

# CS 310: Software Engineering II

## Marshall University, Spring 2007

### Contents

|          |                                                 |          |
|----------|-------------------------------------------------|----------|
| <b>1</b> | <b>Course Description</b>                       | <b>3</b> |
| <b>2</b> | <b>Prerequisites</b>                            | <b>3</b> |
| <b>3</b> | <b>Class meeting time and location</b>          | <b>3</b> |
| <b>4</b> | <b>Instructor information</b>                   | <b>3</b> |
| <b>5</b> | <b>Course topics at a glance</b>                | <b>3</b> |
| <b>6</b> | <b>Detailed description of course topics</b>    | <b>4</b> |
| 6.1      | Software design module . . . . .                | 4        |
| 6.1.1    | Object-oriented design . . . . .                | 4        |
| 6.1.2    | Real-time software design . . . . .             | 4        |
| 6.1.3    | User interface design . . . . .                 | 4        |
| 6.2      | Software development module . . . . .           | 4        |
| 6.2.1    | Rapid software development . . . . .            | 4        |
| 6.2.2    | Software reuse . . . . .                        | 4        |
| 6.2.3    | Component-based software engineering . . . . .  | 5        |
| 6.2.4    | Critical systems development . . . . .          | 5        |
| 6.2.5    | Software evolution . . . . .                    | 5        |
| 6.3      | Verification and validation module . . . . .    | 5        |
| 6.3.1    | Verification and validation . . . . .           | 5        |
| 6.3.2    | Software testing . . . . .                      | 6        |
| 6.3.3    | Critical systems validation . . . . .           | 6        |
| 6.4      | Management module . . . . .                     | 6        |
| 6.4.1    | Managing people . . . . .                       | 6        |
| 6.4.2    | Quality management . . . . .                    | 6        |
| 6.4.3    | Process improvement . . . . .                   | 6        |
| 6.4.4    | Configuration management . . . . .              | 7        |
| 6.5      | Emerging technologies module . . . . .          | 7        |
| 6.5.1    | Security engineering . . . . .                  | 7        |
| 6.5.2    | Service-oriented software engineering . . . . . | 7        |
| 6.5.3    | Aspect-oriented software development . . . . .  | 7        |

|          |                                |          |
|----------|--------------------------------|----------|
| <b>7</b> | <b>Course assessment</b>       | <b>8</b> |
| 7.1      | Written Assignments . . . . .  | 8        |
| 7.2      | Team Project . . . . .         | 8        |
| <b>8</b> | <b>Instructional materials</b> | <b>8</b> |
| <b>9</b> | <b>WebCT Vista</b>             | <b>8</b> |

## 1 Course Description

Software engineering is an *engineering discipline* which encompasses all aspects of software production from requirements elicitation, system specification, design, implementation, testing, deployment, and maintenance. In this course, you will learn about software design, software development environments, software build tools, people and process management, version control and release management, software testing, and emerging technologies. Due to the broad nature of the subject area, this course will place emphasis on the breadth of the subject matter rather than on an in-depth study of few topics.

## 2 Prerequisites

- CS 305: Software Engineering I with a grade of C or better.
- Access to WebCT Vista from your home will be quite helpful.

## 3 Class meeting time and location

- 11.00 AM – 12.15 PM, Tuesdays and Thursdays, GH 211.

## 4 Instructor information

**Name:** Dr. Venkat N. Gudivada, Associate Professor, Engineering & Computer Science.

**Phone and email:** 304-696-5452; gudivada@marshall.edu. Please use WebCT Vista email for course related inquiries.

**Office Location:** GH Room 205A.

**Office Hours:** MW 2.30 PM - 3.30 PM and TuTh 1.30 PM - 3.30 PM. Other times by appointment.

## 5 Course topics at a glance

- Software design
- Design patterns
- Software development environments
- Software reuse
- People and process management
- Versioning and configuration management
- Quality assurance and testing
- Measurement and maintenance
- Emerging technologies

## 6 Detailed description of course topics

### 6.1 Software design module

#### 6.1.1 Object-oriented design

1. What are the *important activities* in a general object-oriented design process?
2. Explain different *models for documenting* an object-oriented design.
3. How do you use *UML* for representing design artifacts?

#### 6.1.2 Real-time software design

1. How are real-time systems usually implemented as a set of *concurrent processes*?
2. Explain the role of a *real-time operating system*.
3. Describe a *generic process architecture* for monitoring and control systems, and data acquisition systems.

#### 6.1.3 User interface design

1. Explain primary user *interface design principles*.
2. Explain different *user interaction styles* and list contexts in which each one is more appropriate.
3. What are the principal activities in the *user interface design process*?
4. List *usability attributes* and explain different approaches to interface evaluation.

### 6.2 Software development module

The goal of this module is to answer the following questions:

#### 6.2.1 Rapid software development

1. Discuss how an *iterative* and *incremental* software development approach leads to faster delivery of more useful software
2. What are *agile* development methods? How are they different from the traditional methods?
3. Explain the principles, practices, and limitations of *extreme programming*.
4. Discuss how *prototyping* can be used to help resolve requirements and design uncertainties

#### 6.2.2 Software reuse

1. Discuss benefits and *problems* of reusing software.
2. Explain ways to implement *software reuse*.

3. Describe *concept reuse* and explain how concepts can be represented as *patterns*.
4. Discuss how systems can be developed by *composing off-the-shelf applications*.
5. Explain means to realize *software product lines*.

### 6.2.3 Component-based software engineering

1. What is *component-based software engineering* (CBSE)?
2. What is a *component model*?
3. What are the principal activities in *CBSE process*? Why do you have to make *requirements compromises* in CBSE?
4. What problems arise during the process of *component integration*?

### 6.2.4 Critical systems development

1. How does fault avoidance and *fault tolerance* contribute to *dependable systems*?
2. What are the principal activities in *dependable software processes*?
3. Explain programming techniques for *fault avoidance*.
4. Discuss ways in which *diversity and redundancy* are used in fault-tolerant architectures?

### 6.2.5 Software evolution

1. List different types of *software maintenance*.
2. What factors affect *software maintenance costs*?
3. What processes are involved in *software re-engineering*.
4. How do you *assess legacy systems* to decide whether they should be maintained or replaced?

## 6.3 Verification and validation module

The goal of this module is to answer the following questions:

### 6.3.1 Verification and validation

1. What are the distinctions between *software verification* and *software validation*?
2. Explain how *program inspection* is used as a method of discovering defects in programs.
3. How is *automated static analysis* used in verification and validation?
4. Explain how *static verification* is used in the Cleanroom development process.

### 6.3.2 Software testing

1. Distinguish between *validation testing* and *defect testing*.
2. Explain the principles of *system testing* and *component testing*.
3. Describe three stages that may be used for *generating system test cases*.
4. Discuss essential characteristics of software tools that support *test automation*.

### 6.3.3 Critical systems validation

1. How do you *measure reliability* of a software system?
2. Explain how *reliability growth models* can be used to predict when a required level of reliability has been achieved.
3. Explain principles of *safety arguments*.
4. What are the *problems in assuring* the security of a system?
5. How do *safety cases* are used to present arguments and evidence of a system safety?

## 6.4 Management module

The goal of this module is to answer the following questions:

### 6.4.1 Managing people

1. What are the issues involved in *selecting and retaining staff* in a software development organization?
2. What factors influence *individual motivation* and discuss their *implication* for software project managers?
3. What are key issues of team working?
4. Describe the structure of People Capability Maturity Model.

### 6.4.2 Quality management

1. Explain *quality management process*.
2. Why *standards* are important in the quality management process?
3. What are *software metrics*? What are the differences between *predictor metrics* and *control metrics*?
4. How are *measurements* used in assessing some software quality attributes?
5. Discuss *current limitations* of software measurement.

### 6.4.3 Process improvement

1. Why software *process improvement* is worthwhile?

2. Discuss how *software process factors* influence software quality and the productivity of developers.
3. Describe simple models for *software processes*.
4. What is *process capability* and *process maturity*?
5. Explain the general form of the *CMMI model* for process improvement.

#### 6.4.4 Configuration management

1. Why software *configuration management* is required for complex software systems?
2. Discuss *four fundamental* configuration management activities?
3. How are *CASE tools* used for configuration management?

### 6.5 Emerging technologies module

The goal of this module is to answer the following questions:

#### 6.5.1 Security engineering

1. Discuss the significance of *security risk management*.
2. How do you derive *security requirements* from risk analysis?
3. How do *security considerations* influence the design of system architecture?
4. What is *system survivability*? Why is it important for complex software systems?

#### 6.5.2 Service-oriented software engineering

1. What is a *Web service*?
2. What are the standards for Web services?
3. Discuss *service engineering process* for developing reusable Web services.
4. Discuss the role of *service composition* as a means for developing service-oriented applications.
5. Explain how *business process models* may be used as a basis for the design of service-oriented systems.

#### 6.5.3 Aspect-oriented software development

1. Explain why *separation of concerns* is a good guiding principle for software development?
2. What are the *fundamental ideas* underlying aspects and aspect-oriented software development?
3. Discuss how you can use aspect-oriented approach to requirements engineering, software design, and programming?

4. What are the problems in *testing* aspect-oriented system?

## 7 Course assessment

The course assessment components include: written assignments (20%), team project (20%), two midterm exams (40%), and a final (20%). Maximum possible score is 100. Course grade is awarded based on the following scheme:

| <i>Score</i>          | <i>Letter Grade</i> |
|-----------------------|---------------------|
| $\geq 90$             | A                   |
| $\geq 80 \ \& \ < 90$ | B                   |
| $\geq 70 \ \& \ < 80$ | C                   |
| $\geq 60 \ \& \ < 70$ | D                   |
| $< 60$                | F                   |

### 7.1 Written Assignments

There will be several, small written assignments. Answers to these questions must be submitted via WebCT Vista.

### 7.2 Team Project

There will be a semester-long team project. This will be continuation of CS 305 project. You will do the software design, implement your design, and test your application. You will learn how to use various IBM Rational CASE tools.

## 8 Instructional materials

**Required Textbook** Ian Sommerville, *Software Engineering* (8<sup>th</sup> edition), ISBN: 0-321-31379-8, Pearson Education, 2007.

**Additional Resources** Course notes and other handouts will be available on WebCT Vista. URLs for additional resources will also be listed on the Vista.

## 9 WebCT Vista

It is important to visit WebCT Vista for up-to-date information about the course. It hosts all the course materials including assignments, handouts, lecture notes, and reading materials. Also, you will use the Vista for submitting your team project.