted before leaving the voting booth. This “voter-verifiable paper trail” idea has polarized the voting community, leading to bills in Congress and in some states to require it; but with vendors, election officials and public advocacy groups strongly in opposition. Each meeting will be devoted to a technical lecture followed by an hour of general discussion. The course is open to juniors, seniors and graduate students. Students from outside SCS are welcome. No advanced technical background is required except for some security and cryptography topics. Each student will participate in a team project, with a presentation to be made on the last day of the course. Grading will be based on class participation, the project paper and a final exam. There will be assigned readings but no midterm or written homework. This course counts as an elective in the Computation, Organizations and Society (COS) Ph.D. program. Topics include: Voting history and administration, vote buying, election rigging, punched cards, optical scanning, DRE machines, paper trails & Internet voting.
Course Website: http://euro.ecom.cmu.edu/program/courses/ctcr17-803

17-413 Software Engineering Reflection
Fall: 6 units
This course is an opportunity to reflect on a software engineering experience you have had in industry. It is structured as a writers workshop, in which you will work with the instructor and other students to identify and flesh out a software engineering theme that is illustrated by your industry experience. You will prepare a 10-page report on this theme, comparable to a practitioner’s report at a conference like ICSE or OOPSLA, and a 30-minute presentation to match. This course fulfills a requirement of the Software Engineering Minor program, but students in other programs may take the course if they meet the prerequisite industry experience and if space is available.

17-607 Predictable Professional Performance
Intermittent: 9 units
This course will focus on the practical and theoretical aspects of software engineering and the professional and technical skills required by software engineers. Students from outside SCS are welcome. No advanced technical background is required except for some security and cryptography topics. Each student developed interactively with the students, will reinforce the basic concepts from PSP and TSP, but will allow the student to further evaluate their processes and to use the processes in a disciplined team environment with a well understood data collection method. The course will culminate with the student evaluating their own performance as a team member and as a developer in a final report. This course will both reinforce and develop the concepts from PSP and TSP so no prerequisite for either of these courses is needed other than software programming skills producing standalone GUI applications in Java.

Course Website: http://euro.ecom.cmu.edu/program/courses/ctcr17-803
17-609 Global Software Development
Intermittent: 9 units
Software development is increasingly a globally-distributed undertaking. The search for talent across national boundaries and the integration of groups thrown together by mergers and acquisitions are but two of the many forces conspiring to fundamentally change the organizational context of software development. The skills that allow developers and managers to thrive in this milieu are among the most important in today’s development organizations. Distributed software development organizations are also receiving attention from researchers interested in communication, collaboration, and coordination over distances. Creating trust, awareness, shared understanding, and many other essentials of teamwork typically relies on face to face interaction. Creating effective technology-mediated mechanisms to support distributed teams requires a deep understanding of how individuals come together to form teams and organizations. This course covers a set of topics that are essential to both professionals who will become participants and leaders in globally-distributed projects, as well as researchers interested in studying virtual teams, distributed organizations, and global software development. Topics covered in this course will include: * Virtual teams, distributed organizations * Architectures and coordination * Distributed development environments * Lessons from open source * Open source ecologies * Challenges of culture * The outsourcing relationship * Facilitating trust, cooperation, social capital * Social networks and knowledge networks * Communication and awareness * Assessing coordination risk

17-615 Software Process Definition
Intermittent: 9 units
A software process definition is the cornerstone of implementing and improving a software process. The objective of this course is to prepare students to understand how processes work within the context of an operational, day-to-day engineering company, and most importantly how they can, as an individual within an engineering environment, change a process for the betterment of all. Although the focus is on software process, this course will be useful to all students who will be executing, improving, or defining most any type of process. An incremental methodology and modular approach to software process definition is used and covers: * guidelines for early success and building a sound foundation * organizing the process definition as it develops * approaches to avoid unnecessarily elaborate or formal notations * developing the process using organizational goals and constraints * using the environmental context that the process resides within and builds upon Although the focus is on software process, this course will be useful to all students who will be executing, improving, or defining most any type of process. Requirement: This course is intended for individuals who have operational software engineering experience or a comprehensive undergraduate coursework in software engineering.

17-619 Introduction to Real-Time Software and Systems
Intermittent: 12 units
Introduction to Real-Time Software and Systems presents an overview of time as it relates engineering complex systems. Any system that responds at the pace of relevant events has real-time constraints whether the timescale is short, like the flight controls for an aircraft, or longer, like the flight reservation system for an airline. Fundamental concepts, terminology, and issues of real-time systems are introduced in this course. The focus in this course is on software solutions to real-time problems—solutions that must be both correct and timely. Software development is examined with emphasis on real-time issues during each phase of the software lifecycle. Real-time requirements analysis, architecting real-time systems, designing and modeling system timing, and implementation and testing strategies are studied. Modeling techniques using UML 2.0 are applied. Particular emphasis is placed on real-time scheduling to achieve desired timing, reliability, and robustness. Languages and operating systems for real-time computing, and real-time problems in concurrent and distributed systems are explored. This course provides a comprehensive view of real-time systems with theory, techniques and methods for the practitioner. After successfully completing this course, the student will be able to identify constraints and understand real-time issues in system development, and propose approaches to typical real-time problems. The aim of this course is to motivate and prepare students to pursue more in-depth study of specific problems in real-time computing and systems development. REQUIREMENT: Proficiency with a high-level programming language such as C, C++, Ada and basic concepts of computing systems. Familiarity with software engineering concepts and system development lifecycle.

17-628 The Modern CIO
Fall: 6 units
In today’s competitive marketplace, technology can deliver the decisive edge as a company interfaces with its existing customers, engages prospective clients, and communicates across internal, local, national and global channels. Chief Information Officers of successful companies must possess skills that reach beyond the technical realm. Research indicates that competence in the areas of business acumen, communication, financial knowledge, project and board management far surpasses the need for technical competence. This course focuses on the non-technical aspects of ‘the corner office’. If the CTO is to be a true strategic business partner with others in the C-suite (CEO, COO, CFO), it is essential that he or she has a grasp not only on technology, but also on the company’s core philosophy, business strategy and drivers. This course will capitalize on the technical knowledge acquired in the Master of Software Engineering Professional Programs, and using it to target high-level job opportunities.

17-634 SE Elective
Intermittent
The goal of this course is to provide an understanding of electronics for students with formal backgrounds in computer science and software engineering to prepare them for work in domains where hardware and software are closely coupled. Example domains include: robotics, avionics, automotive, mobile devices, network switching systems, process/ environmental controls, and many others. Throughout the course, students will have opportunities to experiment with hardware and software in hands-on exercises that include electronics labs, robotics, process control, and others. Prerequisites: A hardware background is not required, but students should have a solid computer science background that includes languages, data structures, operating systems, and basic computer organization. Real-world project experience is preferred, although academic project work or internships will suffice. Undergraduates are welcomed but instructor approval is required (lattanzan@cs.cmu.edu).

17-635 Software Measurement
Fall: 9 units
Sections D and PP are NOT available for on-campus students. The purpose of this course is to introduce students to software measurement — from need identification through analysis and feedback and into the process. Much of the course material demonstrates concepts of software measurement that are used by managers and practitioners in industry today. The course is taught within the framework of the software engineering process. Required text: Selected Readings handed out in class

17-643 Hardware for Software Engineers
Intermittent
Programs, and using it to target high-level job opportunities. The search for talent across national boundaries and the integration of groups thrown together by mergers and acquisitions are but two of

17-643 Hardware for Software Engineers
Intermittent
The goal of this course is to provide an understanding of electronics beyond the average computer organization course. Its purpose is to enable software engineers to be more effective in domains where software and hardware are closely coupled. Examples of such domains include robotics, avionics, automotive, process control, and many others. Students successfully completing this course will be better prepared to communicate with hardware engineers, understand the hardware environment, and the subtle regions where software and hardware meet. Requirement: Students need not have a hardware background, but they should have a solid computer science background including languages, data structures, discrete math, operating systems, and computer organization. It is highly desirable that students have project experience, preferably real-world experience, although project course work and/or internships will suffice. Undergraduates need instructor approval (lattanzan@cs.cmu.edu).

17-690 System Development Lifecycle
Intermittent: 9 units
The search for talent across national boundaries and the integration of groups thrown together by mergers and acquisitions are but two of the many forces conspiring to fundamentally change the organizational context of software development. The skills that allow developers and managers to thrive in this milieu are among the most important in today’s development organizations. Distributed software development organizations are also receiving attention from researchers interested in communication, collaboration, and coordination over distances. Creating trust, awareness, shared understanding, and many other essentials of teamwork typically relies on face to face interaction. Creating effective technology-mediated mechanisms to support distributed teams requires a deep understanding of how individuals come together to form teams and organizations. This course covers a set of topics that are essential to both professionals who will become participants and leaders in globally-distributed projects, as well as researchers interested in studying virtual teams, distributed organizations, and global software development. Topics covered in this course will include: * Virtual teams, distributed organizations * Architectures and coordination * Distributed development environments * Lessons from open source * Open source ecologies * Challenges of culture * The outsourcing relationship * Facilitating trust, cooperation, social capital * Social networks and knowledge networks * Communication and awareness * Assessing coordination risk

Carnegie Mellon University
17-648 Engineering Data Intensive Scalable Systems  
Intermittent: 12 units  
Internet services companies such as Google, Yahoo!, Amazon, and Facebook have pioneered systems that have achieved unprecedented scale while still providing high level availability and a high cost-performance. These systems differ from mainstream high performance systems in fundamental ways. They are data intensive rather than compute intensive as we see with mainstream super computers spending the bulk of their time performing data I/O and manipulation rather than computation. They need to inherently support scalability, typically having high reliability and availability demands as well. Given that they often operate in the commercial space the cost-performance of these systems needs to be such that the organizations dependent on such systems can turn a profit. Designing and building these systems require a specialized set of skills. This course will cover the set of topics needed in order to design and build data intensive scalable systems. In this domain engineers not only need to know how to architect systems that are inherently scalable, but to do so in a way that also supports high availability, reliability, and performance. Given the large distributed nature of these systems basic distributed systems concepts such as consistency and time and synchronization are also important. These systems largely operate around the clock, placing an emphasis on operational concerns. This course will introduce students to these concerns with the intent that they understand the extent to which things like deploying, monitoring, and upgrading impact the design. The course will be a hands-on project oriented course. The basic concepts will be given during the lectures and applied in the project. The students will gain exposure to the core concepts needed to design and build such systems as well as current technologies in this space. Class size will be limited.

17-651 Models of Software Systems  
Fall: 12 units  
Scientific foundations for software engineering depend on the use of precise, abstract models for describing and reasoning about properties of software systems. This course considers a variety of standard models for representing sequential and concurrent systems, such as state machines, algebraic and process models, and temporal logic; and invariant, non-determinism, and inductive definitions are recurrent themes throughout the course. After completing this course, students will: 1. Understand the strengths and weaknesses of certain models and logics including state machines, algebraic and process models, and temporal logic; 2. Be able to select and describe appropriate abstract formal models for certain classes of systems, describe abstraction relations between different levels of description, and reason about the correctness of refinements; 3. Be able to prove elementary properties about systems described by the models introduced in the course; and 4. Understand some of the strengths and weaknesses of formal automated reasoning tools. Prerequisites: Undergraduate discrete math including first-order logic, sets, functions, relations, and simple proof techniques such as induction. Sections D, PP and G are NOT available for on-campus students. Admission to the class is by approval from the instructor. If you are not MSE/MITSE/SE/MITS, send email to garlan@cs.cmu.edu for permission to enroll. The email should briefly describe your background, whether you have taken an undergraduate discrete math course, and why you would like to take the course. The course must be taken for a letter grade (not pass/fail). This is a graduate level course.  
Course Website: http://www.cs.cmu.edu/afs/cs.cmu.edu academic/ class/17651-f01/www/

17-652 Methods: Deciding What to Design  
Fall: 12 units  
Sections D and PP are NOT available for on-campus students. Practical development of software requires an understanding of successful methods for bridging the gap between a problem to be solved and a working software system. In this course you will study a variety of ways of understanding the problem to be solved by the system you're developing and of framing an appropriate solution to that problem. After completing this course, you will be able to: identify different classes of problems and their structures; analyze technical, organizational, usability, business, and marketing constraints on solutions; apply engineering approaches to frame solutions; understand how your understanding of the problem should be reflected in the software design. PREREQUISITE: minimum of 3 mos hands-on software development experience in industry. Students not accepted into the MSE program must present a current resume showing relevant industrial experience to department coordinator. This course is offered to only MSE/MITSE and MSIT-SE students for fall semester. This course is for graduate students only. This course is for letter grade only (no pass/fail grades). To register for Methods course, you will need a requirement of a minimum of 3 mos hands-on software development experience in industry. Please submit a statement that gives the company, the dates, and a sentence or two about what you were actually doing during that time (i.e. programming, testing, other things actually involved in software development) to jdh@cs.cmu.edu. This is a graduate course. Only undergrad SE minors may take this course.  
Course Website: http://spoke.compose.cs.cmu.edu/methods-fall-05/res/bib.htm

17-653 Managing Software Development  
Fall: 12 units  
Sections D, F, PP and G are NOT available for on-campus students. Large scale software development requires the ability to manage resources - both human and computational - through control of the development process. This course provides the knowledge and skills necessary to lead a project team, understand the relationship of software development to overall product engineering, estimate time and costs, and understand the software process. After completing this course, students will: 1. be able to write a software project management plan, addressing issues of risk analysis, schedule, costs, team organization, resources, and technical approach 2. be able to define the key process areas of the Capability Maturity Model and the technology and practices associated with each and a variety of software development life cycle models and explain the strengths, weaknesses, and applicability of each 3. understand the relationship between software products and overall products (if embedded), or the role of the product in the organizational product line 4. understand the legal issues involved in liability, warranty, patentability, and copyright 5. understand the purpose and limitations of software development standards and be able to apply sensible tailoring where needed 6. be able to use software development standards for documentation and implementation 7. be able to apply leadership principles 8. be able to perform requirements elicitation REQUIREMENT: Students must have had industrial software engineering experience with a large project, or a comprehensive undergraduate course in software engineering. This course is for letter grade only. Contact the instructor (droot@cs.cmu.edu) for permission to join the class. This is a course for graduate students. Only undergrad SE minors may take this course.
17-654 Analysis of Software Artifacts
Spring: 12 units
Analysis is the systematic examination of an artifact to determine its properties. This course will focus on analysis of software artifacts—primarily code, but also including analysis of design, architectures, and test suites. We will focus on functional properties, but also cover non-functional properties like performance and security. In order to illustrate core analysis concepts in some depth, the course will center on static program analysis; however, the course will also include a breadth of techniques such as testing, model checking, theorem proving, dynamic analysis, and type systems. The course will balance theoretical discussions with lab exercises in which students will apply the ideas they are learning to real artifacts. After completing this course, students will: * know what kinds of analyses are available and how to use them * understand their scope and power, when they can be applied and what conclusions can be drawn from their results * have a grasp of fundamental notions sufficient to evaluate new kinds of analysis when they are developed * have some experience selecting and writing analyses for a real piece of software, applying them and interpreting the results Ph.D. students taking the 17-754 version of the course will gain a broad overview of the analysis research literature and in-depth knowledge of a particular subarea through a course project. Requirement: A recent discrete math course and programming experience. Strongly Recommended: Models of SW Development course (17-651) before taking this course. This course is for letter grade only (no pass/fail grades). This is a graduate course. Only undergrad SE minors may take this course with the instructor's permission. Course Website: http://www-2.cs.cmu.edu/~aldrich/courses/654/

17-655 Architectures for Software Systems
Spring: 12 units
Successful design of complex software systems requires the ability to describe, evaluate, and create systems at an architectural level of abstraction. This course introduces architectural design of complex software systems. The course considers commonly-used software system structures, techniques for designing and implementing these structures, models and formal notations for characterizing and reasoning about architectures, tools for generating specific instances of an architecture, and case studies of actual system architectures. It teaches the skills and background students need to evaluate the architectures of existing systems and to design new systems in principled ways using well-founded architectural paradigms. After completing this course, students will be able to: 1. describe an architecture accurately 2. recognize major architectural styles in existing software systems 3. generate architectural alternatives for a problem and choose among them 4. construct a medium-sized software system that satisfies an architectural specification 5. use existing definitions and development tools to expedite such tasks 6. understand the formal definition of a number of architectures and be able to reason about the properties of those architectures 7. use domain knowledge to specialize an architecture for a particular family of applications REQUIREMENT: Experience with at least one large software system, either through industrial software development experience or an undergraduate course in software engineering, compilers, operating sys., or the like. This course is for letter grade only. Instructor wants each student who wants to take his Architectures class to have worked on a significant software system in your courses or in industry. Write to garlan@cs.cmu.edu for admission into the course stating the experience you have had. This is a graduate course. Only undergrad SE minors may take this course

17-664 Enterprise Application Integration
Intermittent: 12 units
Modern business enterprises are supported by hundreds of disparate applications that work together to achieve a common goal. These applications are typically large three-tier (or n-tier) application silos developed to support a particular facet of business. Unfortunately, in most cases these systems were never designed to have their services reused transparently across an organization. The result is often a lack of flexibility for large-scale reuse, lack of reliability when integrated, inappropriate security models for organization-level integration, and others. Enterprise Application Integration is a graduate-level course on how to design and deploy large-scale systems for supporting the critical backbone of an organization. Although a particular emphasis is put on Service Oriented Architecture (SOA) and Enterprise Service Bus (ESB), a broad set of topics is covered. These include security for large-enterprise systems, reliability for distributed long-running transactions, standards for intra- and extra-organization system integration, deployment and fault-tolerance of systems, and others. The aim of the course is to prepare future software architects to pragmatically deal with large-scale systems, so as to understand the trade-offs and implications of supporting the goals of an organization. PP section for Portugal students only.

17-690 Seminar in Software Process
Intermittent
The Seminar in Software Process course is a self-study and discussion course. Discussions center on how to implement effective and efficient software processes. The focus of the seminar is on systematic approaches to building software better, faster, and cheaper. A variety of process improvement and quality management strategies are discussed, including Total Quality Management, Baldrige Award, ISO 9001, ISO 15504 (SPICE), and others, but the course is primarily structured around the staged approach to improvement from the Capability Maturity Model (CMM). Specific topics that are covered include software project management, configuration management, quality assurance, organizational learning, process definition, training, peer reviews, team building, change management, measurement, and statistical process control. These topics are addressed from a process management perspective. Required Textbook: M.B. Chirsiss, M.D. Konrad, and S. Shrum, “CMMI: Guidelines for Process Integration and Product Improvement, Second Edition.” Addison-Wesley, 2006, or you can use the online SEI technical reports, which are free. Note that the third edition of this book is planned for March 2011. This is a graduate course. Only undergrad SE minors may take this course.

17-803 Electronic Voting
Fall: 12 units
After the punched-card disaster in Florida in 2000, the U.S. has been rushing to replace old voting equipment with direct-recording electronic (DRE) machines (sometimes incorrectly lumped together as “touchscreens”). Recent examination of these machines by computer security experts has revealed significant security vulnerabilities, leading to a call by some computer scientists to either discontinue use of such machines or equip them with a printing device that would enable the voter to see a paper record of how she had voted before leaving the voting booth. This “voter-verifiable paper trail” idea has polarized the voting community, leading to bills in Congress and in some states to require it but with vendors, election officials and public advocacy groups strongly in opposition. Each meeting will be devoted to a technical lecture followed by an hour of general discussion. The course is open to juniors, seniors and graduate students. Students from outside SCS are welcome. No advanced technical background is required except for some security and cryptography topics. Each student will participate in a team project, with a presentation to be made on the last day of the course. Grading will be based on class participation, the project paper and a final exam. There will be assigned readings but no midterm or written homework. This course counts as an elective in the Computation, Organizations and Society (COS) Ph.D. program. Topics include: Voting history and administration, vote buying, election rigging, punched cards, optical scanning, DRE machines, paper trails & Internet voting Course Website: http://euro.ecom.cmu.edu/program/courses/cr17-803