EECE 636 – Analysis and Verification of Software (3 credits)

Catalog description:

This course introduces the basics needed to understand automation techniques for the analysis and verification of computing systems including logics behind programming languages. We will present tools for automated analysis that improve the reliability and correctness of software that reflect state of the art design and validation techniques that are changing the way software is designed and implemented today. The students will have the chance to practice and possibly advance these techniques in small projects.

Areas: Software engineering, verification

Required or Elective:

Elective for CCE / ECE
Level: undergraduate or graduate standing

Prerequisites:

<u>By topic:</u> Students are expected to have basic knowledge of data structures and algorithms and considerable programming experience.

Textbooks:

The material of the course will be picked from publications in the field and the textbooks should serve as background references.

References:

- Model Checking by Edmund M. Clarke, Orna Grumberg and Doron A. Peled. ISBN: 0262032708.
- Software Abstractions: Logic, Language, and Analysis by Daniel Jackson. ISBN: 0262101149.
- A Practitioners Guide to Software Test Design by Lee Copeland. ISBN: 158053791X.

Course objectives:

The objectives of this course are to give students:	***Correlates to Program Educational Objectives***
Knowledge of the relation between logic systems and programming languages, ability to understand the mathematical foundation of programming languages and computing systems, and skill set needed to formally express, analyze and automate the analysis of correctness properties of computing systems.	**1,2,4**
Advanced knowledge and application of modern team work methodologies in software engineering and their relation to verification.	**1,3,4**

State of the art knowledge of proof and model checking systems.	**1,2,3**
Experience in building, writing and presenting publication quality systems and papers.	**2,3,4**

Topics

No.	**Subjects covered**	**75 Min. Lectures**
1	Dynamic verification—testing	1
2	Test generation and regression testing	1
3	Static verification—formal reasoning	3
4	Logic and programming languages	3
5	Logic solvers—SAT, BDD, SMT	3
6	Model checkers—Alloy, CBMC	2
7	Theorem Proving – ACL2	2
8	Abstraction techniques	2
9	Analysis of concurrent programs	2
10	Temporal Logics	2
11	Design by contract	2
12	Software quality metrics	2
13	Project Presentations	4

Class/laboratory schedule

a) Two 75-minute lectures per week.

b) Use of computer lab or personal computer is needed for working on the projects.

Course outcomes:

At the end of the course students should be able to:	***Correlates to Program Outcomes***		
	H	**M**	**L**
1. Formally express computing systems	**a, e, g,**	**m.n**	**k,h**
2. Formally express complex properties of computing systems	**a,e,g**	**m,n**	**k,h**
3. Use formal methods to verify computing systems	**A,e,k**	**m,n**	
4. Find and resolve bugs and problems in computing systems	**a, b, e**		**K**
5. Use common design patterns to solve computing problems	**a, b, j**	**l**	**K**
6. Use common design patterns to facilitate verification of computing systems	A,b,j	l	
7. Use automated verification techniques	**a, c, k**	**l**	
8. Augment and design automated verification techniques	**A,e,j**		**K**
9. Write and present publication quality verification	**g,f**	**l**	**k**

projects.			

Resources for the course:

Recent arctiles, publications, and online material

Evaluation:

1. Class participation: 10 %
2. Homework, assignments and projects: 50 %
3. Exams: 40 %

Students will submit three assignments one project and one paper. The assignments involve formal descriptions of software designs and their correctness properties. The project will be a case study of a computation system or an improvement on one of the tools and techniques studied in the class. The paper will be a report about the project if the project was a novel or an improvement of a verification technique. Otherwise it can be a comparison between two contemporary papers from the literature. Students will have one exam that will test their understanding of the basic concepts taught in the class.

Professional component:

Engineering topics: 75%
General education: 5%
Mathematics and basic sciences: 20%

Computer usage:

Students pick their platforms and programming languages. Linux, C/C++, Java are recommended.

Person(s) who prepared this description and date of preparation:

Fadi Zaraket, Nov 2009