University of Macau
Computer and Information Science Department
CISB251 – Object-Oriented Analysis and Design Patterns
Syllabus
2nd Semester of Year 2
Part A – Course Outline

Elective course in Computer Science

Catalog description:

(2-2) 3 credits. The course discusses object-oriented analysis and design using Unified Modeling Language (UML). The main contents include use case diagram, class diagram, sequence diagram, state diagram, and activity diagram of UML. Object Constraint Language (OCL) and design patterns are also introduced. The students are asked to analyze and design their course project systems on teams with UML CASE tool.

Course type:

Theoretical with substantial laboratory/practice content

Prerequisites:

- None

Textbook:

- Craig Larman: ***Applying UML and Patterns***, 3rd ed. Prentice-hall, 2005.

References:

- J. Arlow and I. Neustadt: ***UML2 and the Unified Process: Practical Object-Oriented Analysis and Design***, 2nd ed. Addison Wesley, 2005.
- G. Booch, J. Rumbaugh, and I. Jacobson: ***The Unified Modeling Language User Guide***, Addison-Wesley, 1998.
- E. Gamma, R. Helm, R. Johnson and J. Vlissides: ***Design Patterns: Elements of Reusable Object-Oriented Software,*** Addison-Wesley, 1995.

Major prerequisites by topic:

1. Basic concepts of object-oriented programming.

Course objectives:

1. Analyze and design with object-oriented method in UML [a, b, c, d]
2. Introduce OCL and design pattern concepts [a, b, c, d]
3. Apply object-oriented method to the course project system analysis and design [a, b, c, d]

Topics covered:

1. **Introduction of OOA/OOD with UML (4 hours):** Introduce the concepts of object-orientation, object-oriented analysis and design, Unified Modeling Language (UML). In addition, the concepts of software development process and activities, and Unified Development Process are also introduced. A case study is used to illustrate the overview of object-oriented analysis and design with UML.
2. **Use Case Model (4 hours):** Analyze and specify the requirements model, including use case diagram, use case definition, system operation sequence diagram, activity diagram, operation contract with pre and post conditions, and conceptual class diagram by illustrating with case studies.
3. **Static Design Model (4 hours):** Design system static model, including design class diagram, identification of classes, attributes and methods, identification of generalization, aggregation, composition, and dependency relations, defining associations with multiplicities and constraints by illustrating with case studies.
4. **Dynamic Design Model (6 hours):** Design system dynamic model, including design sequence diagram, activity diagram and state diagram, mapping design to codes by illustrating with case studies.
5. **OCL and Design Patterns (4 hours)**: Introduce the concepts of Object Constraint Language (OCL) and design patterns. OCL is used to specify pre, post conditions and invariants by illustrating with examples. The GRASP patterns and some GoF patterns are discussed, such as designing objects with responsibility, expert pattern, creator pattern, controller, the concepts of low coupling and high cohesion, façade and adapter patterns.

Class/laboratory schedule:

Timetabled work in hours per week			No of teaching weeks	Total hours	Total credits	No/Duration of exam papers
Lecture	Tutorial	Practice				
2	2	Nil	14	56	3	2 / 2 hours

Student study effort required:

Class contact:	
Lecture	28 hours
Tutorial	28 hours
Other study effort	
Self-study	30 hours
Homework assignment	10 hours
Project / Case study	20 hours
Total student study effort	116 hours

Student assessment:
Final assessment will be determined on the basis of:

Homework	20%	Project	30%
Mid-term	20%	Final exam	30%

Course assessment:
The assessment of course objectives will be determined on the basis of:

- Homework, project and exams
- Course evaluation

Course outline:

Weeks	Topic	Course work
1,2	**Introduction of OOA/D with UML** The concepts of object-orientation, object-oriented analysis and design, UML, and applying OOA/OOD with UML into a practical case study, software development process, process activities, and unified process.	Introduction of project
3,4	**Use Case Model** Use case model, use case diagram, use case definition, system operation sequence diagram, activity diagram, operation contract with pre and post conditions, and conceptual class diagram, and case studies.	Assignment#1 Assignment#2 Use case model of project
5, 6	**Static Design Model** Design class diagram, identification of classes, attributes and methods, generalization, aggregation, composition, and dependency relations, association, multiplicity and constraint.	Assignment#3 Static design model of project
7,8,9,10	**Dynamic Design Modeling** Design sequence diagram, activity diagram and state diagram, mapping design to codes, component and deployment diagrams, and case studies.	Assignment#4 Mid-test
11,12	**OCL and Design Patterns** OCL is introduced for specifying pre, post conditions and invariants formally. And the concepts of design patterns are introduced, such as designing objects with responsibility, expert pattern, creator pattern, controller, low coupling and high cohesion, façade and adapter patterns.	Assignment#5 Dynamic design model of project
13	**Project Presentation and Report**	Course project report

Contribution of course to meet the professional component:

This course prepares students to work professionally in the area of software development.

(a) An ability to apply knowledge of computing and mathematics appropriate to the programme outcomes and to the discipline

(b) An ability to apply knowledge of a computing specialisation, and domain knowledge appropriate for the computing specialisation to the abstraction and conceptualisation of computing models

(c) An ability to analyse a problem, and identify and define the computing requirements appropriate to its solution

(d) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs with appropriate consideration for public health and safety, social and environmental considerations

Relationship to CS program objectives and outcomes:

This course primarily contributes to Computer Science program outcomes that develop student abilities to:

Relationship to CS program criteria:

Criterion	DS	PF	AL	AR	OS	NC	PL	HC	GV	IS	IM	SP	SE	CN
Scale: 1 (highest) to 4 (lowest)		2					2				3		1	

Discrete Structures (DS), Programming Fundamentals (PF), Algorithms and Complexity (AL), Architecture and Organization (AR), Operating Systems (OS), Net-Centric Computing (NC), Programming Languages (PL), Human-Computer Interaction (HC), Graphics and Visual Computing (GV), Intelligent Systems (IS), Information Management (IM), Social and Professional Issues (SP), Software Engineering (SE), Computational Science (CN).

Course content distribution:

Percentage content for			
Mathematics	Science and engineering subjects	Complementary electives	**Total**
0%	100%	0%	100%

Coordinator:

Prof. Zhiguo Gong

Persons who prepared this description:

Prof. Xiaoshan Li

Part B General Course Information and Policies

2nd Semester of Year 2

Instructor:	Prof. Xiaoshan Li	Office:	E11-4015
Office Hour:	by appointment	Phone:	4471
Email:	xsl@umac.mo		

Time/Venue: Tue. 11:00am-13:00pm, Fri. 11:00am-13:00pm, Jan.-May, 2015

Grading Distribution:

Percentage Grade	Final Grade	Percentage Grade	Final Grade
100 – 93	A	92 - 88	A–
87 – 83	B+	82 - 78	B
77 – 73	B–	72 - 68	C+
67 – 63	C	62 - 58	C–
57 – 53	D+	52 - 50	D
below 50	F		

Comment:
The objectives of the lectures are to explain and to supplement the text material. Students are responsible for the assigned material whether or not it is covered in the lecture. Students who wish to succeed in this course should read the assignments prior to the lecture and should work all homework and project assignments. You are encouraged to look at other sources (other texts, etc.) to complement the lectures and text.

Homework and Course Project:
The project is the important part of this course. Through the project, students can apply the course contents they learn to the practical software system analysis and design. It will be very helpful for them to improve the analysis and design ability of object-oriented method. Project progress stage reports are requested to be delivered as homework assignments round two or three weeks with the progress of course contents course, and are presented, discussed and commented during tutorial classes and in instructor's office outside of class. Finally, each team should deliver their final course project report at the end of semester before final exam.

- The requirements will be announced and discussed in class.
- The students' progress on their project will be discussed in the tutorial class and instructor's office.
- The project will be presented twice formally at the middle and end of semester, and the final project report should be delivered before the final exam.

Note

- Recitation session is important part of this course and attendance is strongly recommended.
- Check UMMoodle (https://ummoodle.umac.mo/) for announcement, homework and lectures. Report any mistake on your grades within one week after posting.
- No make-up exam is given except for CLEAR medical proof.
- Cheating is absolutely prohibited by the university.

Appendix - Measurement Dimensions and Rubric for Program Outcomes (a), (b), (c), and (d)

(a) An ability to apply knowledge of computing and mathematics appropriate to the programme outcomes and to the discipline

Measurement Dimension	Excellent (80-100%)	Average (60-79%)	Poor (<60%)
1. An ability to apply knowledge of computing to the solution of complex computing problems.	Students understand the computing principles, and their limitations in the respective applications. Use the computing principles to formulate and solve complex computing problems.	Students understand the computing principles, and their limitations in the respective applications. But they have trouble in applying these computing principles to formulate and solve complex computing problems.	Students do not understand the computing principles, and their limitations in the respective applications. Do not know how to apply the appropriate computing principles to formulate and solve complex computing problems.

(b) An ability to apply knowledge of a computing specialisation, and domain knowledge appropriate for the computing specialisation to the abstraction and conceptualisation of computing models

Measurement Dimension	Excellent (80-100%)	Average (60-79%)	Poor (<60%)
1. An ability to apply knowledge of a computing specialisation, and domain knowledge to analyse and abstract complex computing models	Students understand the computing specialisation, and domain knowledge. They can also analyze and abstract complex computing models.	Students understand the computing specialisation, and domain knowledge. But they have trouble in analyzing and abstracting complex computing models.	Students have trouble in understanding the computing specialisation, and domain knowledge, and do not know how to analyze and abstract complex computing models.
2. An ability to apply knowledge of a computing specialisation, and domain knowledge to conceptualize complex computing models	Students understand the computing specialisation, and domain knowledge. They can also conceptualize complex computing models.	Students understand the computing specialisation, and domain knowledge. But they have trouble in conceptualizing complex computing models.	Students have trouble in understanding the computing specialisation, and domain knowledge, and do not know how to conceptualize complex computing models.

(c) An ability to analyse a problem, and identify and define the computing requirements appropriate to its solution

Measurement Dimension	Excellent (80-100%)	Average (60-79%)	Poor (<60%)
1. An ability to understand problem and identify the fundamental formulation	Students understand problem correctly and can identify the fundamental formulation	Student understand problem correctly, but have trouble in identifying the fundamental formulation	Students cannot understand problem correctly, and they do not know how to identify the fundamental formulation
2. An ability to choose and properly apply the correct techniques	Students know how to choose and properly apply the correct techniques to solve problem.	Students can choose correct techniques but have trouble in applying these techniques to solve problem.	Students have trouble in choosing the correct techniques to solve problem.

(d) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs with appropriate consideration for public health and safety, social and environmental considerations

Measurement Dimension	Excellent (80-100%)	Average (60-79%)	Poor (<60%)
1. An ability to design, implement, and evaluate a computer-based system, process, component, or program	Students understand how to properly design, implement and evaluate a computer-based system, process, component, or program	Students understand how to design and implement a computer-based system, process, component, or program, but have trouble in evaluating it.	Students do not know how to design, implement, and evaluate a computer-based system, process, component, or program