Robust Object Oriented Process (ROOP)

A model-first approach to learning and teaching object-orientation
What is ROOP?

- ROOP is a modelling process for building object oriented systems
- ROOP is still under development
  - The main focus has been on how to start off an object oriented system development process
- ROOP is the successor activity of Gerhard Skagestein’s article “Are Use Cases Necessarily the Best Start of an OO System Development Process?”
Why we introduce this methodology?

- To make the final model reusable by making the core architecture more robust.
- To avoid setting up a fixed system boundary too early in the system development process.
- To trigger organizational changes.
- To realize that Actor initiative can shift over time.
- The model may change dramatically, if this is not in our consideration early in the developing process.
How do we start?

- Start by identifying "true" objects.
- No consideration of what is computerized.
- Objects are on a relatively high abstraction level.
- No data mover objects are needed.
- Postponing the system boundary decision.

Diagram:

- Abstraction Layer
- Objects 1, 2, 3, 4, 5, 6
Tools in use

- Special designed patterns help us select “true” objects
- CRC cards and role play
- A new perspective on how objects are perceived
  - Computerized part
  - Real world part

The boundary is based on the distribution of responsibilities
Example – Coffee Machine I

- The boundary shifts with the responsibilities
- The boundary runs right through the objects, splitting them in a real world and a computerized world
- The communication between those parts is governed by a protocol, which may be documented by Use cases
Example – Coffee Machine II

- We select our system to fit the purpose
- The responsibility of choosing the right drink is shifted from the real to the computerized part of the consumer
- When we know the purpose of the system, then Use cases can be considered as a tool for describing the communication
Discussion

Will these higher abstraction objects contribute to a better understanding of the design model?

- Fewer objects → better overview
- Fewer objects → more responsibilities

Are split objects easier to find?

- What is included in the computerized world?
- What is included in the real world?