

Software Design

Code: 102759
ECTS Credits: 6

Degree	Type	Year	Semester
2502441 Computer Engineering	OB	3	1
2502441 Computer Engineering	OT	4	1

The proposed teaching and assessment methodology that appear in the guide may be subject to changes as a result of the restrictions to face-to-face class attendance imposed by the health authorities.

Contact

Name: Joan Serrat Gual
Email: Joan.Serrat@uab.cat

Use of Languages

Principal working language: catalan (cat)
Some groups entirely in English: No
Some groups entirely in Catalan: Yes
Some groups entirely in Spanish: No

Other comments on languages

All course materials (slides, handouts) are in English

Teachers

David Fernández Mota
Helena Bolta Torrell

External teachers

Rodolfo Guichón

Prerequisites

There are no official prerequisites, but the student must master the notation of UML class diagrams, the object orientation concepts and their programming in some language, preferably Java.

Objectives and Contextualisation

Once the purpose of a software has been determined through the analysis of requirements, software developers design a solution. This goes from the architecture of its components (classes, modules ...) to implementation in some programming language. This course covers mainly two key aspects of the design: object-oriented design patterns and user interface design. In addition to these, it deals with other subjects such as the design by contract of classes and methods and coding style.

The learning method is based on a project (the practicum) or put another way, 'learn to do': all concepts addressed in the theory are necessary in order to design and implement a software application, starting with a

statement that plays the role of requirements document. The practicum is about the design and implementation (but not requirements analysis, testing, version control or quality assurance, which belong to other courses) of an application.

Competences

Computer Engineering

- Capacity to design, develop, evaluate and ensure the accessibility, ergonomics, usability and security of computer systems, services and applications, as well as of the information that they manage.
- Have the capacity to conceive network technology based systems, applications and services, including Internet, Web, e-commerce, multimedia, interactive services and mobile computers.
- Have the capacity to conceive, develop and maintain computer systems, services and applications employing the methods of software engineering as an instrument to ensure quality.
- Have the capacity to identify and analyse problems and design, develop, implement, verify and document software solutions on the basis of suitable knowledge of current theories, models and techniques.
- Have the right personal attitude.
- Work in teams.

Learning Outcomes

1. Accept and respect the role of the various team members, and its different levels of dependence.
2. Conceive network technology based systems, applications and services, including Internet, Web, e-commerce, multimedia, interactive services and mobile computers.
3. Design a user interface based on customer specifications .
4. Design and document software solutions.
5. Develop curiosity and creativity.
6. Identify the paradigm that best adapts to a design problem in a specific computer software system taking into account the accessibility, ergonomic, usability and security requirements.
7. Identify, manage and resolve conflicts.
8. Know and apply the specific software components for communications based computational problems.
9. Maintain a proactive and dynamic attitude with regard to the development of ones own professional career, personal growth and continuing education. Be able to cope with adversities.
10. Specify the clients needs in a software specification document.
11. Work cooperatively.

Content

1. Principles of object orientation
2. Design patterns: creational, structural, behavioral
3. Design by contract
4. Style of coding and comments. Logging
5. Design of the user interface
 - usability
 - UX research
 - UX design : prototipat, user testing
 - internationalization and localization
6. Programming interfaces in Flutter

Methodology

As stated in the objectives, the subject follows a learning methodology based on the development of a project. Its concrete implementation with regard to the use of class hours is as follows:

Theory classes In them, the teacher introduces the theoretical contents and gives useful references (books,

articles, web pages) so that the student can then continue their learning, as well as develop the software application proposed. The slides used in these classes should not be taken as a source of study, but rather that they can be used to show the teacher in a more agile way. Instead, you need to study the recommended books, and articles, open access book chapters and others that we will indicate.

Problem classes. In these classes we will mainly present exercises, small design plus implementation problems, or the application of specific UX techniques. Depending on the year, we may also do some classes on problems of design patterns.

Practical sessions. The students will have formed at the beginning of the course small groups. In the practical classes students come to 1) show their work to the instructor and pose questions, and 2) given that the practice will be divided into milestones, the weeks that correspond to a milestone, it is evaluated The corresponding part in situ, according to a scale that will be included in the practicum handout. Therefore, the practical sessions are *not* for doing the practicum.

All the lecture notes, slides and more, plus the detailed schedule of the course are in Campus Virtual.

Annotation: Within the schedule set by the centre or degree programme, 15 minutes of one class will be reserved for students to evaluate their lecturers and their courses or modules through questionnaires.

Activities

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
practicum sessions	12	0.48	1, 2, 8, 4, 3, 7, 11
problems sessions	12	0.48	2, 8, 4, 3
theory lectures	26	1.04	2, 8, 4, 3, 6
Type: Autonomous			
individual study	29	1.16	8, 4, 3, 10, 6
preparation of practicum	52	2.08	1, 2, 8, 4, 3, 6, 7, 11
solving problems	12	0.48	2, 8, 4, 3, 10, 6

Assessment

The course follows a continued assessment system. This consists of a group assessment part and another individual assessment. The note of the first (PR) is obtained by making the marks of the practical goals (M1, M2, M3). The second (EX) of the average of two tests written during the quarter (EX1, EX2), or a retake exam (EXR1, EXR2). The suffixes 1 and 2 refer to the 2 parts of the course, before and after the first exam EX1.

The algorithm for calculating the final mark (FM) is the following (all marks are over 10):

```
// EX1, EX2 examens parcials
// EXR1, EXR2 examen de recuperació
// M1, M2, M3 notes de les 3 fites de pràctiques
// FM nota final
```

$EX = (\max(EX1, EXR1) + \max(EX2, EXR2)) / 2.$

if (M1>0) and (M2>0) and (M3>0) {

```

PR = (M1 + M2 + M3) / 3.
} else {
PR = 0.
}

if (EX >=5.) and (PR >=5.) {
FM =(EX + PR) / 2.
} else { // EX<5. o PR<5.
FM = min(EX, PR)
}

```

The practicum

The realization of the practicum is essential to assimilate the concepts of theory. The first thing we want to make sure is that the practicum has really been done by the group that delivers it, and that each member of the group has contributed to the same extent as the others. The second is that the practicum is done progressively during the course, in parallel with the theory. That's why we will do the following:

- The mark for the practice is obtained by combining the grade of the parts into which it is divided. Each part will be evaluated in situ in one of the practical sessions that we will announce at the beginning of the course (the milestones). The points that can be obtained from each part will be specified in the handout. These assessments have no retake, afterwards.
- On the evaluation days of the practicum (milestones) we will do a quick written validation test. If, as a result of the answers to this test or to the questions asked by the teacher during the assessment, we have doubts about the authorship or contribution of the group members, then the mark will change from group to individual and depending of how much and what part of the practice each member has done.

The exercises

We will propose some exercises and small problems that are optional. These exercises can be done individually or in small groups, the same or different than the practice. The exercises may add points to the theory note EX, with a top of 10. We say 'may' because obtaining these extra points is conditioned to a validation test in order to verify that the student has effectively completed the exercise he has delivered. We will do these validation tests only once, in the EX1 and EX2 exams, and they do not have retake afterwards.

Other

We do not preserve the mark of the practicum, exercises or exams from previous courses. Students taking again the course do not receive any differentiated treatment.

If an exam or part of the practicum is not delivered, the mark for that part is zero. The final grade will be 'no-avaluable' if nothing to be evaluated has been delivered, either exam or practicum.

Milestones are assessments equivalent to written exams. If a student does not show up, the mark is zero.

MH marks will be awarded at the discretion of students with a final final grade of 9.0 and taking into account all kinds of work done (exams, practicum and exercises).

The dates of continuous evaluation and delivery of works will be published on the virtual campus of the subject and may be subject to change in response to possible incidents. It will always be reported through the virtual campus, which will be the communication mechanism between teachers and students.

Forwritten exam-type assessment activities, a place, date and time of revision will be indicated in which the student will be able to review the activity with the teacher. If the student does not show up for this review, this activity will not be reviewed later.

Notwithstanding other disciplinary measures that are deemed appropriate, and in accordance with the current academic regulations, the irregularities committed by the student who can lead to a variation of the qualification of an act of self- evaluation. Therefore, plagiarizing, copying or letting copying the practice or any other evaluation activity will imply fail with a zero and can not be recovered in the same academic year. If this activity has a minimum associated mark, then the student won't pass the course.

Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes
group assessment	50%	4	0.16	2, 8, 5, 4, 3, 10, 6, 9
individual assessment	50%	3	0.12	1, 2, 8, 5, 4, 3, 10, 6, 7, 9, 11

Bibliography

Part 1

- Design patterns: elements of reusable object oriented software. E. Gamma, R. Helm, R. Johnson, J. Vlissides. Addison Wesley, 1994.
- Design patterns explained simply. Alexander Shvets. <https://sourcemaking.com/design-patterns-ebook>.
- Head first design patterns. E. Freeman, E. Freeman, K. Sierra, B. Bates. O'Reilly, 2004. 2nd edition appeared in 2020.
- Clean code: a handbook of agile software craftsmanship. R.C. Martin. Prentice Hall, 2008.

Part 2

- The user experience team of one. A research and design survival guide. Leah Buley. Rosenfeld, 2013.
- Understanding your users. A practical guide to user research methods. Kathy Baxter, Catherine Courage, Kelly Caine. Morgan Kaufmann, segona edició 2015.
- Killer UX design. Jodie Moule. Sitepoint, 2012.
- Validating product ideas. Tomer Sharon. Rosenfeld, 2016.
- Rocket surgery made easy: the do-it-yourself guide to finding and fixing usability problems. S. Krug. New Riders, 2010.

Software

Programming languages: Java, Dart (

Programming environments: IntelliJ IDEA (Java, Flutter), plugins Checkstyle, plantUML. Library logback

Other :

- prototyping applications like marvelApp, inVision, Adobe XD etc.
- OptimalWorkshop for user-testing and other UX techniques