**Course Information Handout**

**ECE 60872/CS 590 – Fault-Tolerant Computer System Design**

**Purdue University**

**Fall 2017**

The course provides an introduction to the hardware and software methodologies for specifying, modeling and designing fault-tolerant systems supported by case studies of real systems. The material presents a broad spectrum of hardware and software error detection and recovery techniques that can be used to build reliable networked systems. The lectures discuss how the hardware and software techniques interplay, what techniques can be provided in COTS hardware, what can be embedded into operating system and network communication layers, and what can be provided via a distributed software layer and in the application itself.

The course focuses on hands-on learning through the design and development of innovative systems in the course project, which carries 50% of the weightage for the course grade. This is reinforced by two lectures given by practitioners from the industry who share their experiences and insights in building dependable systems.

We use a modeling software called UltraSAN to model a realistic system and solve the model to evaluate various dependability properties of the system.

*Note: This is* ***not*** *an advanced graduate level course. Any student with a strong undergraduate CS or ECE background, i.e., one who is able to program in at least one high level programming language and has a basic knowledge of probability can take the class.*

**Class hours:** Monday, Wednesday, and Friday 10.30-11.20 am, EE 226

**Instructor:** Prof. Saurabh Bagchi, Professor, School of Electrical and Computer Engineering (ECE) and Department of Computer Science (CS). In addition, there will be 2 guest lectures by practitioners from the industry.

**Office, Phone, Email:** EE 329, 765-494-3362 (Office), sbagchi@purdue.edu

**Office hours:** Tuesday and Friday 3-4

**Administrative Assistant:** Mary-Ann Saterfield, msaterfi@purdue.edu, EE 326B, 494-6389

**Graduate Course Assistants:** Ran Xu (xu943@purdue.edu) and Christopher Wright (wrigh338@purdue.edu)

They are available to help with programming questions on the programming assignments and projects. However, they are not available to code for you (you wish!).

**URL:** https://engineering.purdue.edu/ee695b/

**Textbook:** No text book.

**Reference Books:** (No need to buy since only parts of each will be used and I will provide photocopies of relevant portions.)

1. I. Koren and C. Mani Krishna, *Fault-tolerant Systems*, 1st edition, 2007, Morgan Kaufmann.
2. D. P. Siewiorek and R. S. Swarz, *Reliable Computer Systems - Design and Evaluation*, 3rd edition, 1998, A.K. Peters, Limited.
3. D. K. Pradhan, ed., *Fault Tolerant Computer System Design*, 1st edition, 1996, Prentice-Hall.
4. K. Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications*, 2nd edition, 2001, John Wiley & Sons.

Apart from these, the course will use technical conference and journal papers. You are expected to get the papers from IEEExplore or ACM Digital Library.

**Course Structure:**

*Class project*: There will be separate research projects that each team of 2 or 3 students will work on. Each project will focus on one aspect of fault-tolerant system design and will test the ability to design, model or implement, execute experiments and perform evaluation. The target will be to produce work that can be sent for a conference publication, which has happened with many projects in the past. Being graduate students, it is in the best interest of your career to build your publication record.

There will be the following phases in the project, each with their tentative timeline.

|  |  |
| --- | --- |
| List of suggested projects made available | September 5 |
| Project teams formed, discussion of project ideas with instructor | September 5-12 |
| Project proposals submitted | September 12 |
| Interim project presentations (15 minutes each group) | October 12, 14 |
| Preliminary project report | October 17 |
| Final project presentations | Last two days of class |
| Final project report | Last day of semester |

*Exams*: There will be a mid-term and a final exam. Each exam will be open book, open notes, open computer. The mid-term exam will be a 1 hour exam. The final exam will be comprehensive.

*Homeworks*: There will be three homeworks – two written and one programming-based. The programming-based homework will introduce a widely-used system modeling tool called UltraSAN. You will use it to model a realistic system and solve the model to determine the dependability characteristics of the system. This will give you valuable exposure to how you can evaluate a system by modeling its relevant parts.

*Active Learning Activity:* We will have activities in-class where you solve problems based on material covered in the previous week’s lectures. Some of these would be individual and some would be group-based.

*Dependability in the News*: The class will read articles about dependability issues in the news and will provide analysis of these, including probable cause of the incidents and possible prevention or remediation actions. This will highlight the connections between the fundamental techniques we learn and their applications in the real world.

**Grade Allocation:**

Course project: 50%

Mid-term: 15%

Final: 20%

Homeworks: 15%

For reference, the class performances in the last few offerings of this course were as follows. Fall 2016: 4 A+, 12 A, 1 A-; Fall 2015: 3 A+, 18 A, 2 A-; Fall 2014: 3 A+, 16 A, 1 A-, 1 B-.

**Lecture Outline**

This is the tentative outline of coverage of topics in the class.

|  |  |
| --- | --- |
| Introduction: Motivation, System view of high availability design, Terminology | 2 |
| Stochastic analysis of reliability   * Discrete distributions * Continuous distributions | 6 |
| Hardware redundancy: Basic approaches, Static & Dynamic, Voting, Coding for detection and recovery          *Application*: SEC-DED codes | 3 |
| Error detection and correction techniques: Watchdog processors, Heartbeats, Consistency and capability checking, Data audits, Assertions, Control-flow checking          *Application*: Erasure-coded storage | 3 |
| Software fault tolerance: Process pairs, Robust data structures, N version programming, Recovery blocks, Replica consistency & reintegration, Multithreaded programs          *Application*: Quantitative evaluation of NVP and RB | 5 |
| Secure coding practices: Principles and practice   * *Application*: Coding examples | 2 |
| Network fault tolerance: Reliable communication protocols, Agreement protocols, Byzantine fault tolerance          *Application*: Bitcoin | 6 |
| Modeling          *Application*: UltraSAN, Sharpe | 4 |
| Checkpointing & Recovery          *Application*: SCR checkpointing system for DOE supercomputers | 4 |
| Experimental Evaluation: Simulation and Fault-injection based | 2 |
| Practical Systems for Fault Tolerance: Putting it all together          *Application*: Amazon Web Service          *Application*: Hadoop | 2 |
| Industry presentations | 2 |
| Discussion of projects | 2 |
| Tests | 1 |
| **Total** | **44** |