# Software Design (614G01015)

## 2nd Year. Degree in Computer Engineering

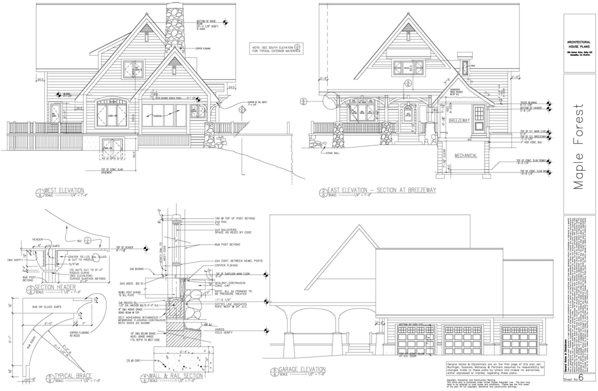
**David Alonso Ríos**
Eduardo Mosqueira Rey (Coordinator)

Department of Computer Science and Information Technology
Faculty of Computer Science

UNIVERSIDADE DA CORUÑA

# Design

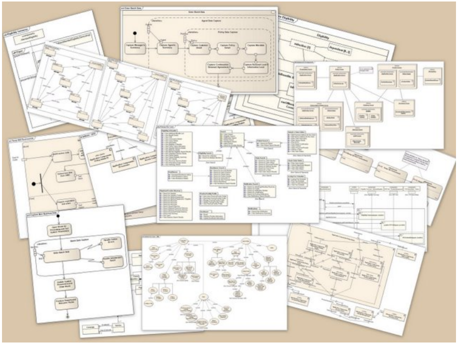
- The construction of any complex element requires a prior design to be carried out successfully.



# Design

## Software Design (IEEE)

The process of defining the architecture, components, interfaces and other characteristics of a software system or component.



# Design

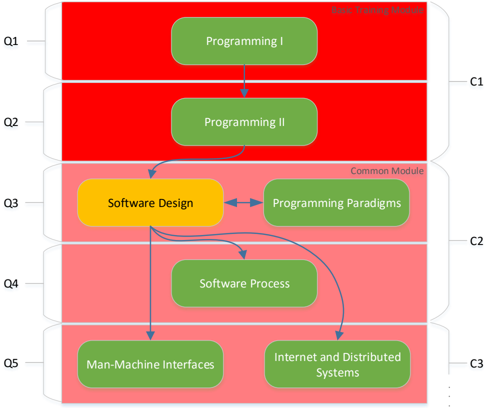
## Object-Oriented Design

Design of a program based on objects that exchange messages

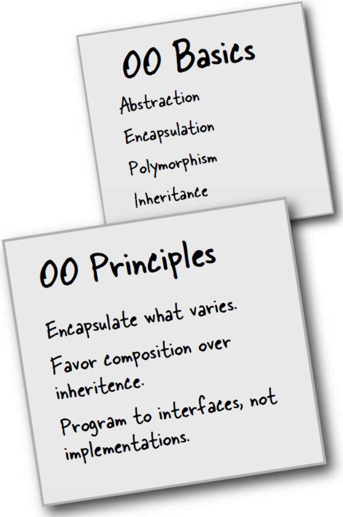
## A cat is object-oriented

After all, a cat exhibits characteristic behavior, responds to messages, is heir to a long tradition of inherited responses, and manages its own quite independent internal state

# Software Design in the Degree in Computer Engineering



# Units



OO Basics

Abstraction

Encapsulation

Polymorphism

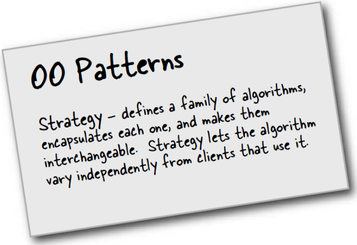
Inheritance

OO Principles

Encapsulate what varies.

Favor composition over inheritence.

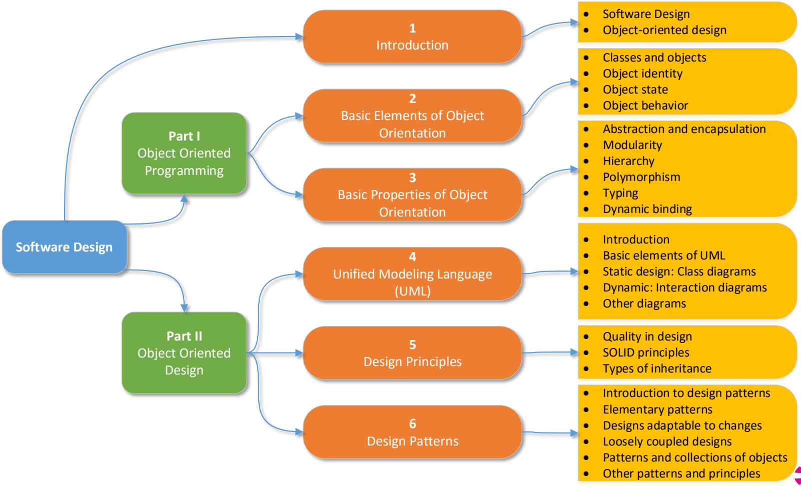
Program to interfaces, not implementations.



OO Patterns

Strategy – defines a family of algorithms, encapsulates each one, and makes them interchangeable. Strategy lets the algorithm vary independently from clients that use it.

# Units

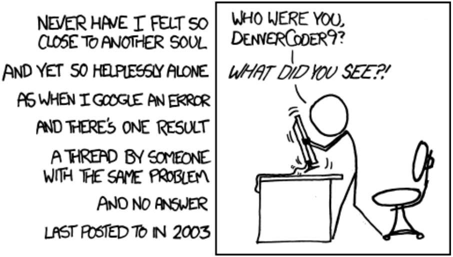

Software Design
Part I
Object Oriented Programming
Part II
Object Oriented Design
1
Introduction
2
Basic Elements of Object Orientation
3
Basic Properties of Object Orientation
4
Unified Modeling Language (UML)
5
Design Principles
6
Design Patterns
Software Design
Object-oriented design
Classes and objects
Object identity
Object state
Object behavior
Abstraction and encapsulation
Modularity
Hierarchy
Polymorphism
Typing
Dynamic binding
Introduction
Basic elements of UML
Static design: Class diagrams
Dynamic: Interaction diagrams
Other diagrams
Quality in design
SOLID principles
Types of inheritance
Introduction to design patterns
Elementary patterns
Designs adaptable to changes
Loosely coupled designs
Patterns and collections of objects
Other patterns and principles

# Classroom Organization

- **Theory Classes**
  - They will be taught in a **remote** way.
  - They will be held synchronously in the class schedule set in the calendar.
  - They will be recorded in *MS Stream* with their corresponding links in *Moodle*.
- **Practice Classes**
  - They will be taught in a **face-to-face manner** in the classroom.
  - Strict separation (we cannot leave our sites).
  - We will use telematic means for problem solving.

# Evaluation

- **Laboratory practice (40 %)**:
  - Exercises about Java, object-oriented programming, testing, design principles, design patterns, etc.
  - The assessment of practices represents the continuous evaluation of the subject.
  - Plagiarism **may result in a grade of zero, both for the original and for the copy**.


NEVER HAVE I FELT SO CLOSE TO ANOTHER SOUL
AND YET SO HELPLESSLY ALONE
AS WHEN I GOOGLE AN ERROR
AND THERE'S ONE RESULT
A THREAD BY SOMEONE WITH THE SAME PROBLEM
AND NO ANSWER
LAST POSTED TO IN 2003
WHO WERE YOU, DENVERCODER9?
WHAT DID YOU SEE?!

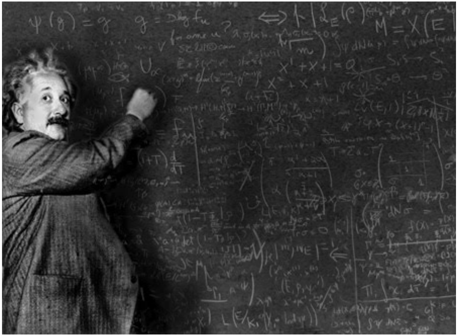
http://xkcd.com/979/

# Evaluation

- **Seminars**:
  - Practical classes designed to explain concepts necessary for the laboratory practices.
  - They are not graded. They help you to solve the assignments, which *are* graded.

# Evaluation

- **Objective Test (60 %)**:
  - Written test conducted at the end of the semester with theoretical and practical content.
  - Obtaining a **minimum grade of 4** is required in order to pass. A failure to reach the minimum score will mean that you cannot get more than a 4.5 in the final grade.
  - **Example**: Identifying patterns and principles,understanding how OO works in a given code, etc.



# 2nd Opportunity

- Percentages are the same as those of the first opportunity. The rule of a minimun grade of 4 in the objective test to pass the course also applies.
- Laboratory practices grades are the ones obtained at the first opportunity (**submission of laboratory practices in the second opportunity is not allowed**).

# Contingency Plan

- The percentages indicated in the above slides correspond to a face-to-face evaluation.
- **If a face-to-face evaluation cannot be carried out, the weight of the objective test will be reduced and the minimum score will be eliminated**, leaving the percentages as follows:.
  - **Non-presential objective test: 40 % (no minimum score)**
  - **Laboratory practices: 60 %**
- The non-presential objective test will be a practice (preferably) or a test type if the appropriate conditions do not exist to carry out said practice.

# Websites

- **Virtual Campus (Moodle)**
  - URL: `https://campusvirtual.udc.gal/`
  - Slides, forums, practices, links, etc.
  - You can request a temporary anonymous access while you do not have a valid user.
- **Microsoft Teams**
  - URL: `https://teams.microsoft.com/`
  - Non-presential classes and tutorials, etc.
  - For the tutorials respect the schedule that can be consulted at `https://espazos.udc.es/tutorials/search`.

# Basic references

- Sierra, K., Bates, B. “Head First Java (2nd ed.)”, O’Reilly, 2005.
- Schildt, H. “Java 9”, Anaya Multimedia, 2018.
- Booch J., Rumbaugh J., Jacobson I. “Unified Modeling Language (2nd ed.)”, Addison Wesley, 2006
- Martin, R.C. “UML for Java Programmers”, Pearson, 2004
- Gamma, E., Helm, R., Johnson, R., Vlissides J. “Design Patterns: Elements of Reusable Object-oriented Software”, Addison Wesley, 1996

# Complementary references

- Eckel, B. "Thinking in Java (4th ed.)", Prentice-Hall, 2007
- Rumbaugh J., Jacobson I., Booch J. "The Unified Modeling Language: Reference Manual", Addison Wesley, 2004
- Bloch, J. "Effective Java ($3^{rd}$ ed.)", Addison Wesley, 2017.
- Martin, R.C. "Clean code : a handbook of agile software craftsmanship.", Prentice-Hall, 2009.
- Larman C. "Applying UML and Patterns, 3rd edition", Prentice-Hall, 2005.
- Freeman E., Freeman E., Bates B. "Head First Design Patterns", O'Reilly, 2004
- Grand M. "Patterns in Java", John Wiley & Sons, 2002

# Software Design (614G01015)

## 2nd Year. Degree in Computer Engineering

**David Alonso Ríos**
Eduardo Mosqueira Rey (Coordinator)

Department of Computer Science and Information Technology
Faculty of Computer Science

UNIVERSIDADE DA CORUÑA