CSCI 360 Software Architecture and Design Spring 2017
Syllabus

Instructor:
Dr. Jim Bowring: http://www.cs.cofc.edu/~bowring/
Office: Harbor Walk East, Room 308
Tel: 843.953.0805
Google Voice: 843.608.1399
Google Chat: bowring@gmail.com
E-mail: Please use BowringJ@cofc.edu with SUBJECT = "CSCI360"
Office hours: TR: 10:00 – 11:00 or by appointment

Class place and time:
Classroom: HWEast, Room 301, Times: TR 2:10-3:25
Website: Section-1

Catalog description:
This course covers the object-oriented analysis and design of software. Topics include the unified modeling-language, domain modeling, software architecture, design processes, principles, heuristics, and patterns. Student teams analyze, design, and implement a software system. Related ethical issues are explored. Lectures three hours per week.
Prerequisite: CSCI 230 – Data Structures and Algorithms
Co-requisite: COMM 104 – Public Speaking

Detailed Course Description with Course Outcomes
Full document is attached to this syllabus, and includes these course outcomes:
Upon successful completion of the course, students will:

Course Outcomes	Program Outcome Linkage
1. Describe the ethical issues associated with software architecture and design.	e
2. Describe and evaluate how iteration with an agile approach works to create a resilient OO analysis and design process such as the Unified Process.	a, c, i
3. Work in teams to design software.	a, b, c, d, f, i, j, k
4. Analyze a software application problem with use cases.	b, c, i
5. Produce a conceptual domain model with UML class diagram, associations, roles, multiplicities	i
6. Use System Sequence Diagrams to illustrate operations.	i
7. Produce operation contracts.	i
8. Describe evaluate the most common logical architectures for software systems and the nature of message passing among architectural components.	b, c, i
9. Explain the nature and use of software patterns and give at least one illustrative example complete with code.	i
10. Demonstrate the basics of software object design and the assignment of responsibilities and collaborations with reference to GRASP.	b, i, j
11. Use UML activity diagrams to analyze and model processes.	b, i, j
12. Use UML state diagrams to analyze and model states.	b, i, j
13. Demonstrate basic design principles: Law of Demeter, Liskov Substitution Principle, SOLID.	b, i, j

14. Explain and implement test-driven development.	c, i
15. Exhibit a basic working knowledge of GUI development using an IDE.	i
16. Produce a Software Architecture Document.	f, b, c
17. Write and present orally analyses of topics in software analysis and design.	f

Required text:
Applying UML and Patterns: An Introduction to OO Analysis and Design and Iterative Development 3rd Edition, by Craig Larman, Prentice-Hall, 2005.

Required Account:
Each student is required to have a GitHub account. Each student will hyperlink their full name [last, first] to their GitHub account on the class wiki.

Electronic Resources:

1) Software Engineering Body of Knowledge (SWEBOK)
1) Google Scholar; Google Documents: http://docs.google.com ;
2) The College of Charleston Libraries supply free full access to the ACM Digital library and the IEEE Computer Society Journals.
3) CofC: Career Center, Cistern Online, Center for Student Learning
4) GitHub

Homework and assignment policy:
All individual and team assignments are due when specified per the class website. Each assignment will be accessed and submitted through a private GitHub repository for each student or team. Assignments must be professional in content and appearance.

Team projects:
Students will form into teams in the first days of class. Various team projects will be assigned during the semester. Teams will arrange to work outside of class. Teams will be assigned GitHub repositories for project development. Each team will maintain a professional-grade wiki within their repository documenting their work.

Attendance, class participation, online presence, and oral presentations:

ATTENDANCE IS MANDATORY.

Each absence after 3 has a penalty of 5 of 100 final points.

Your active participation in class will lead to your success and your team's success. You are required to maintain a professional-grade team repository and wiki where you record your team’s progress and work artifacts, and that will be used to evaluate your work. I expect you in class on time and well-prepared by having read the assigned

readings, performed the assigned tasks, and updated team repositories and wikis. Each student will give oral presentations on demand during the semester and an oral presentation as part of their team's final report.
Please do not attend class if you are sick or believe you are becoming ill. It is best to document your absence through an absence report in Undergraduate Academic Services.

Professional Development:
I highly recommend that you join the Association for Computing Machinery (ACM = $19 for a student), ACM's Women in Computing (WIC), the Institute of Electrical and Electronics Engineers (IEEE) Computer Society, the National Center for Women and Information Technology (NCWIT), or join our student ACM, student ACM-WIC, Cyber Security Club, or Data Science Club for free! A great way to advance your career is to become an active member of at least one of these professional organizations.

Disabilities:
If you have a documented disability and approval to receive accommodations through SNAP Services, please contact me during my office hours or by appointment.

Student Conduct:
I expect you to abide by The College of Charleston Student Handbook, which includes sections on conduct and the Honor Code. If you have a question about how to interpret the Honor Code, ask before acting! I encourage collaboration on assignments and projects, but you must document the collaboration with the names of your collaborators on the assignment.

Evaluation Scheme:

10% : Class preparation and participation
25% : Individual Assignments
25% : Team project including wiki, contributions, and presentations
15% : One test
25% : Final exam

Grading scale: (Each absence after 3 is a penalty of 5 of these 100 points)

Exceptional:	A	(90+) perform all work on time with good quality PLUS initiative
Adequate:	B	(80-89) perform all work on time with good quality
Poor:	C	(70-79) consistently miss deadlines and/or poor quality
Else:	F	

The official Course Description for ABET accreditation follows.

Course Description

Computer Science Department, College of Charleston

Course Number:	**CSCI 360**
Course Title:	**Software Architecture and Design**
Course Coordinator:	**Bowring**

Catalog Description

This course covers the object-oriented analysis and design of software. Topics include the unified modeling-language, domain modeling, software architecture, design processes, principles, heuristics, and patterns. Student teams analyze, design, and implement a software system. Related ethical issues are explored. Lectures three hours per week.

Prerequisite: CSCI230. Pre/Co-requisite: COMM104.

Prerequisites by Topic

OO Programming: Java or C#
Familiar with classes, abstract classes, interfaces, polymorphism
Writing skills for design documentation
Oral communication skills

Major Topics Covered in the Course (Required Topics)

1. Documentation: continuous and agile
2. Ethical issues
3. Teamwork
4. Rational Unified Process
5. OO Analysis
   - 1.1.1.1.1. Business case
   - 1.1.1.1.2. Use cases
   - 1.1.1.1.3. Domain models
   - 1.1.1.1.4. UML class diagrams: associations, roles, multiplicities
   - 1.1.1.1.5. UML System Sequence Diagrams
   - 1.1.1.1.6. System Operation Contracts
   - 1.1.1.1.7. System Logical Architecture: models, representations
6. OO Design
   1. Software objects: responsibilities and collaborations
   2. Design of System Operations: class diagrams, sequence diagrams
   3. Software Design Patterns (Gang of Four)
   4. GRASP: General Responsibility Assignment Software Patterns
   5. UML Activity diagrams
   6. UML State diagrams
   7. Design principles: Law of Demeter, Liskov Substitution Principle, SOLID
7. OO Implementation
   1. Test-driven development
   2. GUI development

Course Narrative

Students will learn formal techniques to aid them in analyzing real-world software application problems and designing working solutions, complete with documentation and tests. Students will learn iteratively and apply their knowledge to the iterative solution of personal and team projects.

Laboratory projects

Students will work all semester on personal and team projects that will evolve with the material learned in class. The approach is agile and iterative, so all work is "live" and continuously improved.

Course Outcomes

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

Course Outcomes	Program Outcome Linkage
1. Describe the ethical issues associated with software architecture and design.	e
2. Describe and evaluate how iteration with an agile approach works to create a resilient OO analysis and design process such as the Unified Process.	a, c, i
3. Work in teams to design software.	a, b, c, d, f, i, j, k
4. Analyze a software application problem with use cases.	b, c, i
5. Produce a conceptual domain model with UML class diagram, associations, roles, multiplicities	i
6. Use System Sequence Diagrams to illustrate operations.	i
7. Produce operation contracts.	i
8. Describe evaluate the most common logical architectures for software systems and the nature of message passing among architectural components.	b, c, i
9. Explain the nature and use of software patterns and give at least one illustrative example complete with code.	i
10. Demonstrate the basics of software object design and the assignment of responsibilities and collaborations with reference to GRASP.	b, i, j
11. Use UML activity diagrams to analyze and model processes.	b, i, j
12. Use UML state diagrams to analyze and model states.	b, i, j
13. Demonstrate basic design principles: Law of Demeter, Liskov Substitution Principle, SOLID.	b, i, j
14. Explain and implement test-driven development.	c, i
15. Exhibit a basic working knowledge of GUI development using an IDE.	i
16. Produce a Software Architecture Document.	f, b, c
17. Write and present orally analyses of topics in software analysis and design.	f

Oral and Written Communications

Every student is required to submit at least __2_ written reports (not including exams, tests, quizzes, or commented programs) of typically _2__ pages and to make _1_ oral presentations of typically _5__ minute's duration. Material is graded for grammar, spelling, style, technical content, completeness, and accuracy.