Software Architecture

Lecture 01
Introduction

Néstor Cataño
Innopolis University

Spring 2016

Software Architecture

Goals

- To be effective software engineers
- To manage complexity in large software systems
- To design, implement, verify and maintain efficient systems
- To work in a development team
- To use principles, methods, and good practices in software architectures, software engineering, and software design for developing software

Software Architecture

What's a Software Architecture?

"The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them"

Software Engineering

What's Software Engineering?

- Software Engineering is the discipline concerned with the building of software systems that meet the requirements of users and customers.

Software Engineering

What's Software Engineering?

- Software Engineering is the discipline concerned with the building of software systems that meet the requirements of users and customers.
- The development of correct, usable and effective software systems is one of the skills that Software Engineering students should acquire

Architecture and Engineering

Architecture and Engineering

- Traditional engineering processes have stable architectures: buildings, airplanes, cars, ships, bridges, etc.
- These stable architectures evolved by:
  - trail and error
  - reusing and refinement previous solutions
- Traditional engineering achieved
  - definition of new engineering process
  - standardization of engineering methods
  - production of new materials

Software Engineering

More difficult?

- Constructions are invisible
  Products/outputs are logical rather than physical
- Huge need of evolution
- Products must adapt
- Rapid evolution of underlying technologies

Software Systems

Software Systems

- They are intangible
- They do not obey physical constraints
- They can become extremely complex, difficult to to understand, and expensive to change

Quoted from Sommerville's Book

There are many types of software systems: its pointless to look for universal notations, methods, or techniques for Software Engineering because different types of software require different approaches

Pre-Requisites

Programming Experience

- Previous exposure to a programming language like Java, C, C♯, Eiffel, etc.

Pre-Requisites

Object Oriented

- What's O-O Programming?
- What's inheritance, encapsulation, polymorphism, etc?

Grading

Activity	%	**When**
Project	25 %	Deliverable 1: due Week 05 Deliverable 2: due Week 10 Deliverable 3: due Week 16
Homework	25 %	(almost) Every week
Mid-term	20 %	Week 09
Final exam	30 %	Week 18

Course Project

Avatar Library - Sustainable Energy Consumption

Deliverable 1	functional and non-functional requirements
Deliverable 2	software architecture of the Avatar library
Deliverable 3	implementation, contracts, software patterns

Project Statement

Check Moodle for a complete version of the Project Statement

Labs and Homeworks

- Labs are often used to work in homeworks
- Homeworks are always due the following week before the next class starts

Course Info

Instructor

Name	Nestor Catano
Email	n.catano@innopolis.ru
Web	http://poporo.uma.pt
Lecture	Monday from 9:00 AM to 10:30 AM, Room 108
Offfice Hours	Tuesday from 9:00 AM to 11:00 AM; otherwise, drop me an email

Course Info

Teaching Assistants

Rafael Bogaveev	r.bogaveev@innopolis.ru
Alexandr Naumchev	a.naumchev@innopolis.ru
Alexander TChitchigin	a.chichigin@innopolis.ru
Evgeny Shakhmaev	bigblackbugg@gmail.com
Timur Shakirov	shakirovtr23@gmail.com

Course Info

Groups

Labs	Group 1 (BS), 318, 10:40-12:10	Rafael Bogaveev
	Group 2 (BS), 307, 10:40-12:10	Alexandr Naumchev
	Group 3 (BS), 312, 10:40-12:10	Alexander TChitchigin
Labs	Group 4 (BS), 318, 13:20-14:50	Evgeny Shakhmaev
	Group 5 (BS), 307, 13:20-14:50	Timur Shakirov
	Group 6 (MS), 312, 13:20-14:50	Rafael Bogaveev

Course Structure

Week 1	Introduction
Week 2	Software Requirements
Week 3	Software Life-Cycles
Week 4	Principles of Agile Soft. Dev.
Week 5	UML
Week 6	Software Testing
Week 7	Software Architecture
Week 8	Architectural Patterns

Course Structure

Week 9	Midterm
Week 10	SOLID
Week 11	ADTs, Modularity
Week 12	Design-by-Contract (DBC)
Week 13-15	Design Patterns
Week 18	Final Exam

Literature

Books and Articles

1. Software Architecture in Practice, Third Edition, by Len Bass, Paul Clements and Rick Kazman
2. Pattern-Oriented Software Architecture: A System of Patterns, by Frank Buschmann et al., WILEY.
3. Software Engineering, by Ian Sommerville, 9th Edition, Pearson Education.
4. Agile Principles, Patterns, and Practices in C$\sharp$, By Robert C. Martin and Mica Martin. Prentice Hall, 2006.
5. Object-Oriented Software Construction, by Bertrand Meyer, Prentice-Hall, 1997

Literature

Books and Articles

1. Design Patterns: Elements of Reusable Object-Oriented Software, by E. Gamma, R. Helm, R. Johnson, and J. Vlissides
2. A Discipline of Programming, by Edsger W. Dijkstra, Prentice-Hall, 1976.
3. Testing Computer Software, Second Edition, by Cem Kaner and Jack Falk, Wiley
4. UML Distilled: A Brief Guide to the Standard Object Modeling Language, 3rd edition, by Martin Fowler.

Next Class

Topics Covered

- What are software requirements?
- How to write software requirements?

Today's Lab

Topics Covered

- Basics about Java on Eclipse
- Homework is due next week (25/Jan before 9AM)
- Use Moodle to turn in your homework
- Moodle will allow you 2 submission attempts only