CSC 323 - SOFTWARE ENGINEERING

CREDIT HOURS: 3
PREREQUISITES: CSC 302
GRADE REMINDER: Must have a grade of C or better in each prerequisite course.

CATALOG DESCRIPTION


PURPOSE OF COURSE

To provide the student with knowledge of software engineering principles that can be applied to the software process.

EDUCATIONAL OBJECTIVES:

Upon successful completion of the course, students should be able to:

1. Identify software development problems that provided the impetus for the start of software engineering.
2. Demonstrate an understanding of the different perspectives from which software is considered by users, clients, and commercial and in-house developers.
3. Describe the importance of software maintenance, and the nature of the software life cycle.
4. Describe the various software process models that have been used for software development and gain familiarity with important software development methodologies.
5. Work in a disciplined software development team demonstrating the use of COCOMO, function points, and other methods to estimate the size of a development effort.
6. Produce important artifacts of software development other than code.
7. Demonstrate an understanding of the role of software quality assurance and practice non-execution based testing.
8. Develop a prototype as a means of requirements validation.
9. Derive and use metrics for software development.
10. Use state-of-the-practice software estimation techniques.

COURSE CALENDAR

This course meets for a minimum of 37.5 lecture contact hours during the semester, including the final exam. Students have significant weekly extracurricular assignments which may involve reading, teamwork and team meetings, or engaging in other forms of preparation. Students are expected to complete 7-10 hours of programming assignments, and 2-3 periodic exams in addition to the final exam. There will also be 3 class presentations based on a 4 – 5 person team. Students are expected to prepare for any class assignments or quizzes over the material covered in class or the extracurricular assignments. Successful completion of these activities requires at a minimum six additional hours of outside of classroom work each week.

CONTENT

<table>
<thead>
<tr>
<th>Description</th>
<th>Hours</th>
</tr>
</thead>
</table>
| Introduction | 3
| History of software engineering | 
| The need for a disciplined approach | 
| Software process models |
## Software Engineering Issues

- Quality, productivity, accuracy, reliability, maintainability, reusability
- The use of metrics
- The role of Computer-Assisted Software Engineering (CASE)

## Requirements Engineering

- Requirements definition and analysis
- Feasibility study
- Cost/benefits analysis
- Prototyping
- Tools

## Design

- Methodologies: structured design, functional decomposition, data-flow oriented, data-oriented, object-oriented design
- Tools

## Implementation and Testing

- Programming environments, teams, languages, and style
- Programming principles: cohesion, coupling, modularity, information hiding
- Test case design, classes of tests
- Quality assurance, verification, validation, reliability
- Testing methods
- Tools

## Evolution

- Operation: performance analysis and measurement
- Maintenance
- Reverse engineering

## Exams (plus final)

<table>
<thead>
<tr>
<th>Percentage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Exams (plus final)</td>
</tr>
</tbody>
</table>

**TOTAL 45**

## REFERENCES