EN
English

Universidad de Oviedo

School of Computer Science

Software Architecture Presentation

SOFTWARE ARCHITECTURE

Course 2020/2021

Jose E. Labra Gayo

Software Architecture

Degree: Computer Science - Software Engineering
Type: Mandatory, third year
Credits: 6
Period: 2nd Semester
Language: English/Spanish

Campus virtual: https://www.campusvirtual.uniovi.es/
Mostly for internal communications
Web page of course: https://arquisoft.github.io/
Slides and public content

Lecturers

Hugo Lebredo Buján	lebredohugo@uniovi.es	
Irene Cid Rico	cidirene@uniovi.es	
Pablo González González	gonzalezgpablo@uniovi.es	
Jose Emilio Labra Gayo	labra@uniovi.es	(Coordinator)

Time dedication

- 6 ECTS credits ≈ 150 working hours
  - 60 on-campus hours, 90 self-study
- Organization (*by week*)
  - 2h lectures (21h total)
  - 1h seminars (7h total)
  - 2h laboratory practice (28h total)
  - 2h group tutories on demand
  - 7,5h self-study (90h total)

Competences & learning outcomes

General competences

Methodological skills

CG-1	Ability to design solutions to human complex problems

Specific competences

Common to Computer Science

Com.1	Ability to design, develop, select and evaluate applications and systems, ensuring their reliability, safety and quality, according to ethical principles, laws and regulations.
Com.8	Ability to analyse, design, build and maintain applications in a robust, secure and efficient way, and choosing the most suitable paradigms and programming languages.
Com.11	Knowledge and application of features, functionality and structure of distributed systems, computer networks and the Internet, and to design and implement applications based on them.

Specific competences

Software Engineering

ISW.1	Ability to develop, maintain and evaluate software systems and services that match all user requirements and behave reliably and efficiently, being affordable to develop and maintain and accomplishing quality standards, applying the theories, principles, methods and Software Engineering good practices.
ISW.3	Ability to solve integration problems in terms of strategies, standards and available technologies.
ISW.4	Ability to identify and analyse problems and to design, develop, implement, verify and document software solutions based on adequate knowledge of the theories, models and techniques.

Learning outcomes

RA.IS-1.	Making complex Software Engineering Projects that provide solutions to complex problems and to solve them using techniques and technologies related to manufacturing processes, including software frameworks, architectural patterns, design and integration patterns, pursuing quality software development
RA.IS-3.	To apply different construction techniques in designing low level software
RA.IS-4.	Develop design and object-oriented programming with a high level of competence
RA.IS-5.	To evolve and refactor existing designs to afford changing requirements
RA.IS-6.	Determining the degree of maintainability, reliability and efficiency of software designs
RA.IS-7	To design and implement software using different middleware technologies
RA.IS-9	To design and to carry out checks and efficient and effective inspections about validation, verification, quality and test plans.
RA.IS-10	Statistically analysing the density of defects and failure probability
RA.IS-11	Evaluating the quality of a software process from the point of view of product quality.

Evaluation & grading

3 possibilities

Continuous evaluation
Differentiated assessment
Extraordinary evaluation

Continuous evaluation

$$Final = Theory \times 40\% + Practice \times 60\%$$

where:

$$Theory = Exam \times 70\% + Intermediate\ exercises \times 30\%$$

$$Practice = Team \times 70\% + Individual \times 30\%$$

Requirements:

- Minimum assistance (80%)
- Minimum mark (theory & seminar): 5
- Minimum mark (lab): 5

Differentiated evaluation

Theory: The same as previous

Practice (2 possibilities)

1) Working in a team (minimal assistance 20%)

   Mandatory: Participate in public presentation session

   The mark will be: 70% team + 30% individual.

2) Working individually

   During the first month the student will be assigned a project similar to the teams projects

   Individual public presentation

***General remark**: Assignments that are not done or not delivered on time will count as 0*

Extraordinary Evaluation

IF continuous evaluation fails

Final mark

$$Final = Theory \times 40\% + Practice \times 60\%$$

where

$Theory = Exam + Individual\ work\ (seminar)$

$Practice = Individual\ project$

Both theory and practice marks must be ≥ 5

Public presentation of Individual Project

Usually after the final exam

Teaching activities

Lectures

Conferences

Seminars

Laboratory sessions

Lectures

- This year, we will use *flipped classroom* methodology
  - All lessons recorded and available:
    - https://arquisoft.github.io/course2122.html
  - I expect you to watch the lessons before each session
- During the class:
  - Explain some concepts with more detail
  - Questions and answers
  - Use Kahoot! for feedback (scores will be part of the mark)

Conferences

- We will organize several conferences during the course
- Software architecture experts
- Conferences from past years are available at:
  - https://arquisoft.github.io/#Conferences
- Attendance is mandatory
- The content from conferences is part of the course
  - We can include questions from conferences in the exams

Seminars

- Team work
  - Teams of 2-4 people
- Subjects proposed by teachers
- The work consists of
  - A small report about the subject (like a blog post)
  - A presentation of the topic to the class
- Public presentations during the seminars
  - At least 2 questions posed by other teams
- Assessment:
  - Report delivered + Presentation + Questions

Laboratory sessions

Project based learning + team work

1. Design and document a software architecture
2. Implement prototype
3. Public presentation

Assessment

70% team mark+ 30% individual mark

- Team mark: Presentation + prototype + docs + github repo
  - Participation in final presentation is mandatory (like an exam)
  - Teachers select the person(s) that do the presentation
  - Other team members can participate
- Individual mark: github contributions
  - Project management tool: github.com

Important: Create your github account
If possible, use a login name that resembles your first name/last name...

About the teams

- Teams created initially by the teachers
  - Size: 5-8 people
  - Teams will work together during the whole course
  - Being able to work within a team is a very important skill
    - Members that abandon will fail continuous evaluation
    - In case of problems within a team:
      - Try to understand & solve the problem
      - If unsolvable communicate to the teachers as soon as possible
      - Possible actions:
        - A problematic person is removed from the team and will fail continuous evaluation
        - All the team members fail continuous evaluation
        - The team is split

Lab sessions

- 13 lab sessions
  - During the lab sessions (2 hours)
    - 1.- A teacher will explain some concept (1 hour approx.)
    - 2.- The team will work on the project
      - That hour counts as a team meeting
      - Teams can do more extra meetings if they want

Team meetings

- Mandatory: Keep record of all team meetings
  - One person must write the minutes
    - Advice: Rotate the role of scribe
  - Minutes must be maintained in the project wiki
  - General structure of minutes:
    - Date/time/place of meeting
    - Participants
    - Register
      - Tasks done (closed github issues)
      - Actions to do (open github issues assigned to someone)
      - Agreements/decisions (maintain *Architecture decision records*)

4 Deliverables

Checkpoint at every deliverable

1st deliverable - Week 4

- Documentation 0.1

2nd deliverable - Week 7

- Prototype version 0.1

3rd deliverable - Week 10

- Prototype version 1.0 + Documentation 1.0

4th deliverable - Week 13

- Prototype version 1.1 + Documentation 1.1
- Public presentation

Before each deliverable
- Each student will write a self-report
- Including a short overview of the tasks done
- Linking it to github issues
- The rest of the team must say if they agree or not

Public presentation

- Last laboratory session
- It acts as a Practical Exam
  - Participation is mandatory
- Each group will present their project to the teachers
  - The teachers select the presenter(s)

Material to follow the course

- Web page: Slides and public information
  - https://arquisoft.github.io/
- Virtual campus (internal information)
  - Forum
  - Other material
    - Manuals, Tutorilas, Videos, etc…

This year's assignment

DeDe: https://arquisoft.github.io/course2122/labAssignmentDescription.html

Decentralized Delivery system

Based on SOLID principles

- Optional participation in Solid Challenge

This year: React + Typescript

About SOLID

- SOLID (SOcial LInked Data)
  - Goal: Decentralized Social Web
    - Separate personal data from apps
  - Project started at MIT
- It uses several W3C specifications
  - WebId
  - Web Access Control
  - Linked Data Platform
  - …
- You must read/learn about that by yourself
  - Lots of materials available

If you have questions…

- About the course…
  - Deadlines, exams, mandatory tasks, etc.
  - Please use the Campus Virtual forum
    - The message will arrive to the rest of the students
    - Every one can see the question and the answer
    - Every one can even answer
- About technical matters…
  - Use public places
    - StackOverflow (general): https://stackoverflow.com/
    - Solid forum (about solid): https://forum.solidproject.org/
- About personal problems or similar questions
  - Send an email to the teacher

Important dates

Assignment deadlines

- 1er deliverable (4th class. 21 - 25 Feb.)
- 2º deliverable (7th class. 14 - 18 March)
- 3er deliverable (10th class. 4 - 8 April)
- 4º delivarable **MANDATORY** (13th class, 3 - 9 May)

Theory exams

- Ordinary May/June??
- Extraordinary July??

End of presentation

Schedule

Subjects	Total hours	Onsite activities						Offsite activities		
		Lectures	Seminars	Lab sessions	Group tutories	Assessment sessions	Total	Work group	Individual work	Total
1 (Concepts)	**40**	7	2	4	1	1	**15**	15	10	**25**
2 (Taxonomies)	**110**	14	5	24	1	1	**45**	45	20	**65**
Total	**150**	**21**	**7**	**28**	**2**	**2**	**60**	**60**	**30**	**90**