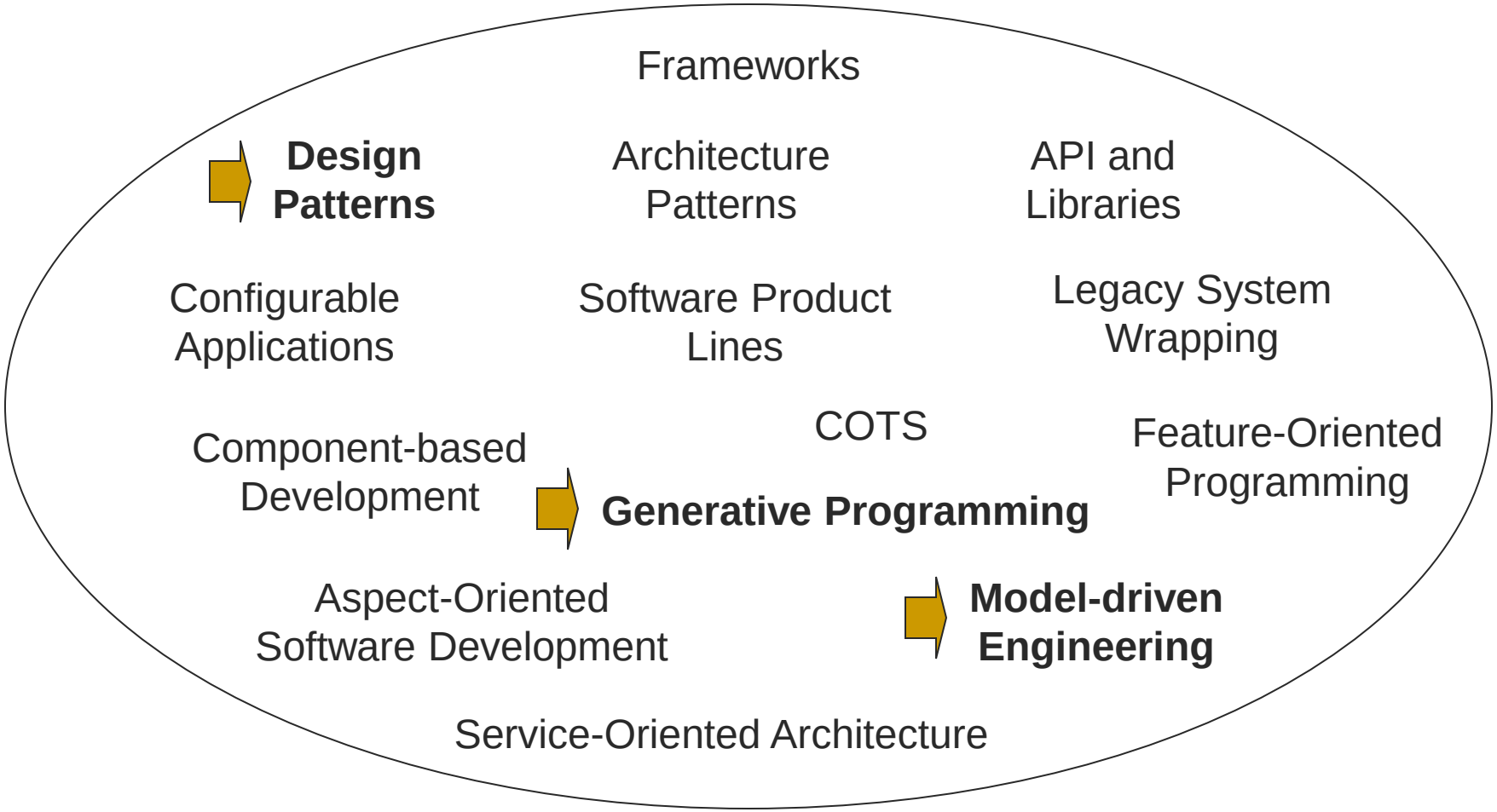
Software Reuse Techniques

Eduardo Figueiredo

http://www.dcc.ufmg.br/~figueiredo

The Reuse Landscape



Design Patterns

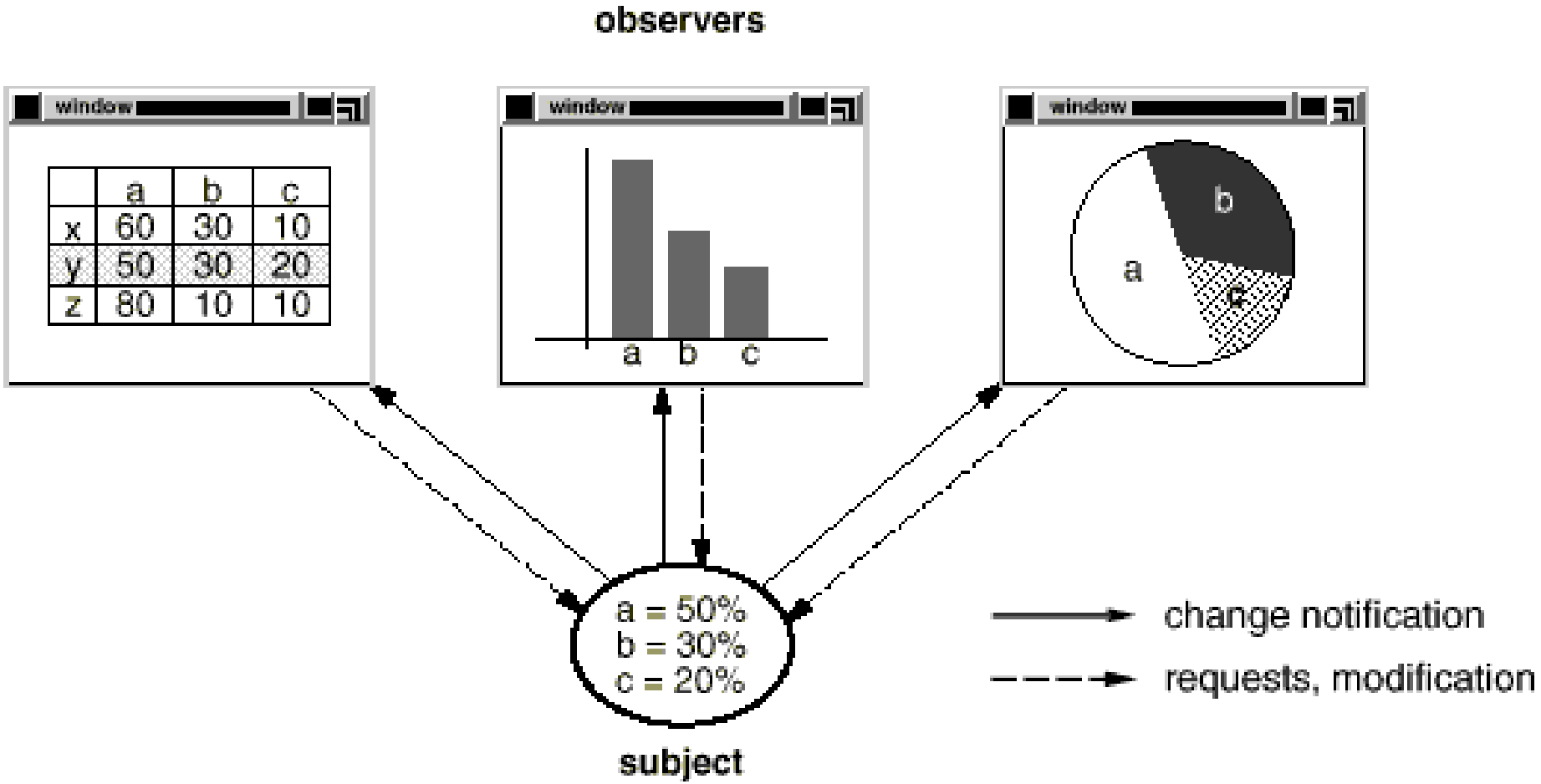
Design Patterns

- A design pattern is a general reusable solution to a common problem
- Patterns are known best practices
  - They allow reuse of knowledge from experts
- They do not describe a complete solution, since it is supposed to be reused in different applications

Elements of a Design Pattern

- Name
  - It is an identifies
- Problem Description
- Solution
  - It is a template of the solution and can be used in different applications
- Consequences
  - Results you get when you apply the pattern

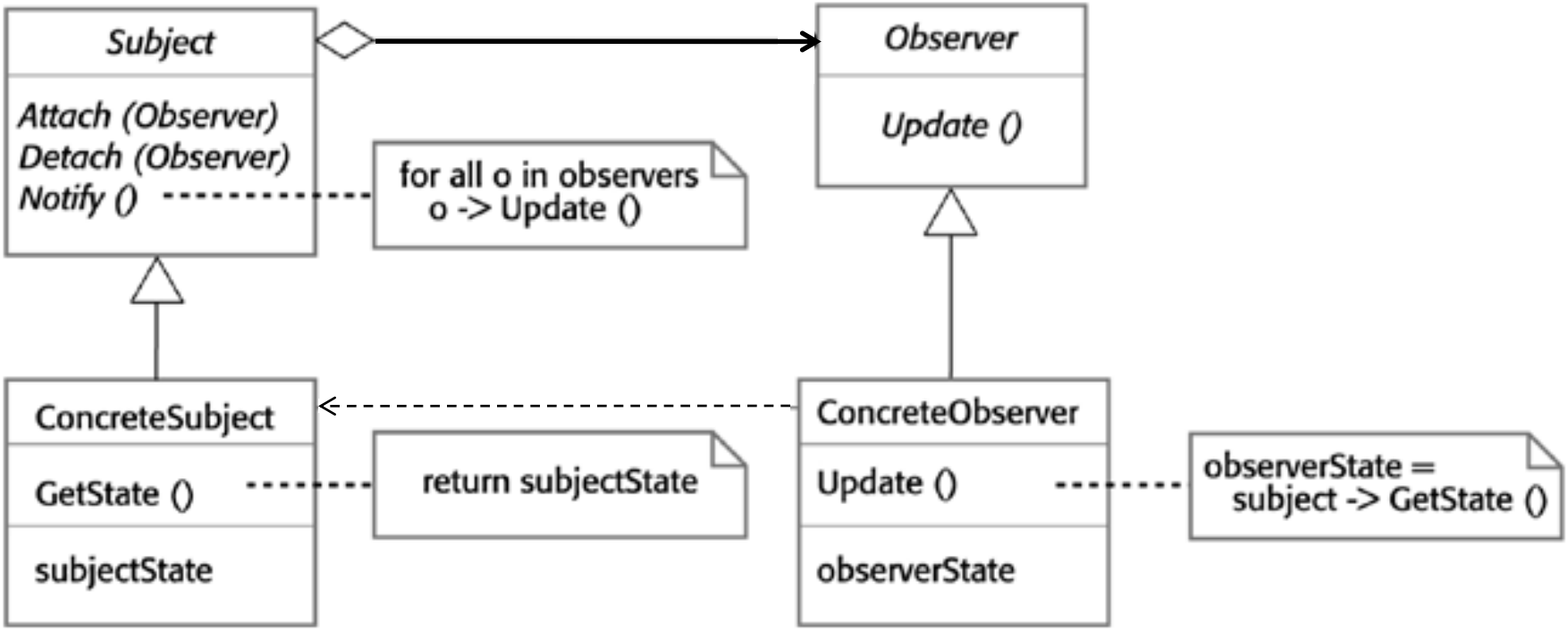
Example of Problem



The Observer Design Pattern

- Name
  - Observer
- Problem Description
  - It separates the subject of its representations
- Solution (next slide)
- Consequences
  - It optimizes updates from and notification to the viewers

Observer Solution



MDE and Code Generation

Motivation

- Reuse of code is usually hard to achieve
  - It involves loads of details which are language and platform dependents
- MDE proposes the development, reuse, maintenance, and evolution at design phase

MDE Solution

- Model-driven Engineering (MDE)
  - Also known as Model-driven Development (MDD) or Model-driven Architecture (MDA)
- It aims to raise the abstraction level
  - Reuse of models instead of code
- By means of code generators, the system code is automatically created

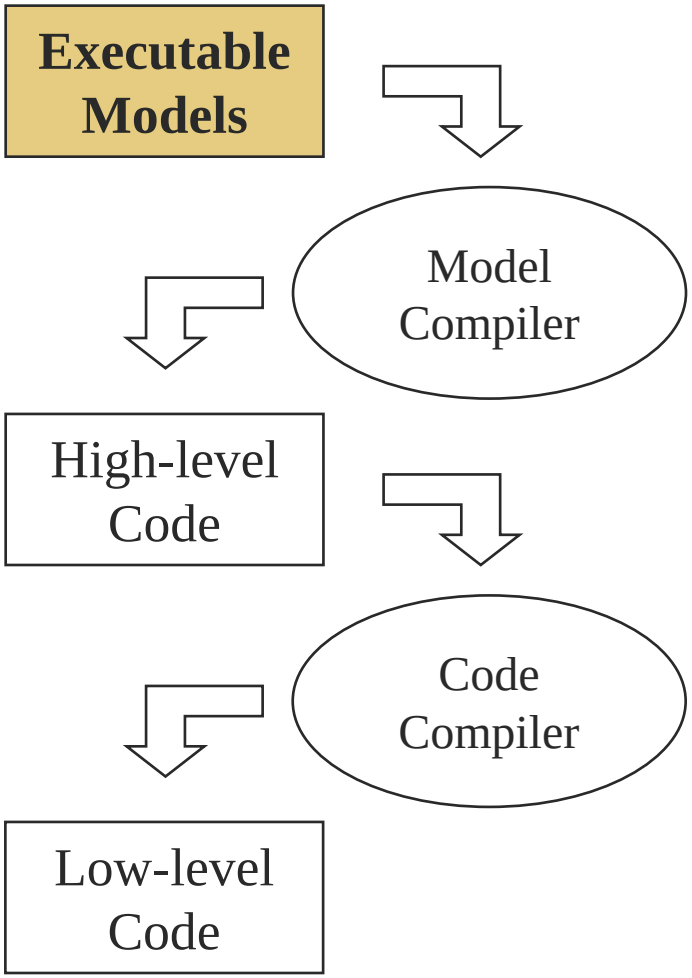
Why Reuse of Models?

- Models are expected to last longer
- Models facilitate communication among developers
  - They are sometimes used to communicate with costumers
- Models are usually developed in mature software process
  - With or without code generation

The MDE Approach

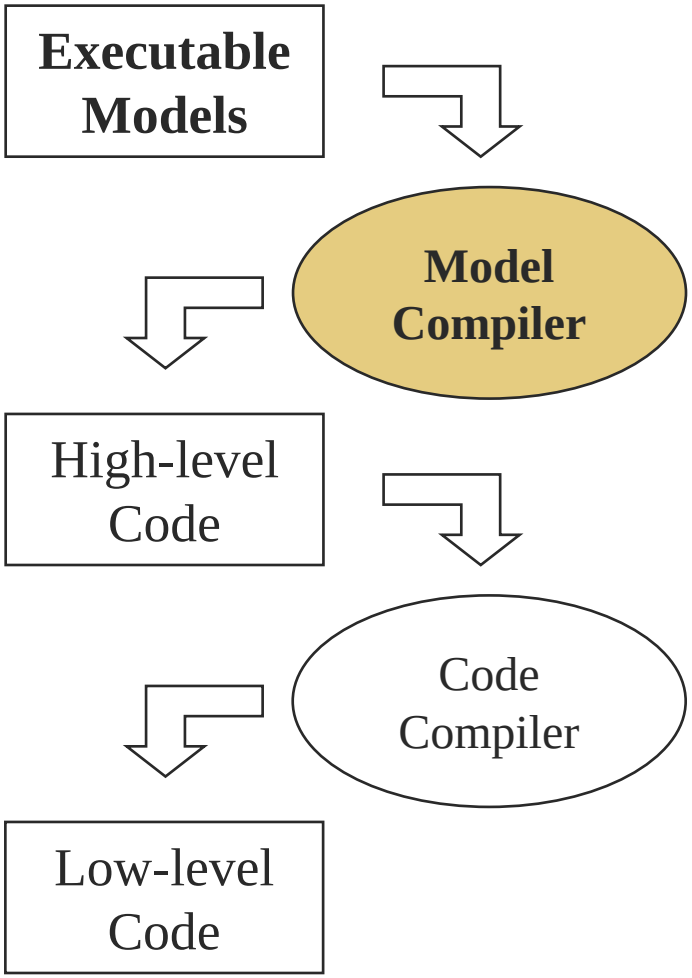
- Model reuse is still in maturing stage (not very frequent in industry)
  - It is more common in specific domains in research centers or academia
- MDE recommends three types of models
  - Computational Independent Model (CIM)
  - Platform-Independent Model (PIM)
  - Platform-Specific Model (PSM)

MDE Process



- Models are software independent
  - As, high-level code is hardware independent

MDE Process



- Models can be compiled to a high-level programming language
  - Models can be (partially) reused in different contexts

MDE Steps

1. Select some of existing models
2. Choose parts of the models you want
   - It might be necessary to adapt a model
   - It might be necessary to create new models
3. Integrate all partial models of a system
4. Choose an implementation technology
   - It might be necessary to describe the mapping from models to implementation
5. Generate the system code

Current Drawbacks

- MDE is still immature
  - For instance, it lacks support from tools and development environments
- Models are usually seen as superfluous
  - Code is the main asset
- Developers resist to MDE
  - They prefer programming than modelling
  - They are afraid of losing their jobs as programmers

Bibliography

- Ian Sommerville. **Software Engineering**, 10th Edition. Pearson Education, 2016.
  - Chap. 15 Software Reuse
  - Section 5.5 Model Driven Architecture