Object-Oriented Software Design **and Software Processes**

Hans Vangheluwe

Modelling, Simulation and Design Lab (MSDL)

School of Computer Science, McGill University, Montréal, Canada

Overview

1. Software Processes
2. The Process influences Productivity
3. The Rational Unified Process (RUP)
4. Extreme Programming (XP)

Software Processes

“The Software Engineering **process** is the total set of Software Engineering **activities** needed to transform requirements into software”.

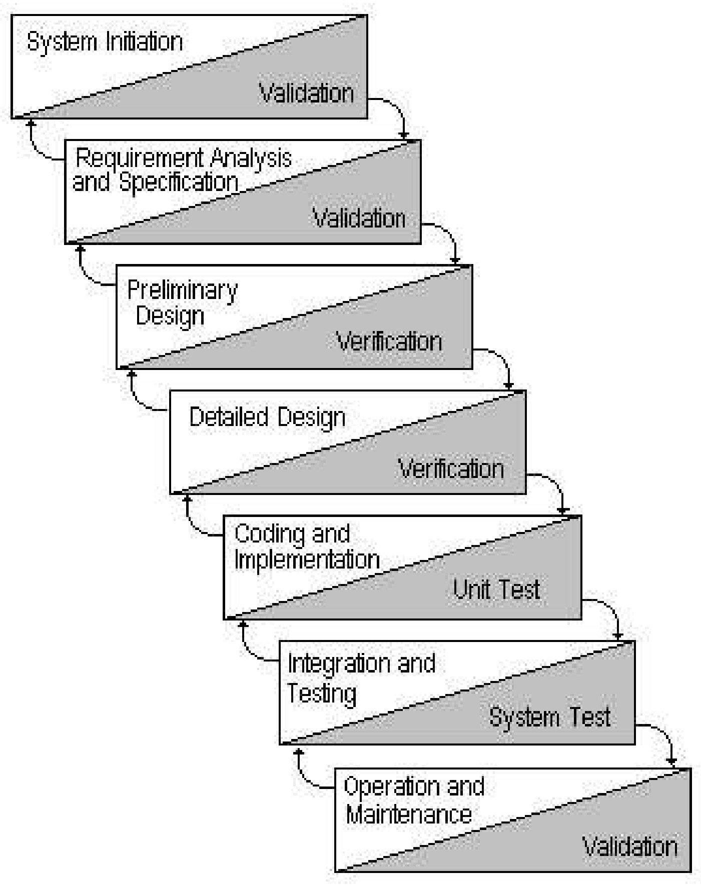
Watts S. Humphrey. Software Engineering Institute, CMU.

http://portal.acm.org/citation.cfm?id=75122

Some Software Processes:

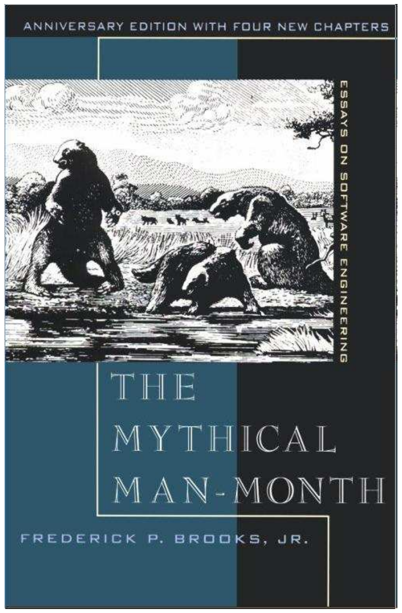
- Waterfall model
- Spiral model
- Throwaway/Evolutionary prototyping model
- Incremental/iterative development
- Automated software synthesis
- . . .

The Waterfall Model (W. Royce. 1970)



http://www.informatik.uni-bremen.de/gdpa/def/def_w/WATERFALL.htm

The Process influences Productivity

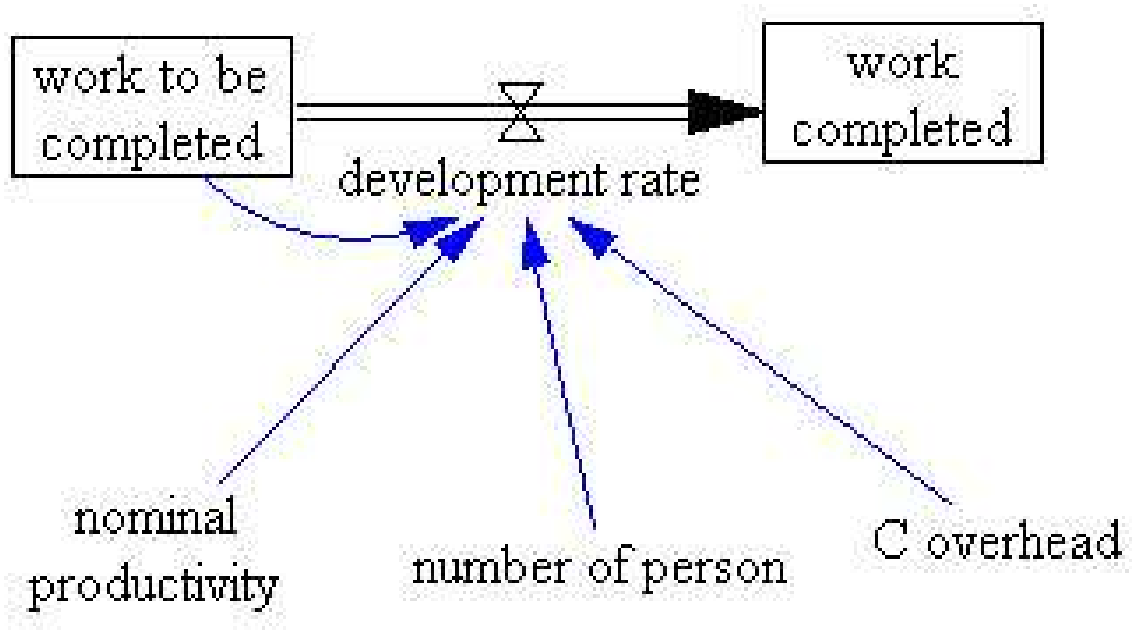

ANNIVERSARY EDITION WITH FOUR NEW CHAPTERS
ESSAYS ON SOFTWARE ENGINEERING
THE MYTHICAL MAN-MONTH
FREDERICK P. BROOKS, JR.

“Adding manpower to a late software project makes it later”.

Fred Brooks. The Mythical Man-Month.

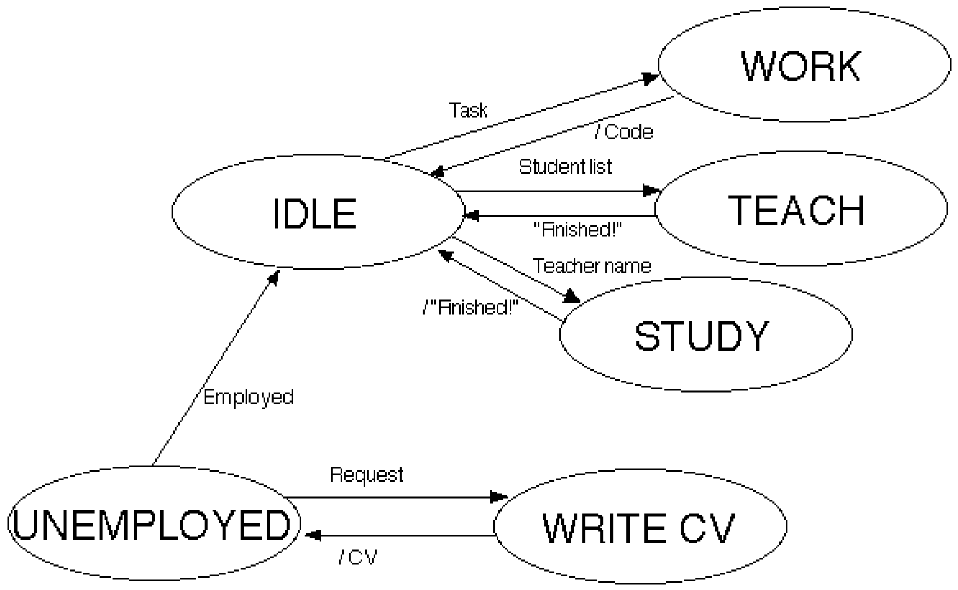
http://www.erdb.com/featu re/f eatu re.0 001 .htm l

Why Brooks' Law ? Team Size.



development rate =
 nominal_productivity*(1-C_overhead *N^2) *N

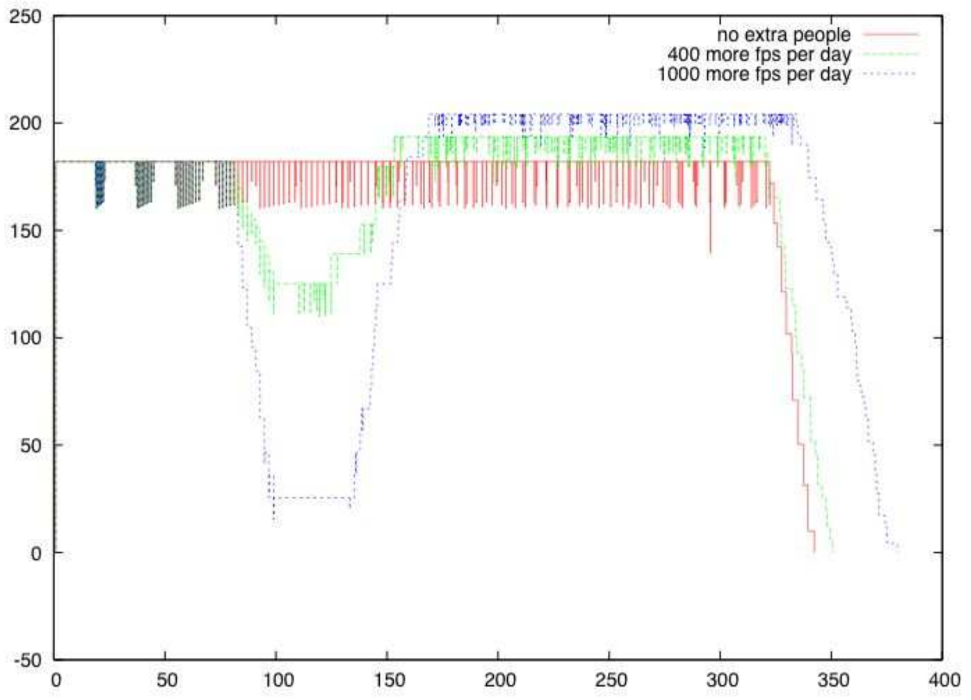
Why Brooks' Law ? Programmer Behaviour.



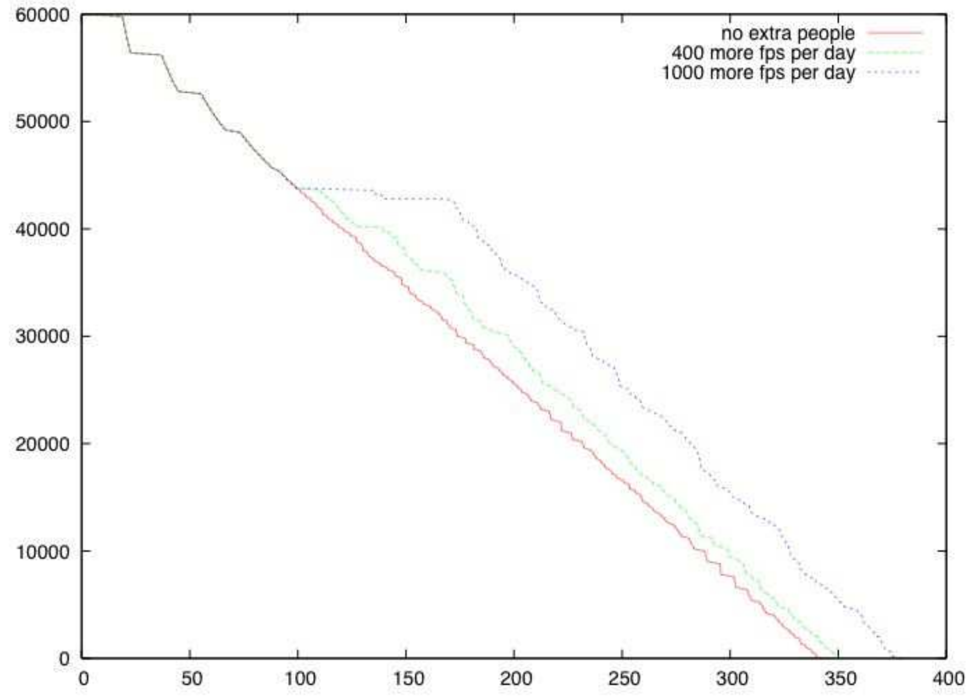
Eystein Fredrik Esbensen's COMP 522 project.

http://www.stud.ntnu.no/~ eysteinf/final.html

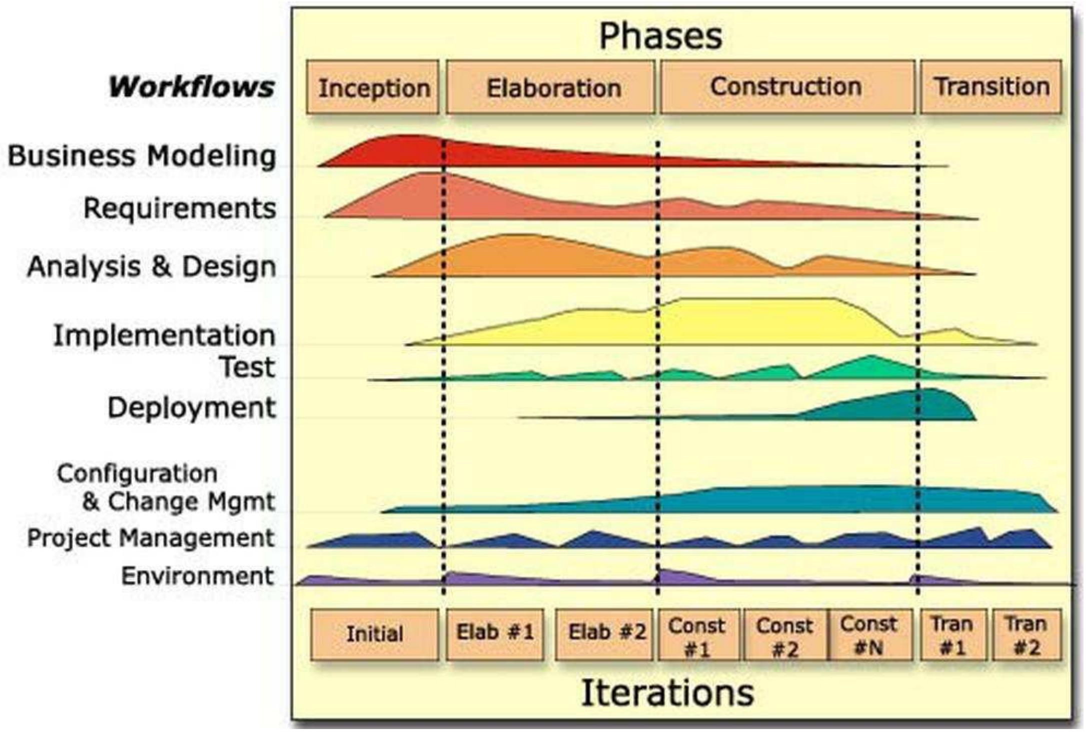
Why Brooks' Law ? Productivity.



Why Brooks' Law ? Remaining work.



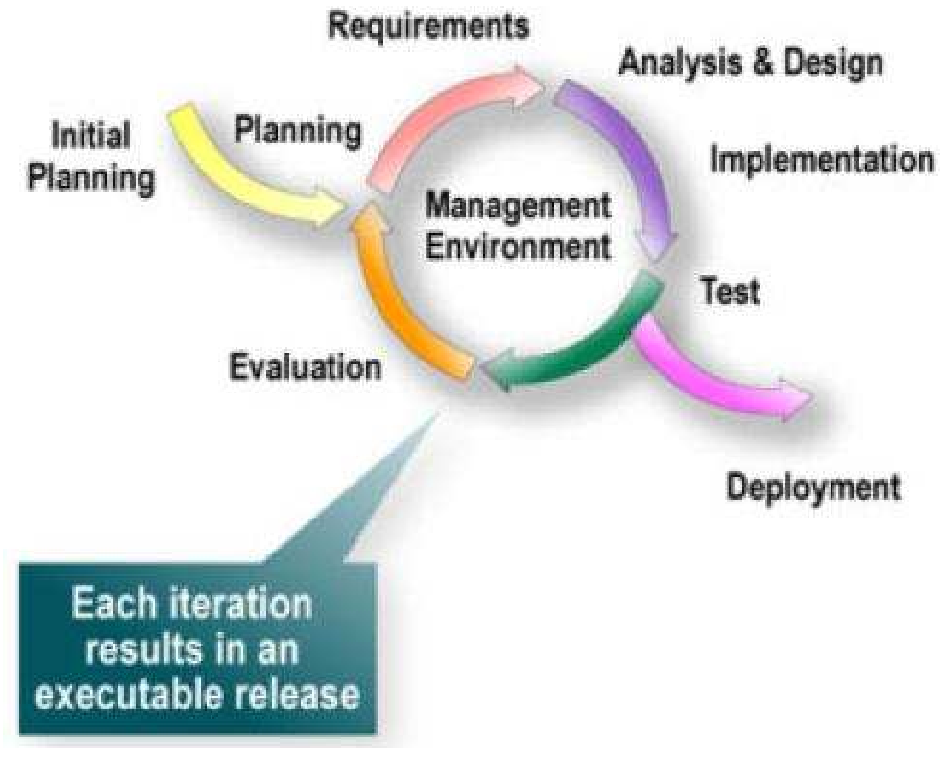
The Rational Unified Process (RUP): Activity Workload as Function of Time



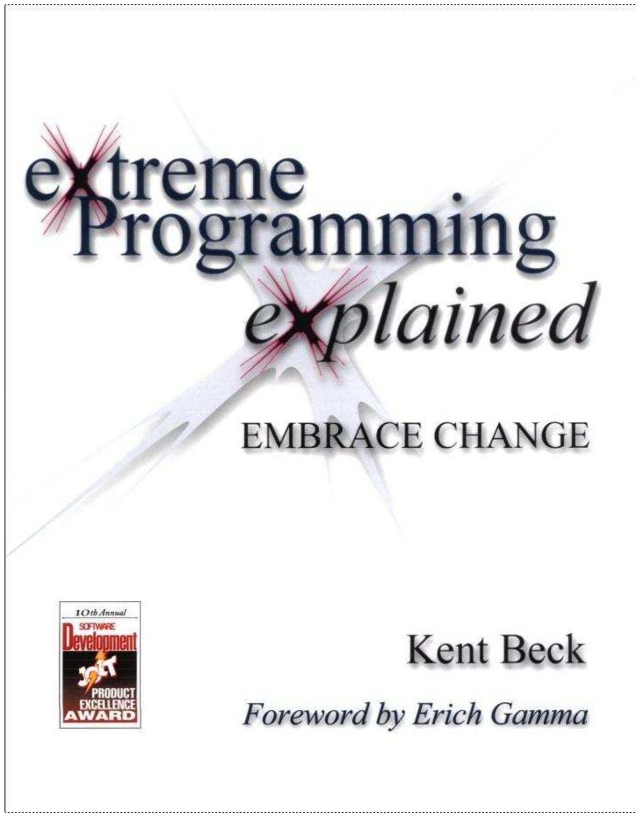
The Rational Unified Process (RUP): Observations

1. Waterfall-like **sequence** of
   Requirements, Design, Implementation, Testing.
2. Not pure waterfall:
   - **Iteration**
   - Overlap (**concurrency**) between activities
3. Testing:
   - **Regression** (test not only newly developed, but also previously developed code)
   - Testing starts **before** design and coding (Extreme Programming)

The Rational Unified Process (RUP)



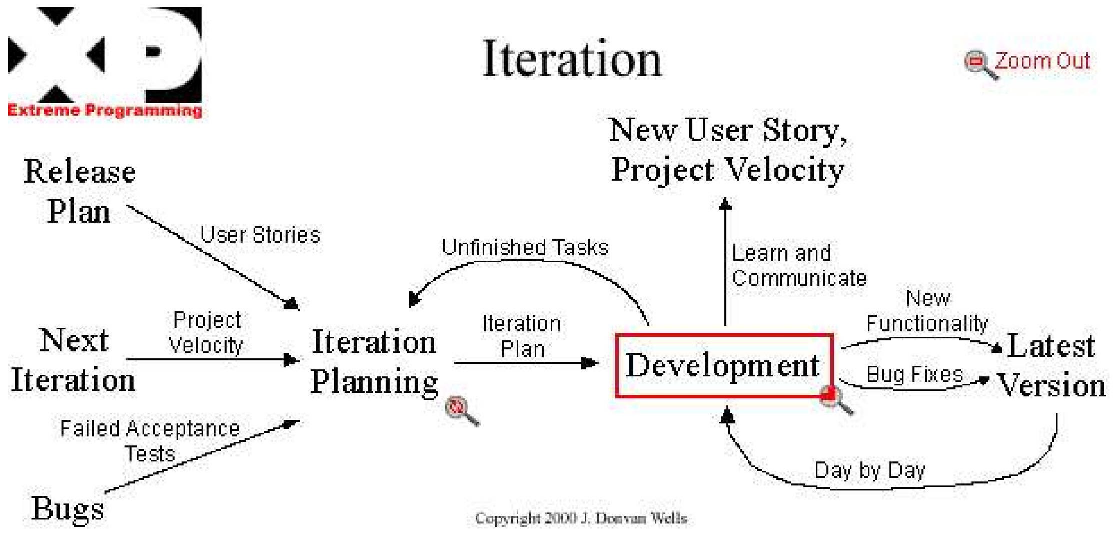
Extreme Programming (XP)



www.extremeprogramming.org

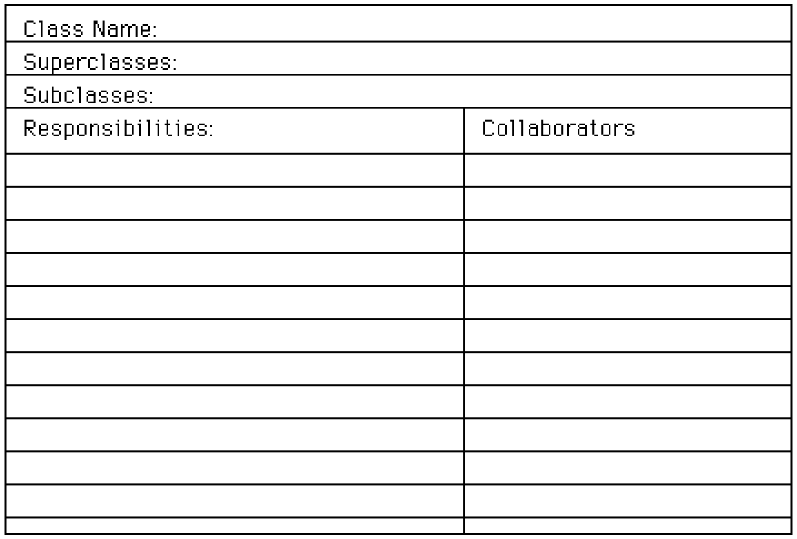
Extreme Programming (XP) highlights

- **User Stories** are written by the customers as things that the system needs to do for them. They drive the creation of acceptance **tests**.
- The project is divided into **Iterations**.



Extreme Programming (XP) highlights

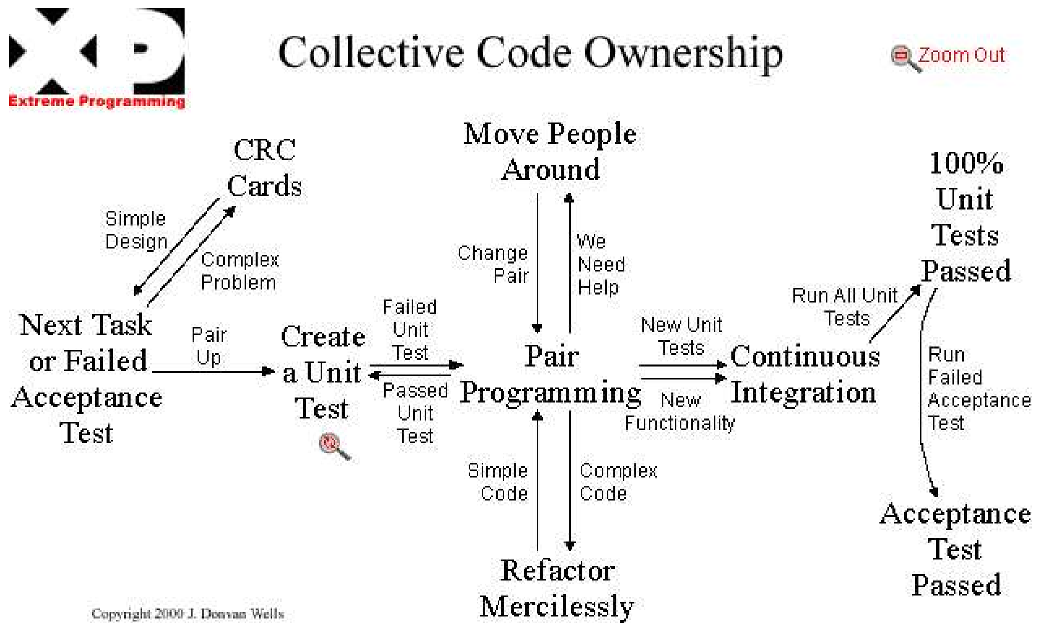
Use Class, Responsibilities, and Collaboration **(CRC) Cards** to **design** the system.



Class Name:	
Superclasses:	
Subclasses:	
Responsibilities:	Collaborators

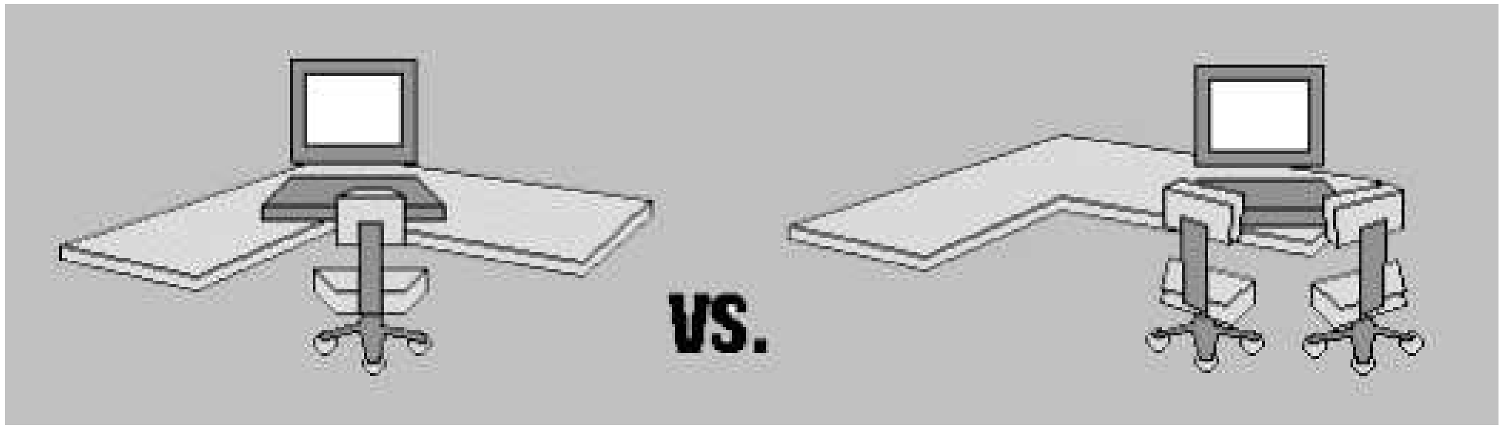
Extreme Programming (XP) highlights

- Code the Unit Test **first**.
- **All code** must have Unit Tests; All code must pass **all** unit tests before it can be released.



- **Refactor** whenever and wherever possible.

Extreme Programming (XP) highlights



Pair Programming