Implementation

Phase 4

Implementation

- Goal is to get the first running version of design model
- As stated out: We do it incrementally "use case by use case" (version by version)
- For now we have the highest priority use case

Single Responibility Principle

- (Remainder) "Everything", package, class, method, attribute, ... should have one well defined responsibility (on it's abstraction level)
  - Possible the most important principle

- We have used this in domain and design model. Now we must **keep it that way**!
  - Easy to get lost when lots of details pop up
  - Switch between domain/design-model and code to keep control (also helps when stuck)
  - Possible change model if code reveals better ideas

Testing

- Unit testing: Testing a single class (object)
- Integration testing: Testing the interaction of many classes (objects), possible a full use case
  - Time consuming when number of objects increase
- Best fit for this course: **Unit testing (only)**
  - The test are an important part of the quality and documentation of the application

Unit Testing Techniques

- Test shouldn't need any human interaction (besides a start command)
  - Implies: No dependency from model to view
- Test nontrivial classes from design model
- Each class that needs testing has a companion "test class" performing the tests (a one-to-one match)
  - Model/design-class Player has test class TestPlayer (Java)
- Test class has "test methods". Each test method tests one (or a few) method(s) of "class under test" (CUT)
  - CUT has move(), test class has testMove(){ // call move() }
  - Sometimes hard to test single method. Test as few as possible
- Possible to debug tests. Very efficient!!!

Unit Testing Graphical Application

- Should be possible but maybe not to the same extent

What's happening in the model?

Unit Testing Technical Details

See Workshop JUnit and Monopoly 0.1 (on course page)

Example: From a cancelled Wordfeud-lab ...

Testing in MP 0.1

- Most classes trivial
- A small test of the move method (in Piece)
  - Must go in circle i.e. use modulus

Domain Driven Design (again)

- We focus on the model, the model is the solution
- We don't want the solution blurred by other code (in particular not tons of awful GUI code...)
  - Develop parallel but wait to connect it to the model
- So..
- ...to be able to "explore" the model we create a simple command line version of the program
  - Again: If needed, hard code, mock-up
  - For now a simple loop will be the only control (i.e. coordinating domain object to fulfill the functionality of a use case)

GUI and MVC, more later

Implementation of MP 0.1

- Technicalities
  - To be able to see the output we override toString() for most domain classes (common development practise)

- Demo run...!!! Now... MP 0.1
- Inspection model vs code...!

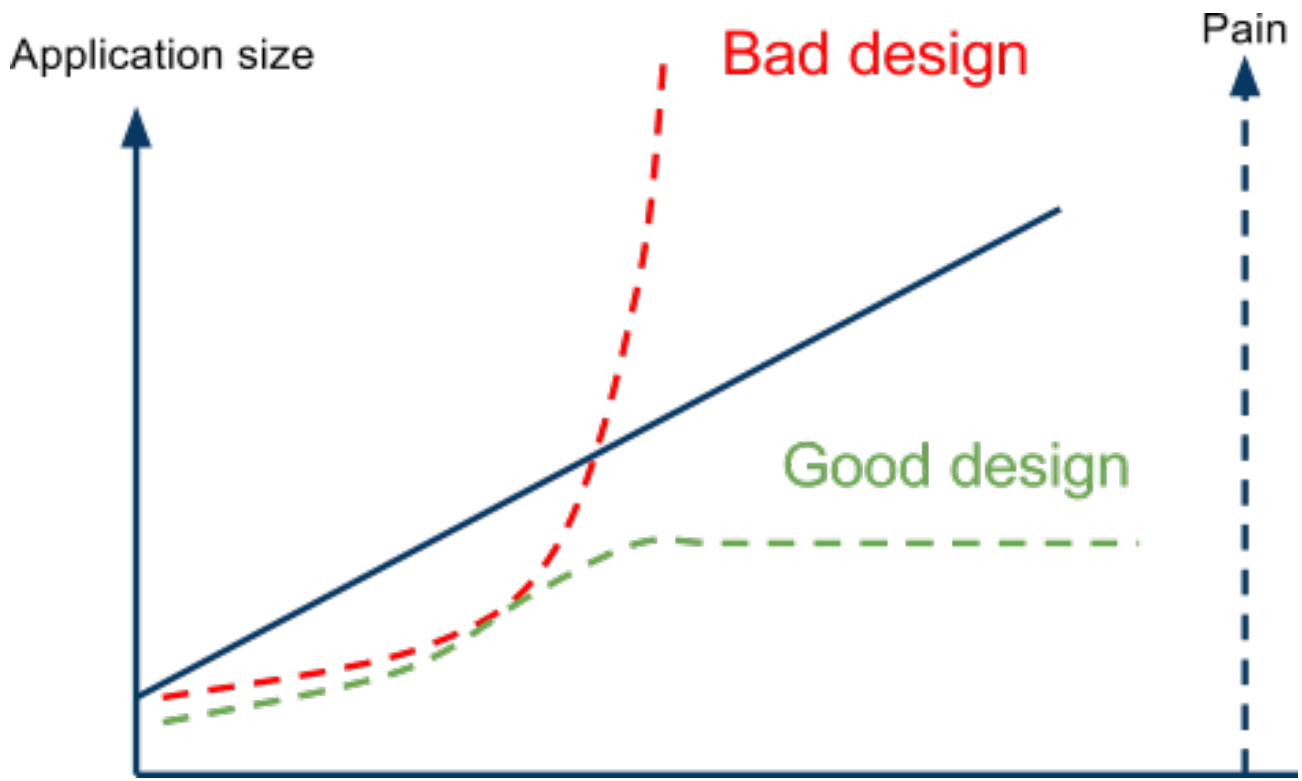
Different Applications

- As said. Not all applications are perfect OO (like MP)
- If application is very "graphical" (2-3D games)
  - Command line version probably not very useful
  - Make fist running version a very scaled down graphical version
    - Again: Goal is to explore the model
    - Much design to get a smooth interaction between model and "view"
- Others...(any one)?

Documentation at Implementation Level

- Code is the ultimate documentation ...
  - ... if it's understandable!
  - Put in a comment if you in **any way** think this could be hard to understand, high level, **what** is happening (not how)
  - All coding and comments in English
- Classes should have a class comment (at top)
  - What is this, responsibility of this, who uses it, @author
- Methods
  - If in need put a comment (better a clear informative name)
- Attributes
  - Why, for what (better a clear informative name)
- Silly/Bad comments are a pain or a risk (low quality)

The Impact of Design



On which curve are we..?

Hmm...


Our Disaster Recovery Plan
Goes Something Like This...
HELP!
HELP!

Summary

- Using the design model we have created a first running increment of the application
- We focus on the model, no GUI for now
  - Possible a special case for MP or alike
- We'll have some tests (start of a full test suite)

Next: The n-1 following iterations....