



# Syllabus

## Design & Tests 1



E.T.S. de Ingeniería Informática

UNIVERSIDAD DE SEVILLA

# Contents

- Coordinates
- Context
- Goals and competences
- Contents
- Methodology
- Evaluation & Grading
- Plan
- Resources
- Tools & Reminders

# Coordinates

# Coordinates

## Theory Lectures

- Thursday, 10:40 – 12:30, A1.13 [G1]
- Thursday, 12:40 – 14:30, A1.10 [G2]
- Thursday, 15:30 – 17:30, A1.13 [G3]
- Tuesday, 17:40 – 19:30, A1.10 [English]

## Laboratory Lectures

- Tuesday, 08:30 – 10:20, A4.32, F1.32 [G1: L1,L2]
- Tuesday, 08:30 – 10:20, F1.33 [G1&G2: L3]
- Tuesday, 10:40 – 12:30, F1.30, F1.33 [G2: L4,L5]
- Tuesday, 17:40 – 19:30, F1.31, F1.32, F1.33 [G3: L6,L7,L8]
- Thursday, 17:40 – 19:30, B1.35 [English Lab]

# Teaching Staff



Ana Belén Sánchez  
[anabsanchez@us.es](mailto:anabsanchez@us.es)  
Office I0.63



Bedilia Estrada  
[iestrada@us.es](mailto:iestrada@us.es)  
Office F1.83



José M. García  
[josemgarcia@us.es](mailto:josemgarcia@us.es)  
Office F0.42



Carlos Müller  
[cmuller@us.es](mailto:cmuller@us.es)  
Office F0.43



José A. Parejo  
(Coordinator)  
[japarejo@us.es](mailto:japarejo@us.es)  
Office I0.71



Manuel Resinas  
[resinas@us.es](mailto:resinas@us.es)  
Office F0.47

# Context

# Software Engineering Degree

---

Basic Foundations

---

Computer Programming

---

Software Engineering, Information Systems and Intelligent Systems

---

Operating Systems, Networks and Distributed Systems, Computers Architecture

---

Projects

---

**Specific Technology about Software Engineering**

---

Mandatory Complementary Topics about Software Engineering

---

Optative Complementary Topics about Software Engineering

---

Other Optative Complementary Topics

---

Practicum

# Module M06 – Specific Technology about Software Engineering





# M06 – Specific Technology about Software Engineering

2º - 2C

- Arquitectura e Integración de Sistemas Software

3º - 1C

- Diseño y Pruebas 1
- Ingeniería de Requisitos
- Proceso Software y Gestión 1

3º - 2C

- Diseño y Pruebas 2
- Proceso Software y Gestión 2

4º - 1C

- Evolución y Gestión de la Configuración

4º - 2C

- Ingeniería del Software y Práctica Profesional

# Goals and competences

## Goal

Learn to **develop and test** a small/medium size web information system based on an informal requirements specification and using a set of tools and components used by the industry

## General competences

G05. Ability to design, develop, and maintain systems, services, and applications using the methods of Software Engineering as a tool for quality assurance.

G09. Ability to solve problems and make decisions with initiative, autonomy, and creativity. Ability to learn to communicate and get the knowledge across, skills, and abilities of the Technical Computing Engineer profession.

## Specific competences

E25. Ability to develop, maintain, and evaluate software systems and services that satisfy the user requirements and behave reliably and efficiently, are affordable to develop and maintain, and meet quality standards, applying the theories, principles, methods, and practices of Software Engineering.

E29. Ability to identify, evaluate, and manage potential risks.

E30. Ability to design appropriate solutions in one or more application domains using Software Engineering methods that integrate ethical, social, legal, and economic issues.

# Learning outcomes



Understand the importance and limitations of software design testing



Design and implement web information systems, and test cases



Automate the execution of tests



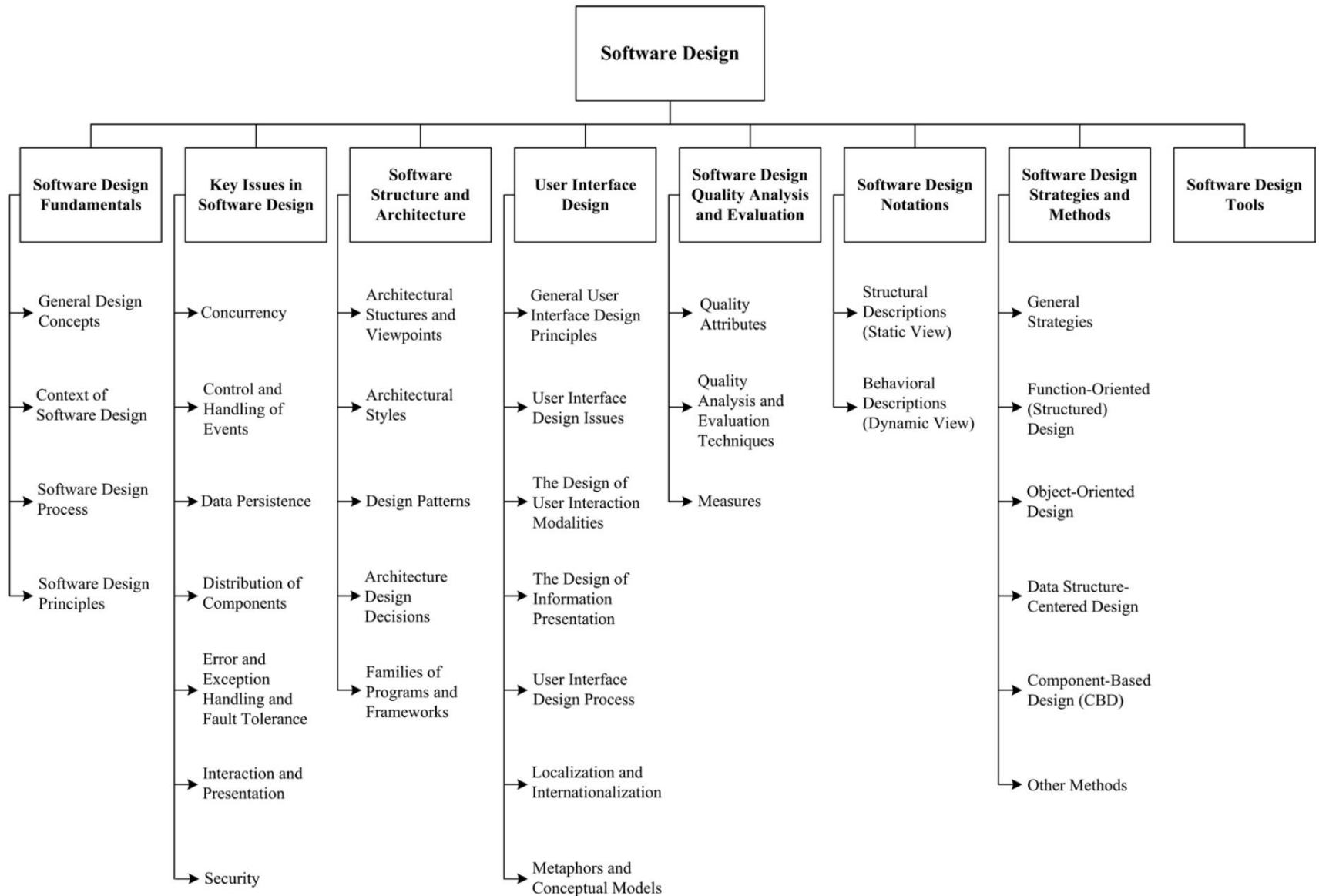
Evaluate different design alternatives and choose among them in an informed way



Understand the implications of architectural and OO design on software properties such as: performance, maintainability, testability, etc.

# Contents

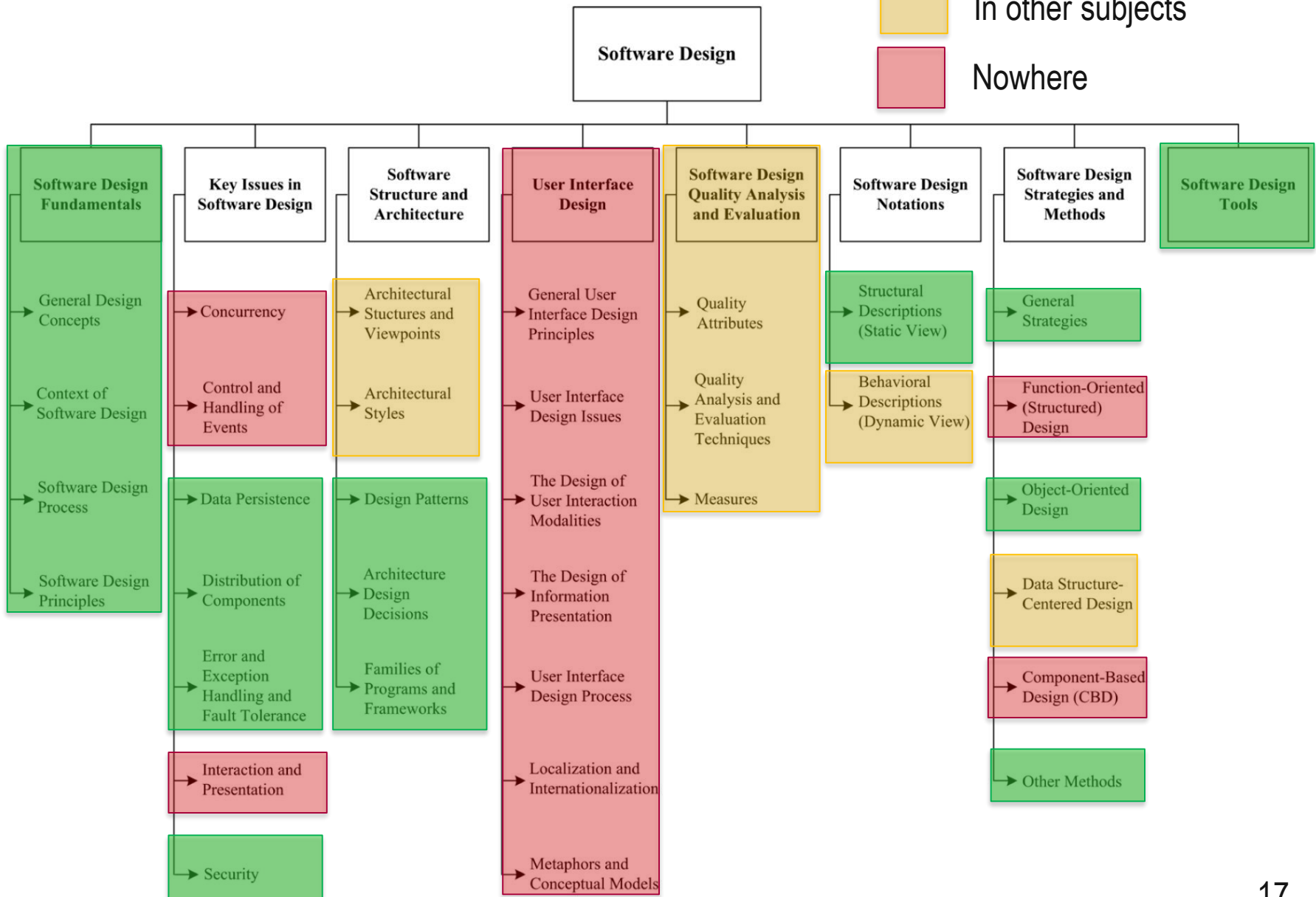
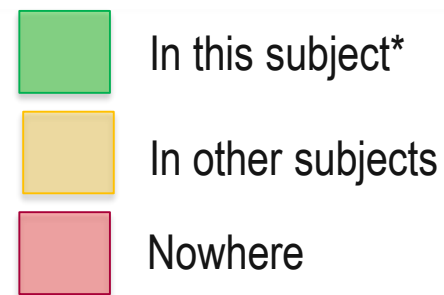
# Contents (Design)



**Figure 2.1.** Breakdown of Topics for the Software Design KA

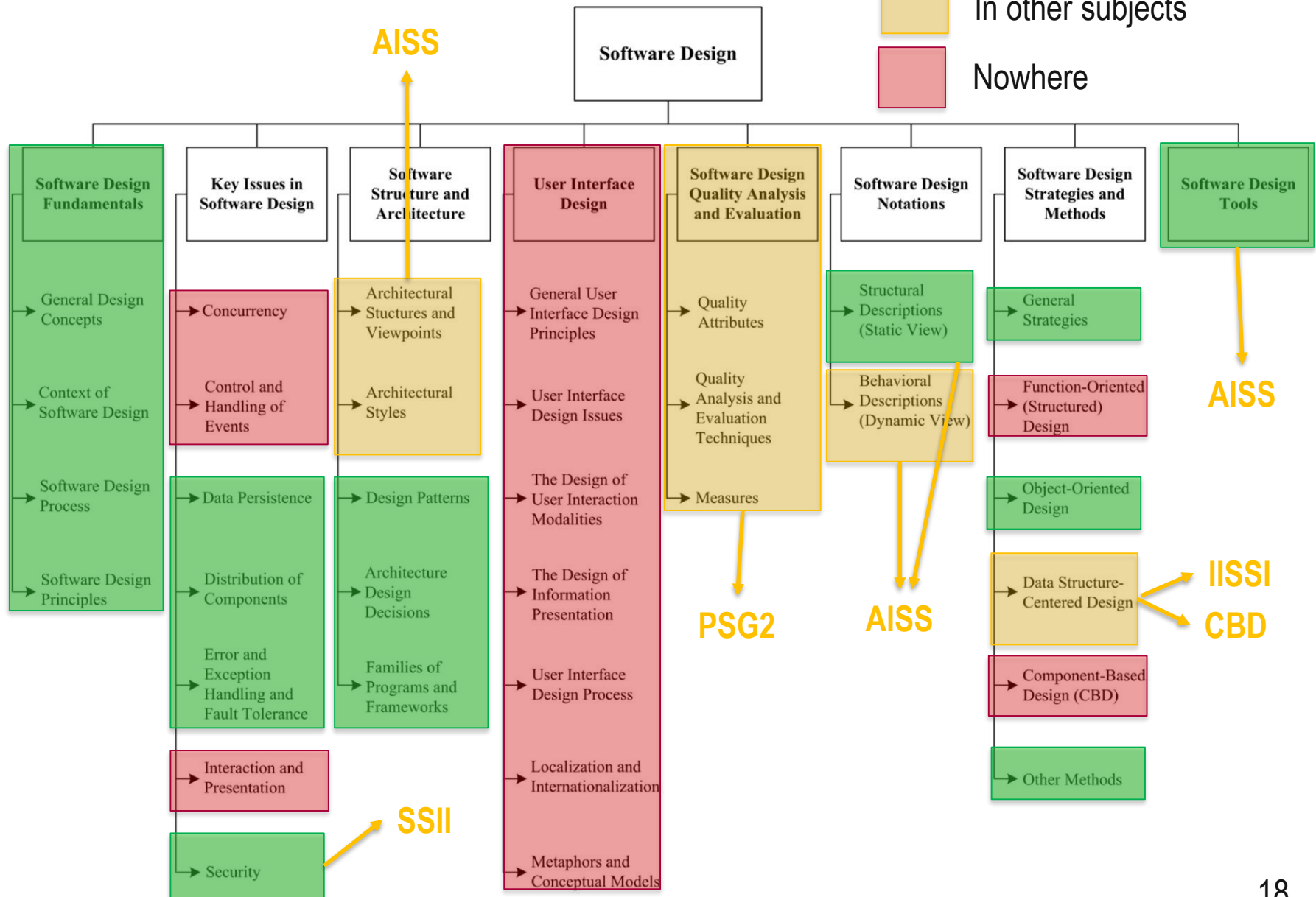
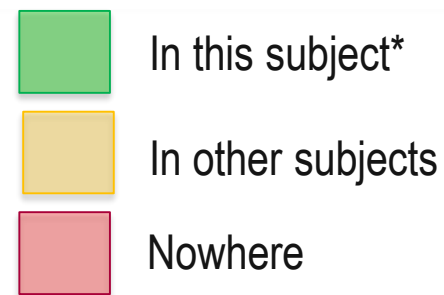


# Contents (Design)



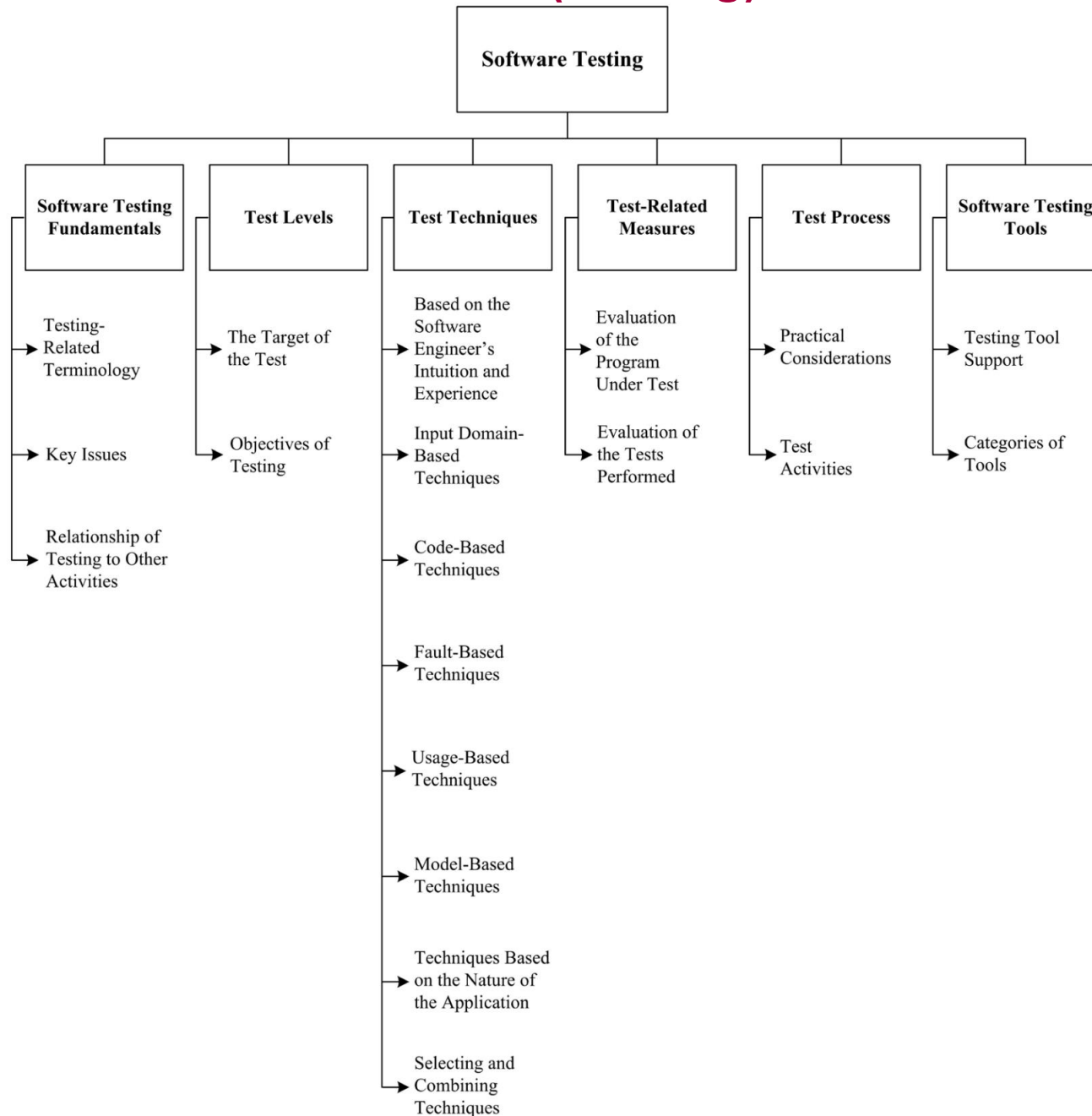
**Figure 2.1.** Breakdown of Topics for the Software Design KA

# Contents (Design)



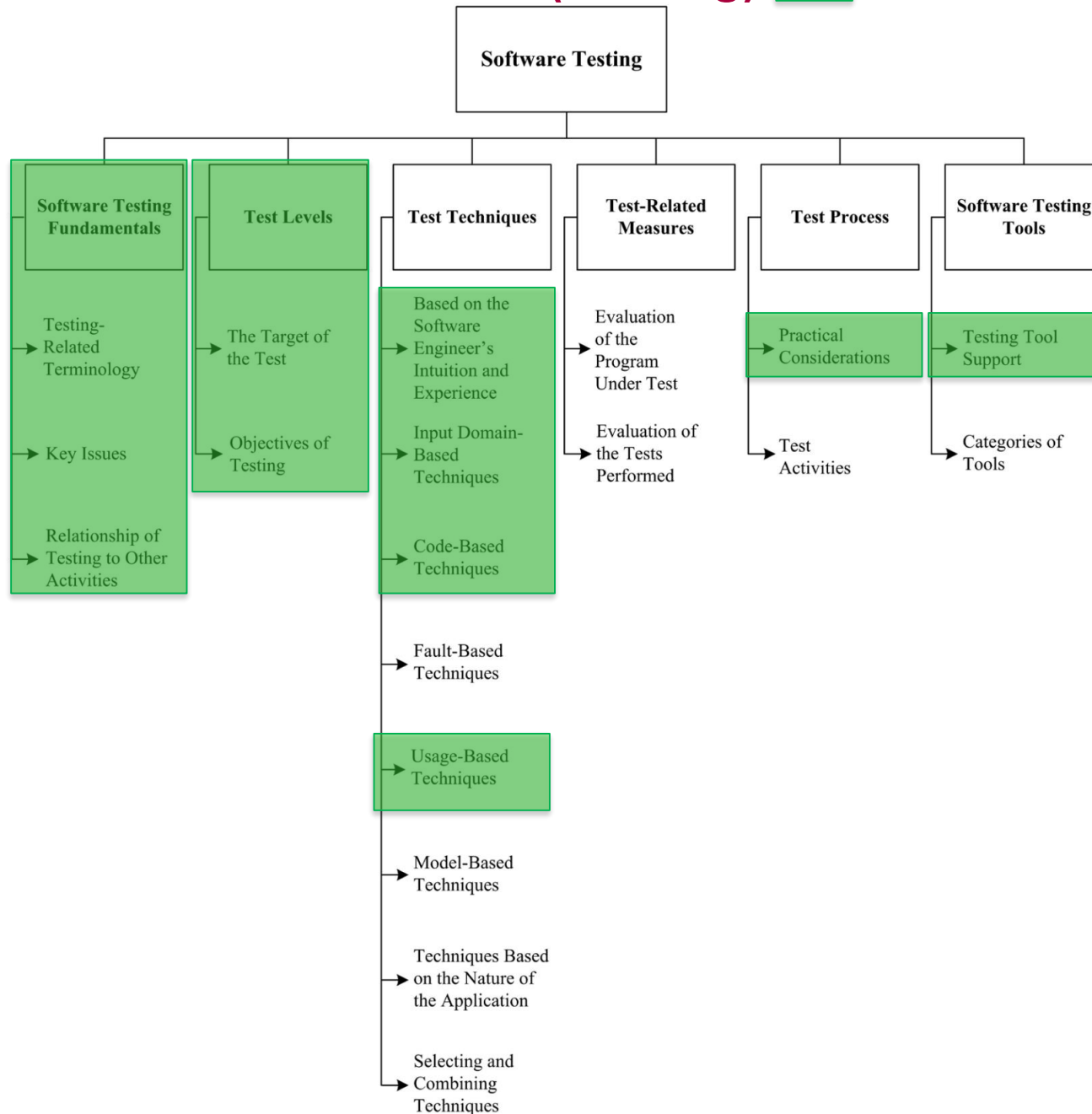
**Figure 2.1.** Breakdown of Topics for the Software Design KA

# Contents (Testing)



# Contents (Testing)

In this subject\*



# Contents

---

- T1- Introduction to software design and testing
  - T2 - Introduction to the design of Presentation Layers
  - T3 - Introduction to the design of Backend Business Logic & Resources Layers
  - T4 - Introduction to data model design
  - T5 - Data Model Design
  - T6 – Information systems design
  - T7 - Introduction to information systems testing
  - T8 – Information systems testing
  - T9 – Refactoring
  - T10 - Design patterns
  - T11- Advanced data model design and testing
  - T12 - Design of reliable, and secure information systems
-

# Methodology

# IT'S A PROJECT-BASED WORLD

## OF BOARD GAMES!

+

## COMMON MODULES



... maybe you could even reach to the [DP1 HALL OF FAME](#)

# Activities



Theory Lectures

Basic concepts and problems



Laboratory Lectures

Tools

Project follow-up / feedback



Homework & technical videos




Consulting hours



# FLIPPED TEACHING

## IN CLASS

### LAB SESSION




Watch Videos  
Get feedback  
Ask for advice  
Fix bugs and  
discuss design  
decisions

### THEORY SESSION



Learn  
theory  
Deep dive on  
the technology  
Live coding  
sessions  
Answer micro-tests



Watch Videos  
Develop  
your project  
working  
with your  
partners



Watch Videos  
about tech. &  
practice  
Do exercises  
experiment  
study



## OUT OF CLASS

# We will perform 4 sprints on your project

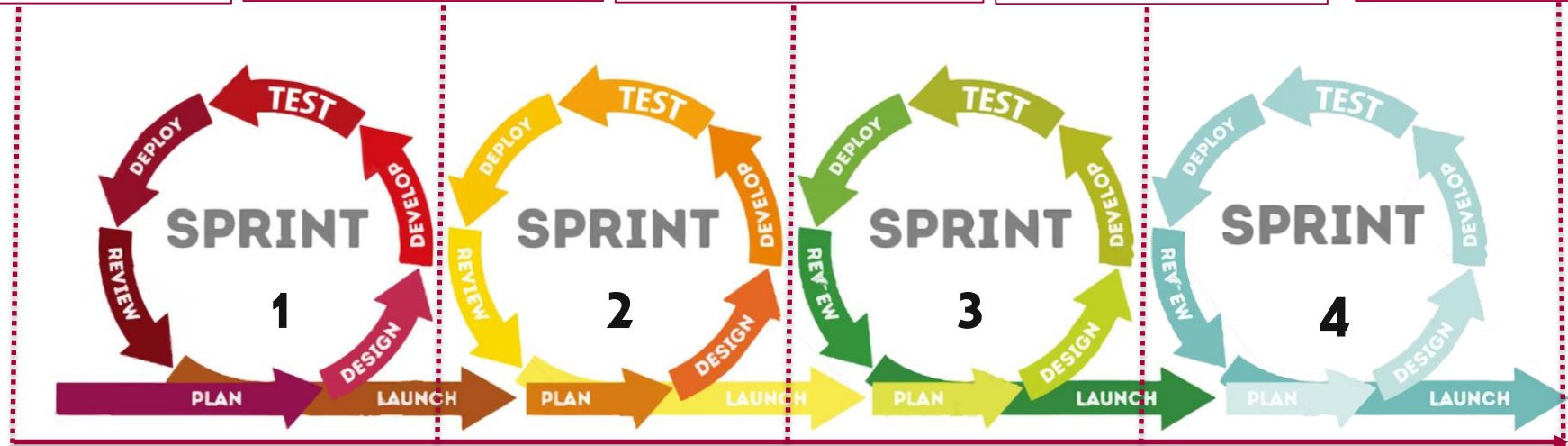
**20 Sept.**  
Group formation  
& Game chosen

**8 Oct.**  
Requirements Doc.  
& Initial project

**12 Nov.**  
Req. Doc. Review  
& 50% impl. & tests

**10 Dic.**  
Design Doc &  
75% impl. & tests

**15 Jan.**  
Final  
delivery



We will have review lab session at the end of sprints 1, 2, & 3 with feedback:



# Evaluation and grading

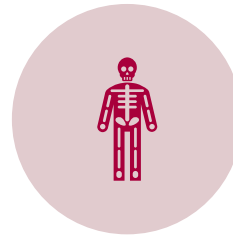
# Grading items



MICRO-TESTS



DELIVERABLES



CONTROL-CHECKS  
(THEORY & LAB)



WORK REPORT  
(PRACTICE)

# Grading Matrix for the project

Additional features	Type of game	
	Simple	Intermediate
No extra module	$\leq 7$	$\leq 8$
1 extra module	$\leq 8$	$\leq 9$
2 extra modules	$\leq 9$	$\leq 10$
2 extra modules + A <sup>+</sup> tasks	$\leq 10$ + eligible for MH	$\leq 10$ + eligible for MH

\* All the use cases in the basic game functionality must work without any error!  
Otherwise, your grade will be at maximum 4

\*\* If any use case of a module does not work, the module will not be taken into account!

# Board Games - Common modules

- Users & Admin interface (Required):
  - Login, Logout, Sign-up.
  - Admin: Registered users (with pagination)
  - Admin: CRUD of users (with delete on cascade of games, movements, etc.).
- Game (Required):
  - Match lobby
  - Match creation
  - Matches listing
  - ....
- Statistics (Optional)
  - Number of matches (global & per user)
  - Duration of matches (global & per user, averages, total, max and mins).
  - Number of players per game (if it makes sense)
  - **Game-specific stats** (points, moves, chosen characters, favourite cards, etc.)
  - Achievements
- Social gaming (Optional)
  - Friendship invitation, management & current friends online notification
  - Game invitations
  - Public comments/chats during games
  - Spectator mode

# Deliverables

- Four deliverables
  - D1: Requirements and analysis documentation
  - D2: Design documentation
  - D3: Test plan and test coverage documentation
  - D4: Project in github
- Each graded between 0 – 10
- The global project grade is:  $0.1 \cdot D1 + 0.2 \cdot D2 + 0.1 \cdot D3 + 0.6 \cdot D4$
- Check the deliverables document for more details

## Work report (Practice)

- **Individual** report in which students are requested to describe their contribution to the project.
- **Explicitly enumerate** the entities, relationships, and user stories **implemented by you** and those in which you collaborated with other partners.
- That information will be **matched with the commits** present in the **git repository** of the Project
- Graded as PASS or FAIL





# Sprint feedback matters



- One red **Or** two yellows



Not eligible for MH



- Two reds **Or** three yellows



You descend one step down in the grading matrix

## Self-grading per Sprint

- Each member will have up to ( $\#members - 1$ ) points to assign in each sprint
  - E.g.: in a group with 6 members each member can assign up to 5 points.
- The final grade will be weighted taking into account the total points obtained by each member.

IndividualPonderationFactor (IPF) =  $\text{TotalPoints} / (4 * (\#members - 1))$

Individual Project Grade = Global Project Grade \* IPF

**Note:** IPF will never take a value below 0.7 or above 1.2.

## Self-grading example

If

Individual Ponderation Factor of John is 0.7

And

the Global Project Grade of his group is 7 points



Then

the Individua Project Grade of John is  $7.0 \times 0.7 = 4.9$



**Note:** As teachers, we can dissolve the group and perform an individual evaluation (even after project submission)

→ Hence the need of a report of the work performed by each member of the group and a deep scrutiny of GitHub commits

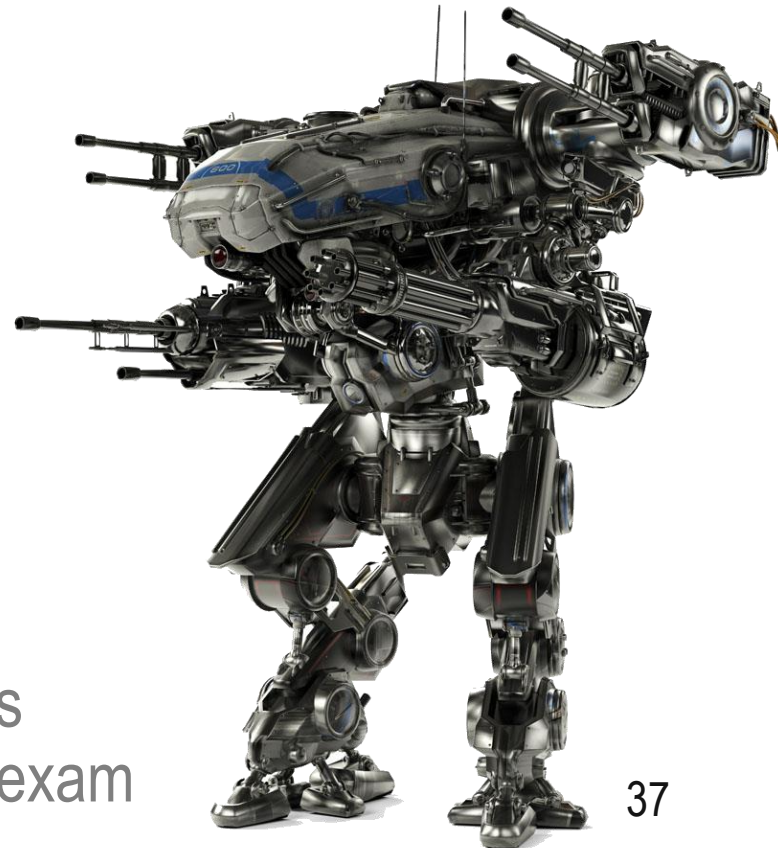
# Theory (Control-check + Micro-tests)

- 25% micro-questionnaires at the beginning / end of theory sessions (WooClap)
- 75% Test of short questions about the theoretical concepts studied during the course (EV)
- No additional material can be used during the control-check
- Theory control-check (19/12/2024) and final exam (23/01/2025)



# Control Check (Laboratory)

- You will have to implement certain functionality (business logic, services, controllers, repositories, etc.)
- **Grading based on test case execution** (unit & integration tests)
  - ➔ **You must be extremely careful and execute the tests prior to submission!**
- The test suite will be available during the control check
- We will provide an example (**24/10/2024**).
- We will have two partial lab-control checks (**5/11/2024** and **17/12/2024**) and the final exam



## Final Grade

If

- Theory Grade  $\geq 4$  &
- Practice Control-check Grade  $\geq 4$

Then

$$\text{Final grade} = 0.2 * \text{Theory Grade} + 0.15 * \text{Practice Control-check Grade} + 0.65 * \text{Min}(\text{Individual Project Grade}, 10)$$

Otherwise

- Final Grade is the minimum between 4 and the grade of the deliverables

## Final Grade (Recovery)

- If the grade of theory or practice is less than 5, the student can retake the control-checks (Theory or practice) during the final exams period in January.
- In this case, the final grade will be computed in the same way, but using the last grades obtained for the control-checks.

## Grading in July and November

- The grading items and grading system will be exactly the same (except for theory, that is computed using the control-check only)
- The dates for the control-checks and the deliverables will be the dates set for the July and November evaluation respectively
- We will retain the grading of the control-checks and the Project in July and November **if its value is  $\geq 4$**



# Grading Examples

JOHN

G  
I  
V  
E  
N

Evaluation item	Grade
Lab control-check 1	7
Lab control-check 2	5
Theory control-check	6
Final Lab control-check	-
Final Theory control check	-



Lab control-check Grade: **6**  
Theory control-check Grade: **6**

I  
F

Evaluation item	Grade
Project	3

**THEN**



Final grade is **4,7**, in July John will present only a new version of his project, and his theory & lab grades will be **6**

# Grading Examples

ANNA

G  
I  
V  
E  
N

Evaluation item	Grade
Lab control-check 1	2
Lab control-check 2	5
Theory control-check	2



Practice control-check Grade: **3,5**  
Theory control-check Grade: **2**

I  
F

Evaluation item	Grade
Final Lab control-check	3
Final Theory control check	4
Project	6



Practice control-check Grade: **3**  
Theory control-check Grade: **4**  
Project Grade: **6**



**THEN**

Final grade **4**

In July will do the Lab control-check only. Her July Theory Grade will be **4** and her July. Project Grade will be **6**

# Plan

# Course Plan

Month		Lab Session		Theory Session	SPRINT	Events
September	10	T0 - Presentación + Norma proyecto	12	T1 - Introduction to Software Design		Groups creation <b>13 Sept</b>
	17	L1 - Crash course on Spring boot and Github	19	T2 - Introduction to the design of Presentation		Game choice and decision on modules to develop <b>20 Sept</b>
	24	L2 - Frontend: Listings and Forms	26	T3 - Introduction to the design of Backend Bu	SPRINT 1	
October	1	L3 - Feedback	3	T4 - Introduction to data model design		
	8	L4 - Review	10	T5 - Data Model design		
	15	L5- Live coding session -Implementing CRUD with SPRING & REACT ( incl. Basic	17	T6 - Information Systems Design	SPRINT 2	10% User Stories, Analysis and Design documents <b>8 Oct</b>
	22	L6 - Feedback	24	T7 - Introduction to Information Systems Testin		
	29	L7 - Feedback	31	T8 - Information Systems Testing		
November	5	L8 - Control Check	7	T9 - Refactoring	SPRINT 2	Control Check 1 (5 Nov)
	12	L9 - Review	14	T10 - LiveCodingSession - Applying desing patterns on your projects		
	19	L10 - Live coding session - Implementing Tests	21	T10 - LiveCodingSession - Applying desing patterns on your projects 2		
	26	L11 - Feedback	28	T10 - LiveCodingSession - Applying desing patterns on your projects 3	SPRINT 3	
December	3	L12 - Feedback	5	T11 - Advanced data model design		
	10	L13 - Review	12	T12 - Design for traceability, reliability, and se	SPRINT 3	75% User Stories , Analysis, Design and Test docs <b>10 Dic</b>
	17	L14 - Lab Control Check	19	Theory Control Check		
	...	CHRISTMAS				
	January	...				
15	Project			SPRINT 4	Full projectProject Deadline (15 Jan 2025)	
23	Final Control Checks					Final Theory and lab Control Checks (23 Jan 2025)

# Resources

# Teaching program and projects

- You can find them at: <https://www.us.es/estudiar/que-estudiar/oferta-de-grados/grado-en-ingenieria-informatica-ingenieria-del-software/2050048>

# Enseñanza Virtual



## Resolución de problemas de acceso

Guía de resolución de posibles problemas en el acceso a aplicaciones Web de la US (Enseñanza Virtual, SEVIUS, ...). Puede acceder a la guía pinchando [aquí](#).

## Contactar

La creación de los espacios en Enseñanza Virtual de las asignaturas de docencia oficial, y las inscripciones de los profesores y los estudiantes en esos espacios, se realiza de forma **automatizada** a partir de la información registrada en los Sistemas de Gestión Académica.

## Doble factor en EV

Activación del [doble factor](#) para el acceso a EV

## Inicia sesión en EV



Utilice esta opción para identificarse sólo una vez y poder acceder a otros servicios de la Universidad de Sevilla, además de la Enseñanza Virtual.

Para acceder a EV debe usar su "Usuario EV" que puede consultar en <https://sevius.us.es> (Mi perfil)

## Anuncios

### Disponibilidad de SafeAssign en Enseñanza Virtual

*(miércoles 24 de julio de 2024)*

Estimado/a profesor/a,

Se va a realizar un mantenimiento en SafeAssign el día 31 de julio de 2024, por lo que el servicio no funcionará en ese período. El mantenimiento se realiza fuera de horas punta, y puede llegar a tardar 7 horas.

# Tools & Reminders



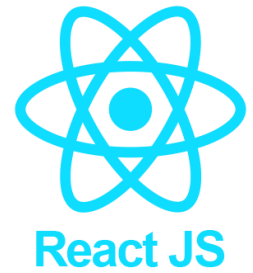
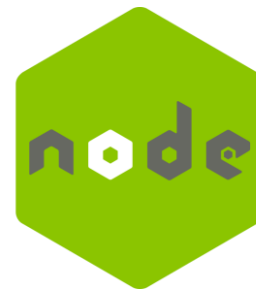


- We will use git for managing the source code and publishing changes (to other group members)
  - You should never exchange source code through other mechanisms, such as e-mail, pendrives, dropbox, etc.
- We recommend you installing:
  - The git client for your OS: <https://git-scm.com/downloads>
- You can learn Git & GitHub fundamentals by completing this optional activity:  
[https://classroom.github.com/a/7O6fug\\_V](https://classroom.github.com/a/7O6fug_V)



- We will use java as the programming language for the development of the backend in our projects, and the Java VM as its execution platform
- You should install a Java JDK with version  $\geq 17$
- We recommend you installing from:  
<https://www.oracle.com/java/technologies/downloads/#java17>

# Node & React



- We will use JavaScript and node.js for the development of the frontend in our projects
- You should install Node.js **18 LTS**
- We recommend you installing it from: <https://nodejs.org/en>
- React will be installed automatically by the project as a library (but you can install the corresponding plugins in your favourite IDE to ease its use)

# Build Tools & IDEs / Code Editors



- We recommend you installing MAVEN
  - Follow this guide: <https://maven.apache.org/install.html>
- Your favourite IDE/Code editor
  - We (teachers) use [Visual Studio Code](#) with the [Java extension pack](#)
  - If you want to use Eclipse (we don't like it 😞), we recommend you using the latest version of the Eclipse IDE for Enterprise Java Developers: <https://www.eclipse.org/downloads/packages/release/2022-06/r/eclipse-ide-enterprise-java-and-web-developers>

# Lombok



- Lombok is a java library that automatically plugs into the IDE/editor and build tools.
- It automates the generation of:
  - trivial getters/setters.
  - constructors (taking no arguments, one argument per final / non-null field, or one argument for every field)
  - logging fields/variables
  - ...
- **You have a video on how to install it (on Eclipse and VS Code) at EV**



- You can use whatever UML-compliant modelling tool you like
- In the project template we use [Mermaid](#) and [PlantUML](#)
- Some alternatives:
  - As students you have access to [MS Visio](#)
  - [Astah](#) is a good desktop editor with free licenses for students
  - If you prefer online editors and you are a fan of plant text files you can use [PlantTex/PlantUML](#)
  - Finally, [Papyrus](#) is the most popular option for UML modelling in Eclipse
  - [UMLet](#) is also a good option (with a compatible online editor). We used it for the examples and materials of this course.
- **Each group should agree on a common tool**

# That's All

- No server
- No database
- No virtualization
- If you want to use a different technology (such as Angular or vue), **ask your lab teacher.**



## Reminder 1: Github username survey in EV

- If you don't have a GitHub user, please register (using your UVUS preferably) and **submit the username through the survey in EV.**

# GitHub



**DEADLINE: 15 September!**



## Reminder 2: You must choose a group for the project!

- **Groups with 6 members** are preferred (the volume of work to be performed is the same for groups with 1 or 6 members).
- All the members of the group must **inscribe themselves in the corresponding Laboratory Group** in EV.

**DEADLINE: 13 September!**



## Reminder 3: Each group must choose a game

- We will provide a link to a spreadsheet where each group will **choose among the available games** (in EV).
- For each theory group (G1, G2, G3, G<sub>long</sub>), **no game will be chosen by more than one project group** (no games repeated in a theory group) → **Run!!!!**
- You should take into account game complexity

**OPENING OF GAME CHOICE  
FORMS: 18 Sept 8:00 am**



**DEADLINE:  
20 September!**



## Reminder 4: Each project group must play a game and create a video explaining the rules

- Upload a video explaining the rules to the GitHub repository of the project or Youtube.



**DEADLINE: 27 September!**

## Reminder 5: Read, sign & upload your learning commitment contract

- It is an agreement designed to outline the goals, expectations, and responsibilities of students participating in a specific group project.
- It contains an explicit statement of the game you choose as a group, the grade you aim for, and the modules that you plan to develop
- All the members of the group must read and sign it and upload (in pdf format) to the docs folder of your project repo (upload one single document signed by all the members)

**DEADLINE: 8 October!**

**Questions?**

