FACULDADE ENGENHARIA
Departamento de Informática

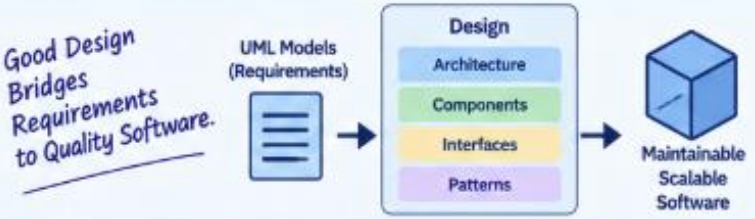



# Engenharia de Software
# (14341, 16230, 15386)

## L08: Software Design

# L08

# Practical Lab #8
## Software Design
### From Models to a Solution Architecture




## Today's Competency Readiness Indicators

By the end of this lab, you will be better able to:

**Define a software architecture**

Translate requirements and UML models into a coherent software architecture, identifying main components, layers and their responsibilities.

**Design components and interfaces**

Specify component structure, responsibilities and interfaces, applying design principles (e.g., low coupling, high cohesion) and design patterns where appropriate.

**Allocate functionality to components**

Map use cases and class models to components, defining their interactions, dependencies and interfaces.

**Evaluate design alternatives**

Compare alternative design solutions considering quality attributes such as maintainability, scalability, performance and reusability.

**Ensure design consistency**

Verify that the design is consistent with requirements and previous models (e.g., use cases, class diagram, activity and sequence diagrams).

**Communicate and justify design decisions**

Document and present the design using standard notations (e.g., UML component and deployment diagrams) and justify the main architectural and design choices.

**Related SFIA Competencies (main areas)**

- **DESN** Systems Design
- **ARCH** Enterprise and Solution Architecture
- **SCTY** Software Construction
- **DEPI** Deployment and Implementation

# SMART CAMPUS

Engineering the Digital University

# L08 – SOFTWARE DESIGN

*Goal: Transform the validated models into a robust, maintainable and user-centred software design.*

## PART 1 Project Brief

10 min

**CLIENT REQUEST #5**

"We need a well-structured and maintainable solution for the Smart Campus Reservation system."

Your task:
Design the software solution by defining the architecture, main components and interfaces.

Think about:

- How will the system be organised?
- What are the main modules?
- How will components interact?
- What data do we need to manage?
- Which patterns or principles will guide the design?

**TODAY'S DELIVERABLES**

- Software Architecture (high level)
- Design Decisions
- Class Diagram (Domain Model)
- Updated Sequence & State Diagrams (if design impacts them)



## PART 2 Core Project

60 min

Design the solution that will implement the required functionality.

1. **Define Architecture**
   Choose the architectural style (e.g., Layered, MVC, Microservices) and justify your choice.
2. **Identify Components**
   Define the main packages/modules and their responsibilities.
3. **Define Interfaces**
   Specify how components will interact (APIs, services, events).
4. **Define Data Model**
   Refine the domain model into classes, attributes, relationships and constraints.
5. **Apply Design Principles**
   Apply SOLID, DRY and other principles to improve quality and maintainability.

**DELIVERABLE**

Software Architecture (high level)
+ Design Decisions

## PART 3 Engineering Challenge

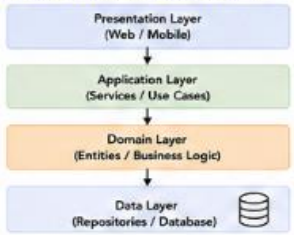
25 min

**DESIGN CHALLENGE**

Answer the following:

1. What is the overall architecture of the system? Why is it appropriate?
2. What are the main components/modules and their responsibilities?
3. How do components communicate?
4. Which design patterns are applied and why?
5. How is data managed and persisted?
6. How does the design support scalability, security and maintainability?

**EXAMPLE: HIGH-LEVEL ARCHITECTURE**




**DELIVERABLE**

Class Diagram (Domain Model)
+ Updated Sequence & State Diagrams
(if design impacts them)

## PART 4 AI Challenge

20 min

Use an LLM (e.g., ChatGPT) to support your design decisions.

**SUGGESTED PROMPT**

"We are designing the software for the Smart Campus Reservation system. Based on our requirements and validated models, propose a suitable architecture, main components, interfaces and design patterns. Justify your choices and highlight trade-offs."



Evaluate the AI suggestions critically and adapt them to our context.

**CHECKLIST**

- [x] Is the architecture appropriate?
- [x] Are components responsibilities clear?
- [x] Are interfaces well defined?
- [x] Are design principles respected?
- [x] Are quality attributes addressed?

## PART 5 Reflection & Discussion

15 min

Discuss with your team and be ready to share.

- Which design decisions were the most important? Why?
- What trade-offs did you consider during the design?
- How does this design support future changes and new requirements?
- What did you learn today about designing quality software?

**ENGINEERING INSIGHT**

Good design reduces complexity, improves understandability and speeds up future development. Invest time in design to save time in implementation.

## PROJECT MILESTONES

**1 Requirements Engineering**

- [x] Stakeholders
- [x] Features
- [x] Personas
- [x] Scenarios
- [x] User Stories

**2 Requirements Analysis**

- [x] Use Case Specifications
- [x] Use Case Diagram
- [x] Class Diagram (Domain Model)

**3 Dynamic Modelling**

- [x] Activity Diagram
- [x] Sequence Diagram
- [x] State Diagram

**4 Software Design**

- [x] Software Architecture
- [x] Design Decisions

**Legend**

- [x] Completed milestone
- [ ] Pending milestone

## TIPS FOR SUCCESS

- Start from validated requirements and models.
- Keep the design simple and focused.
- Use design patterns when they add value.
- Think about quality attributes (performance, security, scalability, maintainability).
- Document your assumptions and decisions.

## DESIGN PRINCIPLES (REMINDER)

- Single Responsibility Principle (SRP)
- Open/Closed Principle (OCP)
- Liskov Substitution Principle (LSP)
- Interface Segregation Principle (ISP)
- Dependency Inversion Principle (DIP)
- DRY – Don't Repeat Yourself

## DIAGRAMS REFERENCE

- Class Diagram: structure of the domain model
- Sequence Diagram: interactions over time
- State Diagram: lifecycle of key objects
- Activity Diagram: workflows and processes

## TOOLS YOU CAN USE

- diagrams.net (draw.io)
- Lucidchart
- Visual Paradigm Online
- PlantUML
- miro

## TODAY'S SPRINT GOAL

Deliver a solid and well-justified software design that will guide the implementation and support the evolution of the Smart Campus Reservation system.

# QUESTIONS?

Let's clarify, explore and learn together!




**No question is too small!**
Your questions help everyone learn and grow.

Ask freely, challenge ideas, and let's build understanding **together!**