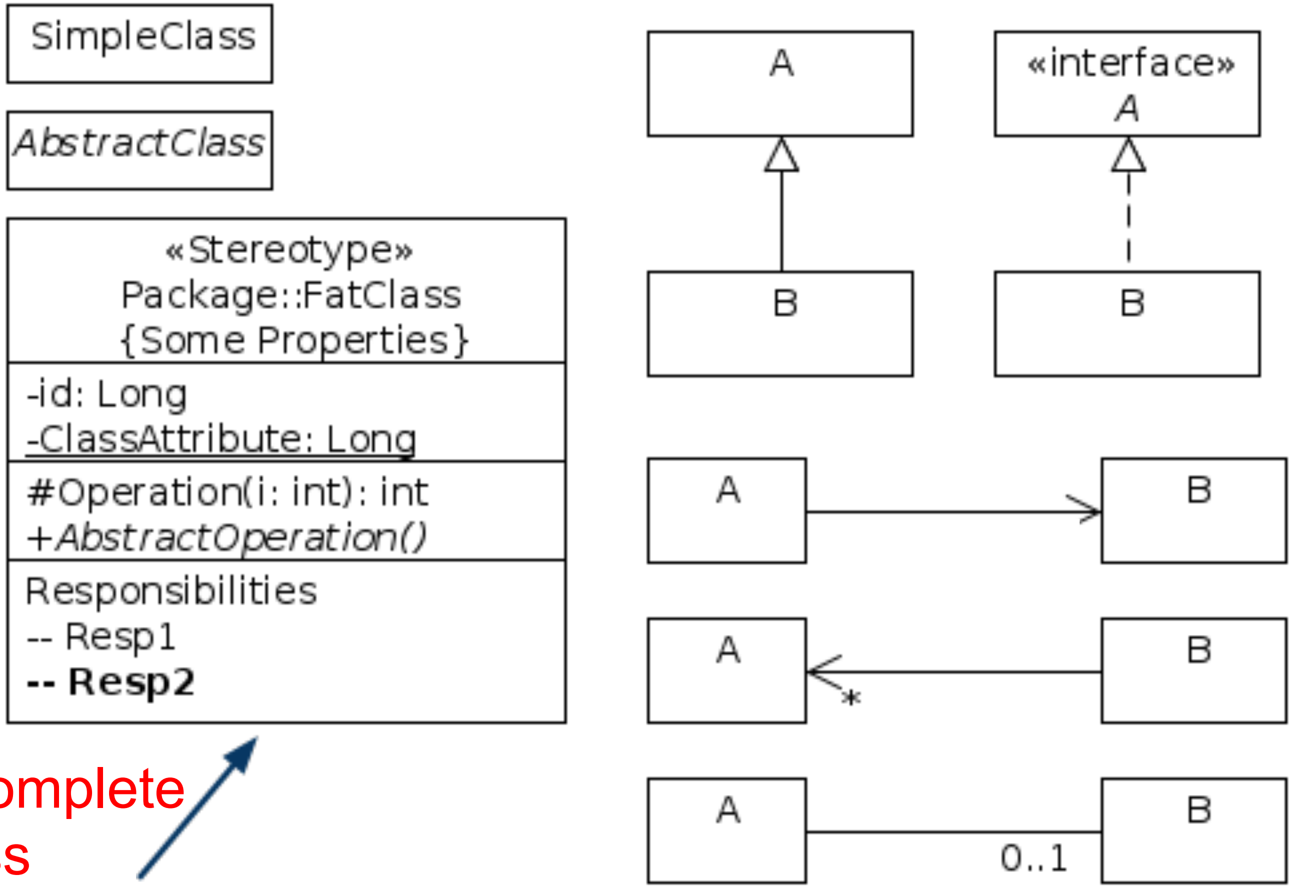
Analysis

Phase 2

Analysis

- During analysis we try to create an understandable model of the problem domain as a group of interacting objects
  - This is the **analysis model** (aka domain model)
- Have to find...
  - the objects (often called entities)
  - the relation between objects
  - the attributes of the objects
  - the behavior of the objects
  - the objects life cycles
- Model expressed in UML
  - class diagram (static view)
  - sequence diagrams (dynamic view)

UML for Static View of Analysis Model



SimpleClass

AbstractClass

«Stereotype»
Package::FatClass
{Some Properties}

-id: Long
-ClassAttribute: Long

#Operation(i: int): int
+*AbstractOperation()*

Responsibilities
-- Resp1
-- Resp2

A complete class (probably overkill)

Draw on white board/paper later
UMLet (Eclipse plugin)

Static Analysis Method

- Have the use cases
- Simple method
  - Underline nouns in use cases, will be classes
  - Underline verbs, will be methods
  - Find attributes/relationships from text (has, uses, is a, owns, knows, ...)
  - Include as much as possible. Easy to skip later, ...
- This is a **critical activity**
  - If model wrong, not complete, inconsistent, ...
  - ... trouble later!!!

Automate a mess and you get an...(guess)!

// The 21 laws of programming

Static Analysis Model for MoPro

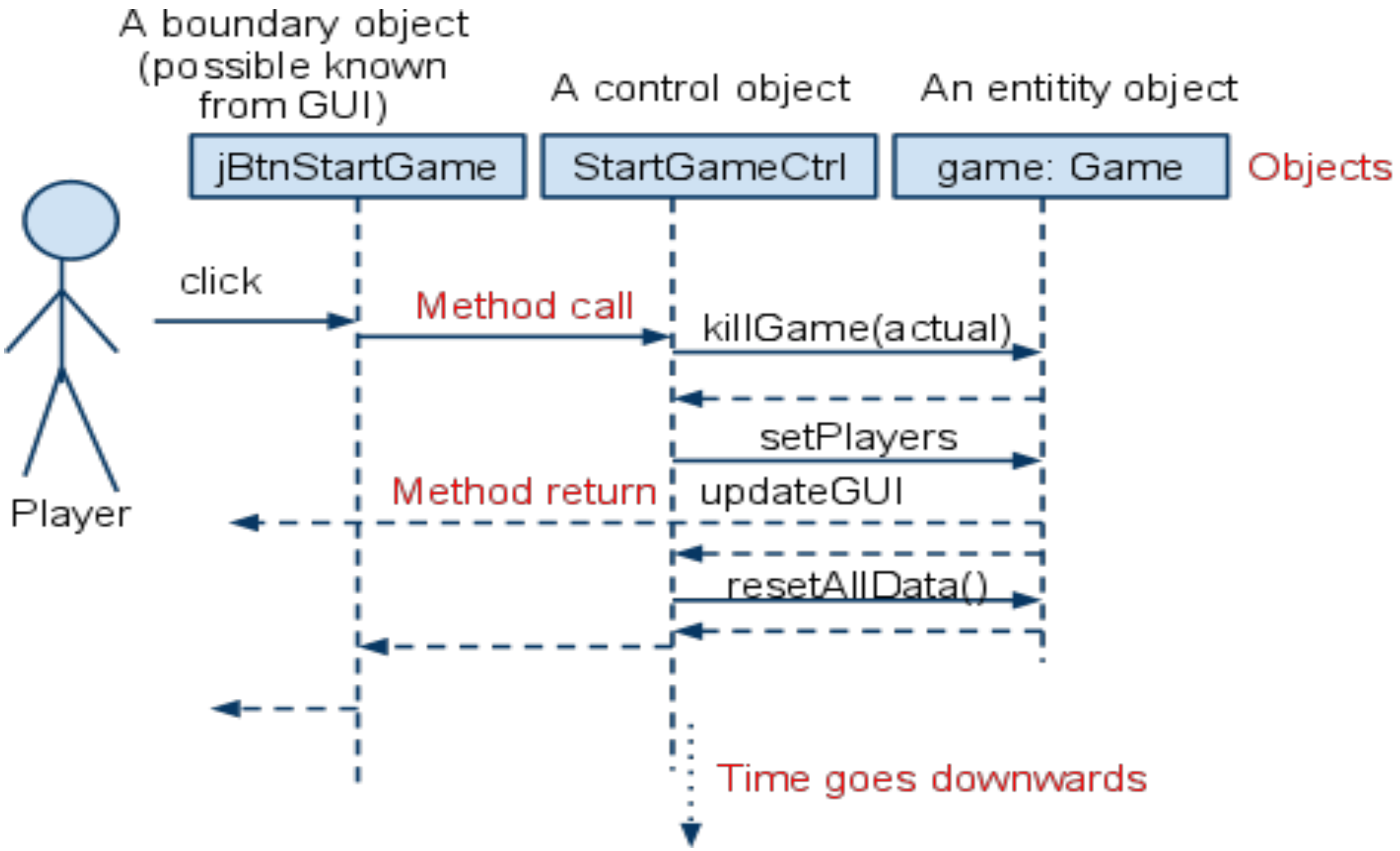
- We'll try to create...(use case text in monopoly-1.0.ep/doc, an Eclipse project, on course page in near future)
- ..then inspect RAD section 2.4.3 and appendix

Dynamic Analysis Model

- Have the entities and their connections from static model, how do they interact (runtime)?
- The dynamic model describes the interaction (simplified)
- During dynamic analysis possible new objects (not in static model) will show up
  - Boundary objects, represents the borders of the system (input, output). Example: GUI components, file handling,...
  - Control objects, manages the flow of the use case (the algorithm)
    - If introducing control objects we'll possible end up with an **anemic model**
    - Else we'll probably get a **fat model**
    - ... also possible with mixed approach (some control objects)

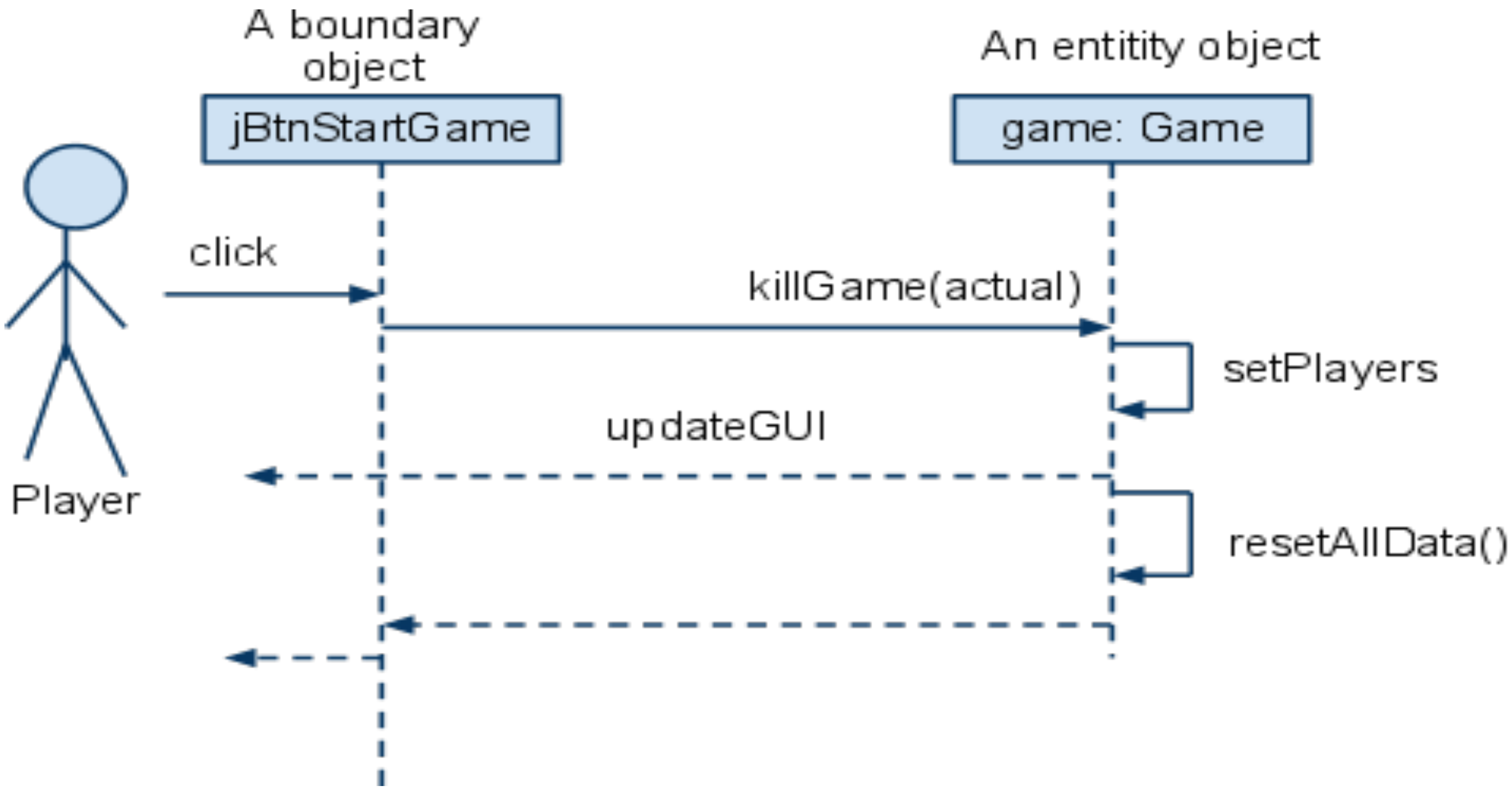
UML for Dynamic view of Analysis Model (1)

- Sequence diagram (**anemic** model)



UML for Dynamic view of Analysis Model (2)

- Sequence diagram (**fat** model)



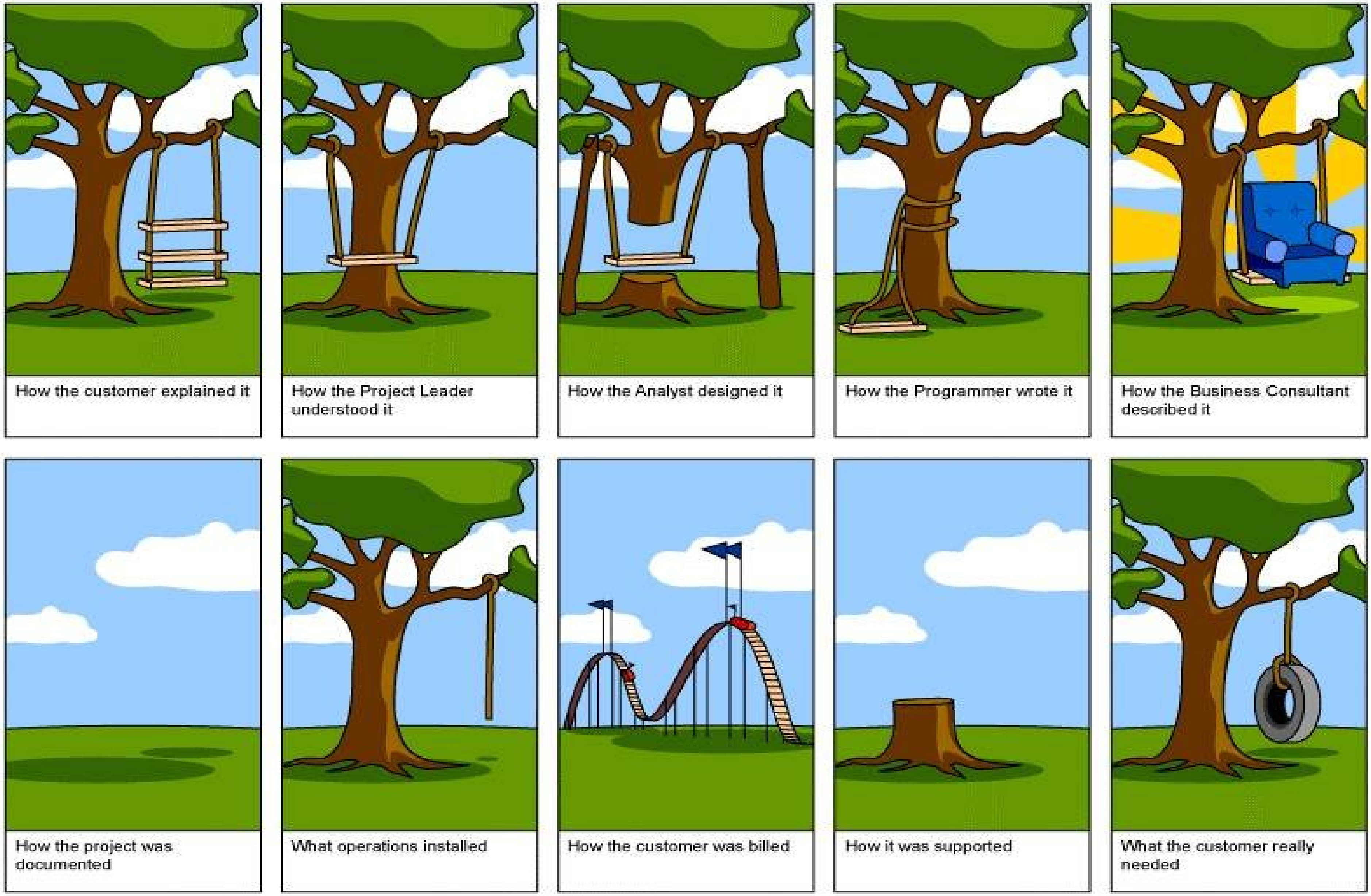
Dynamic Analysis Model for MoPro

- We'll try again
  - Using the "Move" use case from monopoly-1.0.ep/doc
  - Using the GUI mock-up
  - We use a control object

Documenting the Analysis

- Analysis is documented in RAD
- A final look at the RAD

Hmmm...


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

First Running Increment!!!

- First two phases finished, have preliminary RAD
- Must get something up and running
  - Implement use cases "Move" and "EndTurn"
- Have a detailed look at monopoly-1.0.ep
  - Demo run
  - Preliminary packages
  - Classes
  - Some code