BOWIE STATE UNIVERSITY
Syllabus
Department of Computer Science

COSC 665 (3 Cr) **Software Engineering II**

Instructor:	Sharad Sharma
Office Location:	Computer Science Building Room 317
Phone:	301-860-4502
Email:	ssharma@bowiestate.edu
Class Hours:	Thursday: 4:55 PM- 7:25PM
Office Hours:	Thu: 12:25 PM to 4:55 or by appointment

COURSE DESCRIPTION

The course will cover software life-cycle models and different phases of the software development process. Object-oriented techniques are applicable. Students will have a group project on developing complex software systems.

COURSE PREREQUISITE COSC 475 or COSC 565

REQUIRED TEXTS

Bernd Bruegge and Allen H. Dutoit (2004) *Object-Oriented Software Engineering: Using UML, Patterns and Java*, Second Edition, Prentice Hall, Upper Saddle River, NJ.

STUDENT EXPECTED OUTCOMES

Upon completion of this course, the student will be able to:

- Develop and produce software process artifacts, most importantly the code and user documentation.
- Demonstrate skills that help you work effectively as a member of a software development team.
- Analyze the fundamental principles of software engineering. Be able to identify and describe the software life cycle, roles, artifacts, and activities.
- Demonstrate the concepts of software "best practices" and when they apply.
- Synthesize the concepts of software "best practices".

STUDENT LEARNING OBJECTIVES

The essential objectives for this course are to:

1. Describe and define a feasibility plan, requirements, and design documentation.
2. Develop proficiency in developing and producing software process artifacts, most importantly the code and user documentation.
3. Acquire skills that help you work effectively as a member of a software development team.
4. Analyze and learn how to identify and describe the software life cycle, roles, artifacts, and activities.
5. Synthesis the concepts of software "best practices" and when they apply.
6. Show that you are able to adapt a process to your needs and select an appropriate set of best practices that will guide you in completing a software development project.

TEACHING MODES

All course material will be provided on a course web site including lecture notes, useful links on the web, recommended references, time schedule, and contact information for faculty, guidelines for projects, coding standards, and more. The primary teaching mode will be lecture and discussion.

COURSE REQUIREMENTS AND EXPECTATIONS

Policy on Attendance: Regular attendance in the class is mandatory. Students will be responsible for any loss of information, assignments, and projects due to absence from class.

Departmental Policy on Submission of Late Work*:* There will be no make-up for any missed classes, projects, assignments, and exams. 1/2 letter grade off for assignment each day late without documented excuse; papers more than one week late will not be accepted.

Academic Integrity: Academic dishonesty includes plagiarism, cheating, and other illegal or unethical behaviors in doing the work of the course. Plagiarism is the act of representing another's ideas, words or information as one's own. If you receive assistance on an assignment from someone else, you must avoid plagiarism by giving proper credit for this assistance. Include in your assignment a comment naming the person who assisted you and stating what the assistance was. Students who are guilty of academic dishonesty are subject to severe penalties ranging from a reduction in points (and possible failure) for the assignment/project, to failing the course, or in extreme cases, dismissal from the University. Do not copy other student's projects, codes, and design. A group of students working together on a project must change their forms and codes to differentiate from others.

EVALUATION: Following is the Evaluation system for the Final Grade. Each assignment will be graded. Students are responsible for completing them as scheduled.

1.	Assignments	25%
2.	Presentations	15%
3.	Mid-Term Exam	20%
4.	Final Exam	20%
5.	Final Project	20%

Final Project, Mid-term and Final exams are mandatory.

Assignments:
The assignments include research paper critiques and a research paper.

Paper review assignments and guidelines:

For each paper, students should write a review answering each of the following questions:
1. What problems (with prior work or the lack thereof) were addressed or surveyed by the authors?
2. What solutions were proposed or surveyed by the authors?
3. What are the technical strengths and main contributions of the paper's proposed solutions?
4. What are the technical weaknesses of the paper's proposed solutions?
5. What suggestions do you have to improve upon the paper's ideas?

Research papers will be assigned to students to read, analyze and present to the class. Presentations will be structured as follows:

- Presentation
- Questions to presenter
- Open discussion

On the day of your paper review, you should bring your review presentation, i.e., power point file (flash drive), to the class. In total 15 ~20 minutes each, including:
- Brief description of (1) introduction/idea; (2) method (experimental design, participants, apparatus, experiment procedure, data collection); (3) results; (4) discussion and/or conclusion; and (5) etc.
- What knowledge did you learn from the paper/work, e.g., anything you've never known before; which part of the work interests you most…

In-Class Paper Presentation Grading:
Items above, quality of oral/written presentation and visuals, timeliness, etc. The grading rubric are

- Content (35% Weighting)

- Preparedness (35% Weighting)
- Visual Aids/Handouts (15% Weighting)
- Discussion or Questions raised(15% Weighting)

Final Project: The purpose of the course project is to provide the students with the knowledge of software engineering methodology and the skills to apply it. The particular project is not the goal in itself; rather, it serves as a vehicle to apply your knowledge and to develop the skills. Projects also introduce students to team work, which is a must for large-scale software development. It also emerges as a key methodology for any- and every-scale software development, something called extreme programming. *Team work is **required** since team work is an integral part of large-scale software development.*

Research Paper: Each student is expected to do a research paper on a topic. Topics can be drawn from the student's research area, project from previous software Engineering I class, or in consultation with the instructor.

Research paper will be graded through following rubric

- Quality of Information (25% Weighting)
- Organization (25% Weighting)
- Mechanics & grammatical or spelling (20% Weighting)
- Paper properly formatted in IEEE format (15% Weighting)
- Research Paper Construction (15% Weighting)

GRADING: Academic dishonesty will result in grade F. The following grade scale will be used:

90 % - 100% = A
80 % - 89% = B
70 % - 79% = C
60 % - 69% = D
0 - 59% = F

Final grades will be computed based upon credits earned for all the five components mentioned above.

COURSE/TOPICAL OUTLINE

WEEK 1 & 2: Software Lifecycle and Team Projects
WEEK 3: Requirements Elicitation and Use Cases
WEEK 4 & 5: Object-Oriented Analysis
WEEK 6 & 7: Object-Oriented Design
WEEK 8: Implementation and Testing
WEEK 9 & 10: Software Architecture
WEEK 11: System Specification
WEEK 12 & 13: Design Patterns

WEEK 14: Software Components
WEEK 15: Presentations

Reminder: English Proficiency Examination
After successfully completing ENGL 101 and 102, Composition and Literature I and II, students must take and successfully pass the Bowie State University English Proficiency Examination. Transfer students who completed their English composition requirements at another university should take the English Proficiency Examination during their first semester of enrollment at Bowie State University.

ADA Statement:
Students with disabilities who wish to receive ADA accommodations should report to the Office of Special Populations, Center for Learning and Technology (CLT) building, Suite 302 (301-860-3292).

SELECTED BIBLIOGRAPHICAL REFERENCES

1. Barnes D. J., and Kölling, M. (2003) *Objects First With Java*, Prentice Hall,.
2. Bruegge, B., and Dutoit, A. (2000). *Object-Oriented Software Engineering Conquering Complex and Changing Systems,* Prentice-Hall.
3. Bruegge, B., and Dutoit, A. H. (2004). *Object-Oriented Software Engineering: Using UML, Patterns and Java,* Second Edition, Prentice Hall.
4. David, K. (1998). *The Art of Computer Programming*, V. 1-3, 2nd ed. Boxed Set, Addison-Wesley.
5. Eckel, Bruce. (2006). *Thinking in Java,* Fourth Edition, Prentice Hall PTR, Upper Saddle River, NJ.
6. Ghezzi, J., and Mandrioli, P. (1991). *Fundamentals of Software Engineering* by Hall.