EEL 6935 Software-defined Computing Systems

1. *Catalog Description (including credit hours)*: Hardware and software models, systems software and software platforms, optimization techniques, predictive modeling, experiment design, adaptive and feedback-based computing, monitoring techniques and applications in networking, cluster systems, cloud computing, storage, data systems, systems management and datacenters. 3 credits.
2. *Pre-requisites and Co-requisites*: introduction to programming or data structures and algorithms (EEL4834 or equivalent) and principles of computer systems design (EEL5737 or equivalent) or instructor approval.
3. *Course objectives*: Large scale computer and software systems are increasingly complex, diverse and outsourced to computational clouds and other large infrastructures. The cooperative management of these systems by either their owners or their providers can only be done effectively and efficiently if programmatic interfaces are exposed for that purpose. This requires computer systems designers to understand a variety of fundamental modeling concepts and management techniques spanning multiple system layers, for both the systems to be designed and the components and services used to build these systems. This course introduces these models and techniques to students, by way of illustrative cases of software-defined systems deployed in industry and academia for a variety of computer, storage and networking applications.
4. *Textbooks and Software Required*
   None. Instructor will provide papers and access to software and resources.
5. *Recommended Reading*:
   a. Recent conference papers and online resources/documents provided by the instructor
6. Course Outline (provide topics covered by week or by class period):
   - Introduction
     - Software-definition versus virtualization and IT-as-a-Service
     - Consumer versus provider perspectives
   - Software-defined networking
     - Openflow and related concepts
   - Software-defined execution environments
     - Definition and scope
   - Examples
   - Abstractions and modeling techniques
     - Abstractions of resources, workloads and service-level agreements
     - Software and hardware development, structural and behavioral models
     - Predictive modeling (linear models, regression, least square, K-nearest neighbor, random forests, etc)
   - Project
   - Software-defined infrastructure
     - Software-defined datacenters
     - Software-defined storage
   - Management, monitoring and operational aspects of software-defined systems
   - Language, runtime and workload orchestration for software-defined systems
   - Other case studies: software-defined file systems, caching, topology, etc