

**CIS 320 – Introduction to Software Engineering**v 1.0

|  |  |  |
| --- | --- | --- |
| **Professor:** | |  | | --- | | Dr. André Fonteles | |
| **Credits:** | |  | | --- | | 3 | |
| **Class Time and Place:**  OHSN 163 | |  |  | | --- | --- | | Mondays, Wednesdays Fridays: | 12:20PM – 1:15PM | |
| **Office & Hours**  OHSN 170-D | |  |  |  |  | | --- | --- | --- | --- | | *Monday* | *1:30pm-5:00pm* | *Thursday* | *2:40pm-3:10pm* | | *Tuesday* | *2:40pm-4:10pm* | *Friday* | *-* | | *Wednesday* | *1:30pm-5:00pm* | *Also by appointment or walk in* | | |
| **Contact** | |  |  | | --- | --- | | [andre.fonteles@indwes.edu](mailto:andre.fonteles@indwes.edu) | 765-755-3772 (Office) | |

### Advice & Encouragement:

1. “*To everything there is a season, and a time for every matter under heaven: […] a time to keep silent, and a time to speak; […]*” Eccl 3:1, 7. In a computer science course, there may be more problem with students keeping silent than with them speaking. Questions are appropriate and appreciated at any time. Giving answers to my questions is also appreciated (unless you just answered the last question; then giving another student the opportunity to answer would be appropriate.)
2. “*Ponder the path of your feet; then all your ways be sure.*” Prov 4:26. The schedule shows what we will be covering the next class period. The wise will have previewed the chapter(s) before coming to class; all the relevant studies in learning show that having some prior knowledge greatly increases the rate and level of comprehension.
3. “*Iron sharpens iron, and one man sharpens another.*” Prov 27:17. Students perform better (often much better) when they make an effort to become part of the class “community.” Academic benefits of studying and socializing together include: extra feedback to get your misconceptions corrected, other perspectives on what the important issues really are, emotional energy when preparing for tests, an emergency contact when you forget what assignment is due…
4. “*To one he gave five talents, to another two, to another one, […]*” Matt 25:15. There will be a wide range of backgrounds and abilities in the class. Be respectful of others. Strive to develop your God-given ability, not to be better than others. Don’t be too embarrassed to ask “stupid” questions. Love and help your neighbor.
5. “*Whatever you do, work heartily, as for the Lord and not for men, knowing that from the Lord you will receive the inheritance as your reward. You are serving the Lord Christ.*” Col 3:23-24. Jesus is glorified when you do your best in this course. Do it as for the Lord!
6. “*For what does it profit a man to gain the whole world and forfeit his soul?*” Mark 8:36. Work hard and take this course seriously, but know that: even if you become the best computer scientist in the world, what gain would you have if you don’t have Jesus and forfeit your soul? Jesus is the most precious pearl and the most valuable hidden treasure one can ever find (Matt 13:44-46).

### Prerequisites

* Prerequisites or corequisite: CIS-221.

### Course Description

This course provides an introduction to the process of developing large scale software systems. Included is the study of the procedures and tools of software engineering. Concepts introduced are software life-cycle models, quality factors, requirements analysis and specification, software design (functional design and object-oriented design), implementation and testing.

### Course Objectives

After taking this course, the students should be able to:

1. Have gained an understanding of software engineering principles;
2. Have explored such topics as systems life cycle models, requirements definitions and analysis, specifications, software testing techniques and verification and validation, software quality factors;
3. Have considered the goals and requirements of software project management;
4. Have completed an overview of current software methodologies and design practices; and
5. Have considered how the practice of software engineering and its associated implications in ethics and professionalism may be practiced by Christians intent on changing their world.

### Required Text and Materials

### Software Engineering ISBN: 9780133943030 Authors: Sommerville, Ian Published by Pearson Education 10th e. Publication Date: Mar. 24, 2015

### The Mythical Man-month ISBN: 9780201835953 Authors: P. Brooks, Frederick Published by Addison Wesley Anniversary Edition (2nd e.) Publication Date: Aug. 2, 1995

### Late work

### Work turned in after the due date & time may receive a mark of 0. This policy is meant to help the students in regards to accountability.

### Grade Distribution

**Grading Scale:**  **Grade Weights:**

95 – 100 --- A 73 – 76 --- C Participation 10%

90 – 94 --- A- 70 – 72 --- C- Book Analysis 20%

87 – 89 --- B+ 65 – 69 --- D Project 40%

83 – 86 --- B 60 – 64 --- D Midterm Exam 15%

80 – 82 --- B- 0 - 59 --- F Final Exam 15%

77 – 79 --- C+

### Description of Assignments

#### Participation: The classroom times are meant to be interactive eliciting comments, questions, and concerns. The expectation for the course is that students will not only attend but participate in the discussions.

#### Assignments grading: Most assignments count as participation only and are likely to be graded in terms of completeness, not correctness. However, you are encouraged to request help or feedback from me with any questions you may have during your homework. I reserve the right to grade any homework in terms of correctness if the students are believed to be attempting to trick this system.

#### Project & Chapter Reflection: By the end of most weeks, each student should prepare a text document containing no less than 200 words reflecting on how the project you are working on in this class relate to the subjects discussed in class and in the week reading. This document should include a statement explicitly informing how many words your reflection contains. Although the reflection refers to the group project, this is an individual assignment. Your reflection counts towards participation.

#### Book Analysis: *see attached document.*

#### Project: *see attached document.*

#### Midterm Examination: The midterm examination happens in the same place and time as normal classes. The midterm examination will be based on material from the classroom, lab discussions, textbook, and quizzes. The exact date is available below.

#### Final Examination (During Exam Week): The final examination will be comprehensive and cover material from both before and after the midterm. The final examination will be based on the classroom, lab discussions, textbook, and quizzes.

**Course Outline**

(subject to change)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Week** | **Date** | **Course Topic** | **Assignments** | **Readings** |
| Week 1  (08/29) | Mon. | No class | - | - |
| Wed. | No class | - | - |
| Fri. | Class Introduction | - | - |
| Week 2 (09/05) | Mon. | No class | - | - |
| Wed. | Introduction to SE | - | - |
| Fri. | Introduction to SE | Chapter exercises | Chapters 1 |
| Week 3 (09/12) | Mon. | Software Process | - | - |
| Wed. | Software Process | - | - |
| Fri. | Software Process | Project/Chapter reflection  Chapter exercises | Chapter 2 |
| Week 4 (09/19) | Mon. | Agile Software Development | PML 1 | - |
| Wed. | Agile Software Development | - | - |
| Fri. | Agile Software Development | Project/Chapter reflection  Chapter exercises | Chapter 3 |
| Week 5 (09/26) | Mon. | Requirements Engineering | PML 2 | - |
| Wed. | Requirements Engineering | - | - |
| Fri. | Requirements Engineering | Project/Chapter reflection  Chapter exercises | Chapter 4 |
| Week 6 (10/03) | Mon. | System Modeling | PML 3 | - |
| Wed. | System Modeling | Project/Chapter reflection  Chapter exercises | Chapter 5 (up to 5.3) |
| Fri. | Fall Break | Fall Break | Fall Break |
| Week 7 (10/10) | Mon. | Architectural Design | - | - |
| Wed. | Architectural Design | - | - |
| Fri. | Architectural Design | Project/Chapter reflection  Chapter exercises | Chapter 6 |
| Week 8 (10/17) | Mon. | Midterm review | IR 1 | - |
| Wed. | Midterm review | - | - |
| Fri. | **Midterm Exam** | - | - |
| Week 9 (10/24) | Mon. | Design and Implementation | - | - |
| Wed. | Design and Implementation | - | - |
| Fri. | Design and Implementation | Project/Chapter reflection  Chapter exercises | Chapter 7 |
| Week 10 (10/31) | Mon. | Software Testing | IR 2 | - |
| Wed. | Software Testing | - | - |
| Fri. | Software Testing | Project/Chapter reflection  Chapter exercises | Chapter 8 |
| Week 11 (11/07) | Mon. | Software Evolution | - | - |
| Wed. | Software Evolution | - | - |
| Fri. | Software Evolution | Project/Chapter reflection  Chapter exercises | Chapter 9 |
| Week 12 (11/14) | Mon. | Project Management | IR 3 | - |
| Wed. | Project Management | - | - |
| Fri. | Project Management | Project/Chapter reflection  Chapter exercises | Chapter 22 |
| Week 13 (11/21) | Mon. | **Book Analysis Presentation** | Book Analysis paper and presentation material | - |
| Wed. | Thanksgiving | Thanksgiving | Thanksgiving |
| Fri. | Thanksgiving | Thanksgiving | Thanksgiving |
| Week 14 (11/28) | Mon. | Quality Management | - | - |
| Wed. | Quality Management | - | Chapter 25 |
| Fri. | Quality Management | Project/Chapter reflection  Chapter exercises | Chapter 24 |
| Week 15 (12/05) | Mon. | **Project Presentation** | IR 4 | - |
| Wed. | Configuration Management | - | - |
| Fri. | Configuration Management | Project/Chapter reflection  Chapter exercises | Chapter 25 |
| Week 16 (12/12) | Mon. | Final Review | - | - |
| Wed. | Exam Week | Exam Week | Exam Week |
| Fri. | Exam Week | Exam Week | Exam Week |
| **FINAL EXAM (Tuesday, Dec. 14, 2021 - 1:00-2:50pm)** | | | | |

### Academic Honesty: *see Brightspace.*

### Student with Disabilities: *see Brightspace*.

### Writing Center: *see Brightspace.*

### Syllabus Content:

### The professor reserves the right to make changes to this syllabus, if deemed necessary. All changes will be provided to the students orally or in writing before the implementation of the change.

**Coding Guidelines**

1.     Be consistent in your formatting and indentation (as your submissions may be graded on this point).

2.     Don’t use magic numbers (*i.e.*, non-obvious values that aren’t attributed to variables or constants).

3.     Use descriptive variables names. Typically, single letter variables are ***not*** descriptive (*e.g.*, x) – they are just lazy names.

4.     There should be no warnings or errors when compiling and running your code.

5.     **Any submission that does not run may receive a 0. Any late submission may receive a 0.**

Project Specification

This project will provide opportunities for students to discuss, explore, and exercise the topics addressed in the classroom via group work. Group are composed of two students.

The goal of this project is to have each group going through the implementation of a large-scale software project using an agile-like development approach. You will learn by doing. Therefore, every member of the team must contribute.

Since we are adopting an agile-like development approach, groups will be turning in working executable software, that create value to stakeholders, by the end of each iterations. Waiting until the end of the course and trying to code everything (regardless if it works) will produce a poor grade. Here are some guidelines for the end of each iteration:

* + Your code should be fully functional, integrated into the system and bug free.
  + Your source code as well as the binary executable files should be available on Github.
  + The issues representing each user story must be marked as complete on Github.
  + At least one *unit test* or *integration test* must have been created to demonstrate that the user story can be executed correctly. The tests can be created using a technology such as *JUnit* (Java) or *unittest* (Python).

A key part of the course is staying on the schedule and following guidelines.

NOTICE: Projects will be group work and graded by group. Each group member will get the same grade for project work, except for exceptional circumstances.

**Schedule**

All items here discussed must be ready and/or submitted (where applicable) before class.

**Planning Mile Stones (PML)**

**PML 1**: each pair of students must have submitted a one-page document stating what software they will be building during the semester. This document must include at least two related, and possible competing, software/services.

**PML 2:** each group must have created and experimented with a Github account. Your project will be hosted on Github. Each group must showcase some hello-world applications using the main technologies (e.g., languages, frameworks, etc.) intended to be used in the development process.

Your Github repository should contain a *README.md* file adequately describing the project (you can use the description turned in at PML1).

**PML 3:** each group must have a complete set of user stories defining the main behavior of the software to be created. These user stories must be posted on Github by means of issues. A few user stories will be distributed (assigned through Github) to each member of the group in order to have them implemented during the first development iteration.

**Iterations Reviews (IR)**

**IR 1:** each user story assigned must be fully implemented and executable in this first version of your software. In order words, a user must be able to use your software to reproduce the user story. Someone from a different group should be able to download and execute your software in order to test the new functionalities. Consider it to be a “final version” of whatever has been implemented when it comes to quality.

More user stories, not yet implemented, will be assigned to each member in order to have them implemented during the next iteration.

**IR 2:** each user story assigned must be fully implemented and executable in this first version of your software. In order words, a user must be able to use your software to reproduce the user story. Someone from a different group should be able to download and execute your software in order to test the new functionalities. Consider it to be a “final version” of whatever has been implemented when it comes to quality.

More user stories, not yet implemented, will be assigned to each member in order to have them implemented during the next iteration.

**IR 3:** each user story assigned must be fully implemented and executable in this first version of your software. In order words, a user must be able to use your software to reproduce the user story. Someone from a different group should be able to download and execute your software in order to test the new functionalities. Consider it to be a “final version” of whatever has been implemented when it comes to quality.

More user stories, not yet implemented, will be assigned to each member in order to have them implemented during the next iteration.

**IR 4:** each user story assigned must be fully implemented and executable in this first version of your software. By this iteration, your software must be complete. All user stories should have been implemented throughout the different iterations. Be ready to present the final version of your software to the class.

Each presentation should last approximately 15 minutes and highlight:

- Description of the software created  
- How was the development process  
- What were the main challenges  
- What did you enjoy the most  
- What did you learn.

Book Analysis: The Mythical Man-Month

The Mythical Man-Month is a classic book on software engineering that back in the 90’s had already sold more than 250,000 copies. Despite its first edition having being written in 1975, this classic remains extremely popular, relevant and educational for the aspiring software engineer and project manager.

**Paper**

In this course, groups of students will study specific chapters of The Mythical Man-Month, prepare a corresponding write-up paper and present on their findings. For each chapter assigned, the team must produce at least one page (excluding figures, references, etc.) with:

1. Summary description of the chapter and its main ideas.
2. Utilize our main textbook and at least one peer-reviewed paper (e.g., ACM, IEEE, Springer) to discuss the chapter’s relevance in today’s software engineering. Do the propositions still hold? Why, or how?

As part of the grade for each paper students need to schedule and attend a session at the Inkwell to cover areas of writing such as (but not limited to) citations, punctuation, grammar, transitions, introduction, and conclusion. 20% of the write-up grade will be deducted if proof of an Inkwell session is not included.   
  
**Presentation**Each group should perform a presentation based on the content of the paper previously written. A student should present no less than 5 minutes and no more than 10 minutes. Thus, a group of, for example, two students should present between 10 and 20 minutes.

The use of a presentation software package is required (e.g., Microsoft PowerPoint, LibreOffice Impress), but other forms of creatively disseminating information are also encouraged (videos are not allowed). Any presentations slides used need to be numbered for easy reference.

**Grading**

|  |  |
| --- | --- |
| **Paper** | 70% |
| **Presentation** | 30% |

|  |  |
| --- | --- |
| **Paper** | |
| Summary of chapter | 30% |
| Discussion based on textbook | 30% |
| Discussion based on research paper | 30% |
| Overall quality | 10% |

|  |  |
| --- | --- |
| **Presentation** | |
| Quality of information and organization | 30% |
| Speaker’s confidence[[1]](#footnote-1) | 30% |
| Visual tools quality | 20% |
| Overall adherence to guidelines | 20% |

1. Just reading the content from some source does not display any confidence. [↑](#footnote-ref-1)