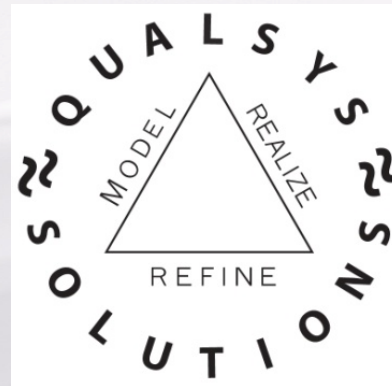


# How Can a Tester Cope With the Fast Paced Iterative/Incremental Process?

by  
Timothy D. Korson



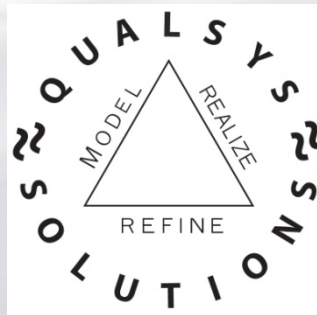
Version 7.0814

# Restricted Use

This copyrighted material is provided to attendees of QualSys Solutions courses under a restricted licensing agreement exclusively for the personal use of the attendees. QualSys Solutions retains ownership rights to the material contained in these course notes.

Please respect these rights.

Any presentation or reuse of any part of this material in any form must be approved in writing by [tim@qualsys.org](mailto:tim@qualsys.org)



Copyright © 2009 Qualsys Solutions. All rights reserved.

# Outline

- What is iterative/incremental software engineering?
- How can a tester cope?

**Analysis**

**High-level design**

**Detailed design**

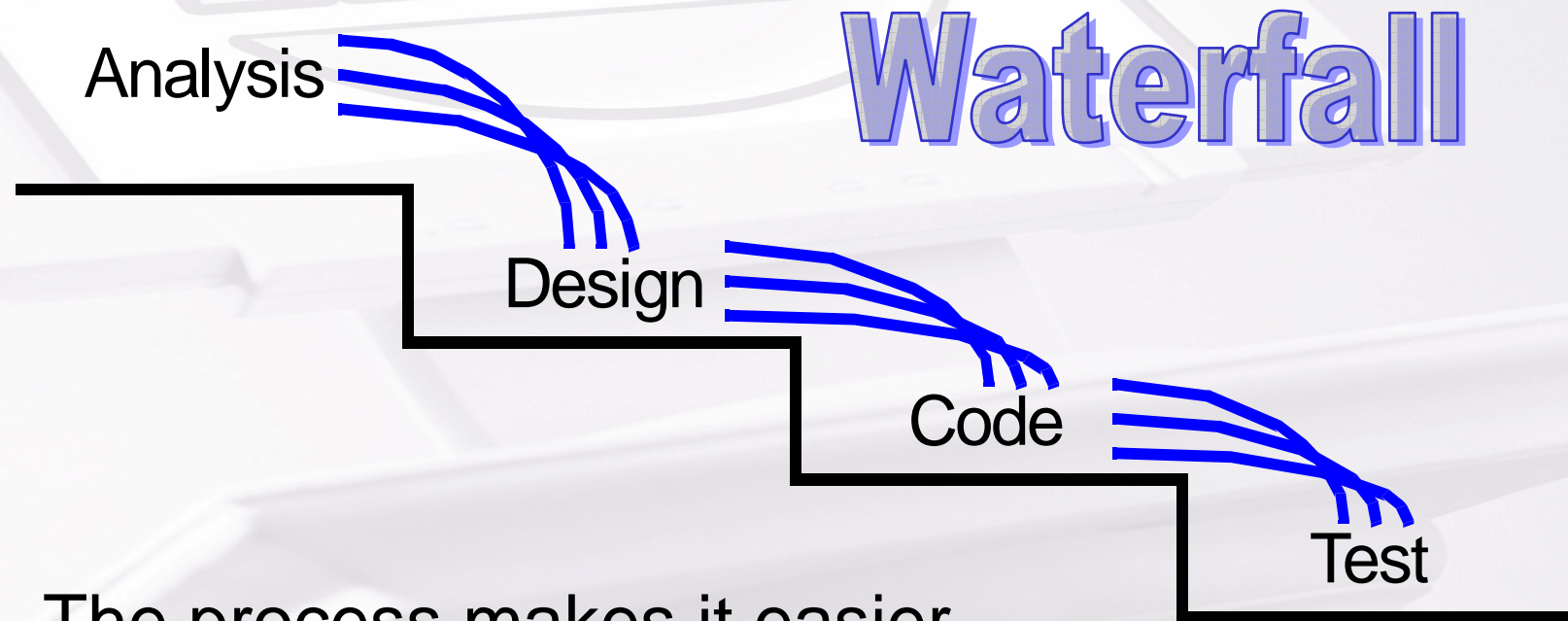
**Implementation**

**Testing**

**Production**

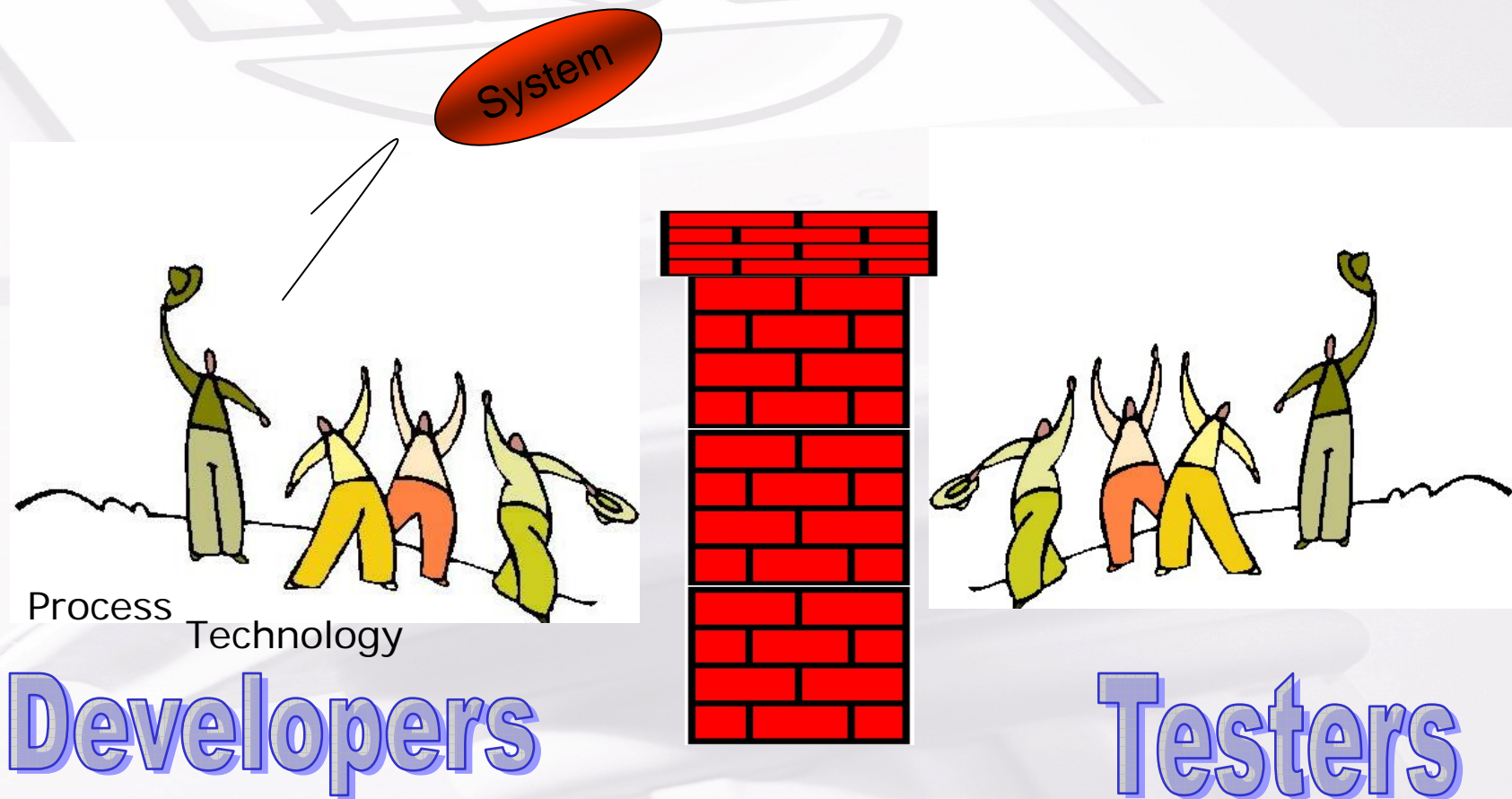


# Traditional Software Engineering



The process makes it easier  
to specialize and handoff

# Over the Wall

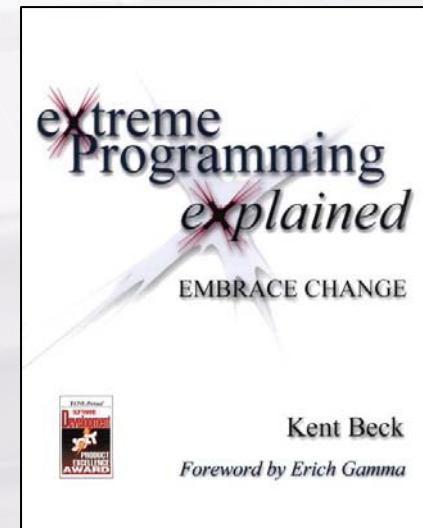
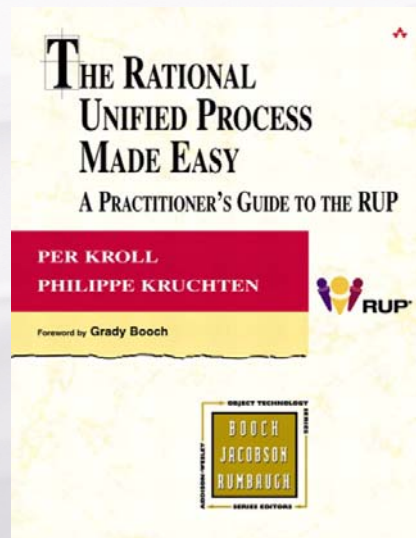
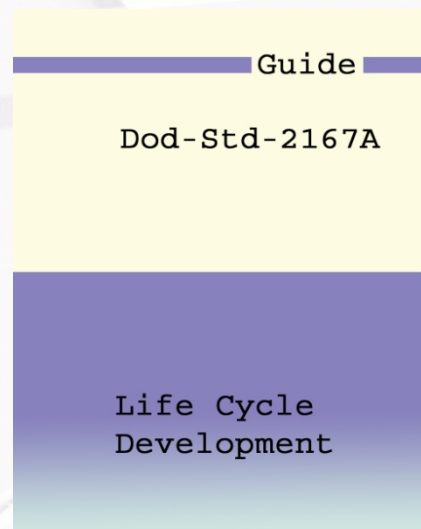


# Process Spectrum

Waterfall

RUP

XP



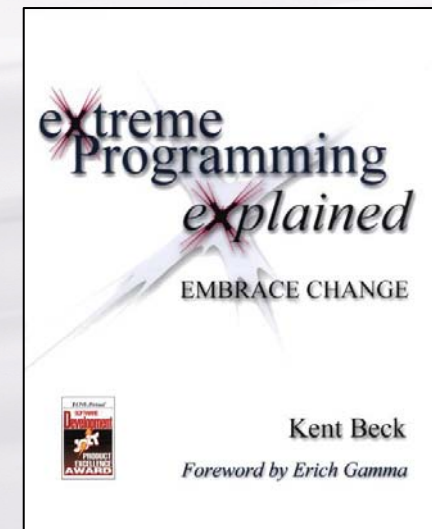
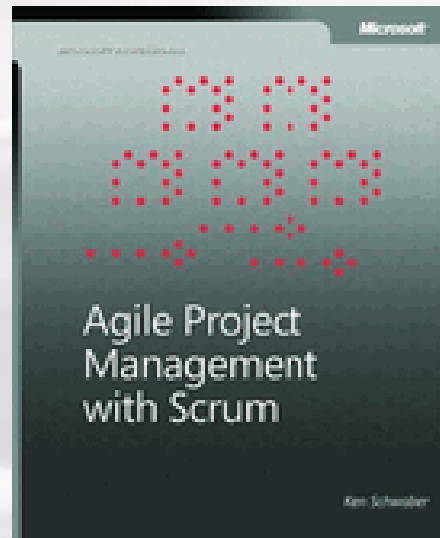
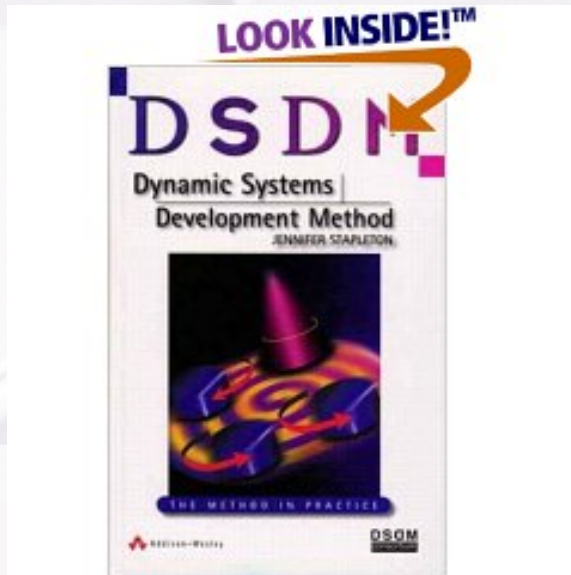


# Process Spectrum

Waterfall

RUP

Agile



# Incremental Model

- An increment is some subset of the system that is “completely” coded and **tested**.

**Analysis**

**High-level design**

**Detailed design**

**Implementation**

**Testing**

**Production**



# Fixed Delivery Dates

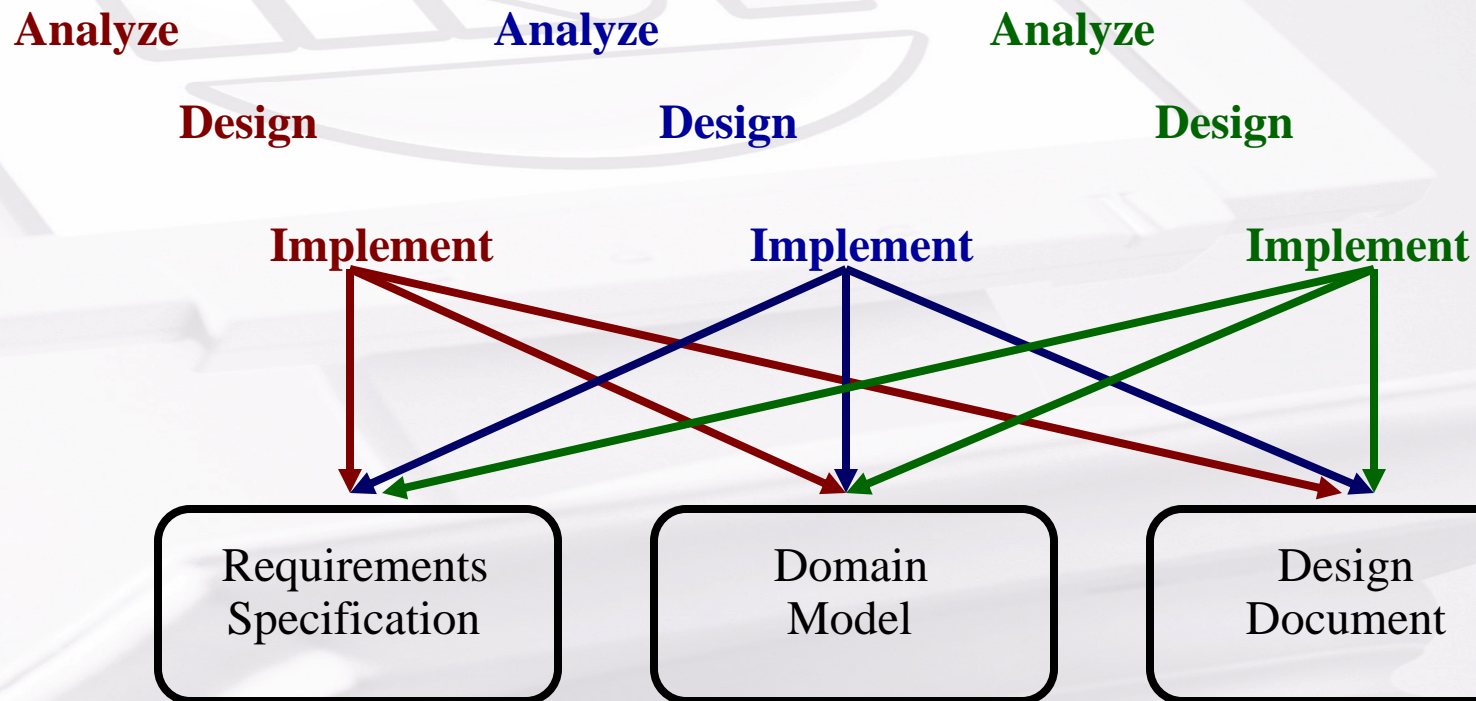
- Delivery dates are fixed
- Scope is negotiated as necessary

**Predictability**

**Trust**

**Feedback**

# It's Incremental Everything



*Other important documents, such as the user manual and design documents, are also developed, and **verified** incrementally.*

# Example

- Business logic to post a simple transaction
- GUI to enter transaction data
- DBMS to persist transaction
- Log on and user authentication code
- Additional business logic to all layers
- ...

Multi-currency, fund accounting system



# Iteration Length

- Rational Unified Process (RUP)
  - Up to 6 months
- Scrum
  - One Month
- XP
  - Could be as short as one week

# How Can I Cope?

- Process
- Organization
- Collaboration
- Automation
- Participate in Planning
- Test first development



# How Can I Cope?

- **Process**
- Organization
- Collaboration
- Automation
- Participate in Planning
- Test first development

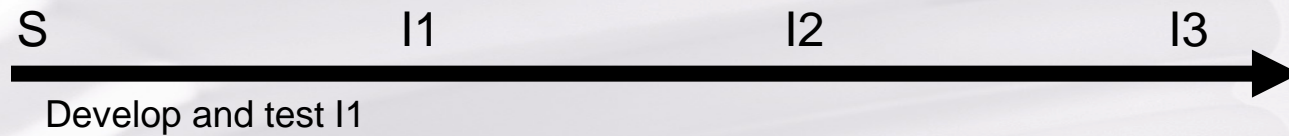




# Testing Process Should Match the Development Process

- **Basic development process is incremental**
  - Iterate within increments, with prototype support as necessary. *Often this will involve reworking one piece of the system several times before an increment is finished. Previous increments are refactored as necessary.*
- **Testing process should be incremental**
  - Early increments may be gray boxes
  - The testing perspective is black box, but an early increment may have lots of incomplete functionality and may not be executable as a “stand alone” system.

# Testing Increments



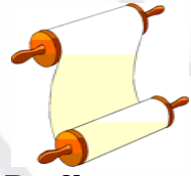
# How Can I Cope?

- Process
- **Organization**
- Collaboration
- Automation
- Participate in Planning
- Test first development

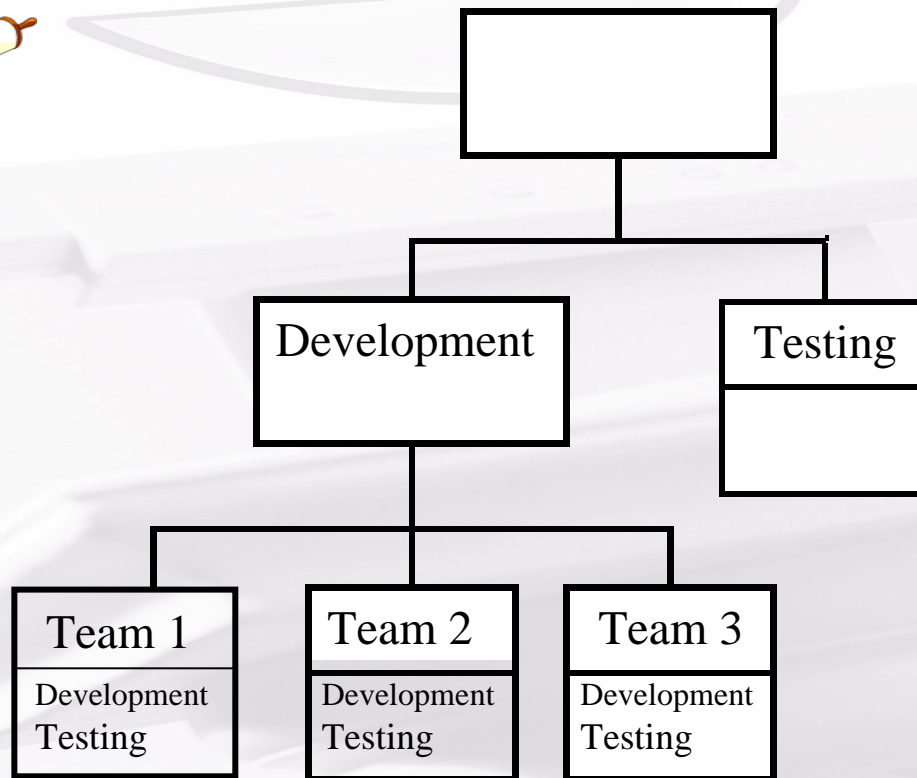




# Recommended Organizational Approach



**Define**



# How Can I Cope?

- Process
- Organization
- **Collaboration**
- Automation
- Participate in Planning
- Test first development

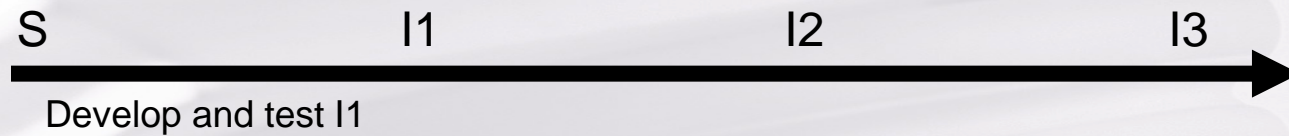


# No One Is Done Until Everyone Is Done





# Testing Increments

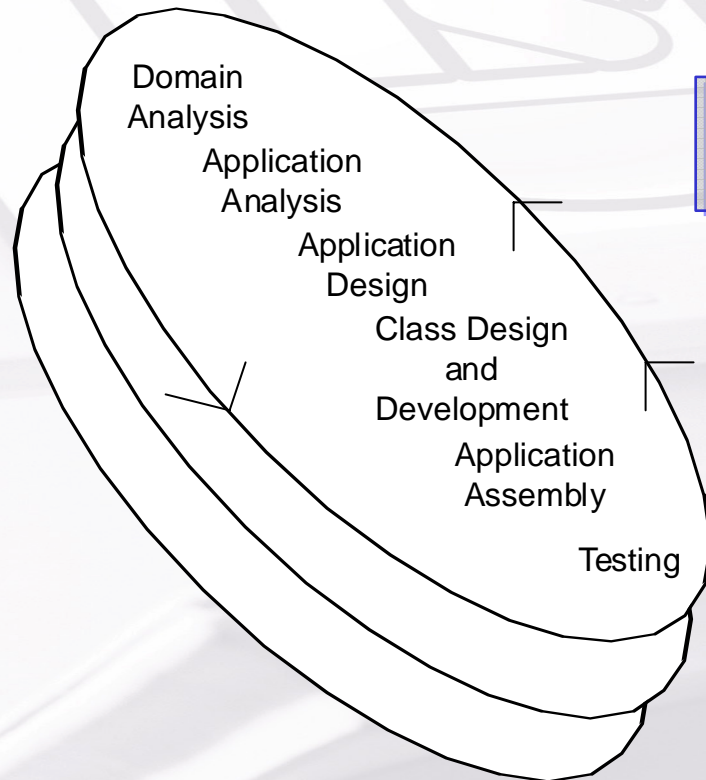


# Pair Programming



Guarantees 100% code inspections

# Modern Software Engineering



Iterative

Incremental

The process encourages integrated teams





# Requirements

- RUP
  - Use cases have lots of detail
  - Business analysts document the requirements
  - Testers work from written requirements expressed as use cases
- Agile
  - Stories lack detail
  - Stories are reminders to have a conversation
  - The details from the conversations are recorded directly in test cases
  - The test cases become the detailed specifications
  - Testers become business analysts

# How Can I Cope?

- Process
- Organization
- Collaboration
- **Automation**
- Participate in Planning
- Test first development



# Almost Continuous Integration

- Almost continuous integration avoids or detects compatibility problems early. Integration is a "pay me now or pay me more later" kind of activity.
- Developers should be integrating and releasing code into the code repository every few hours, when ever possible. In any case never hold onto changes for more than a day.
- Each development pair is responsible for integrating their own code when ever a reasonable break presents itself.

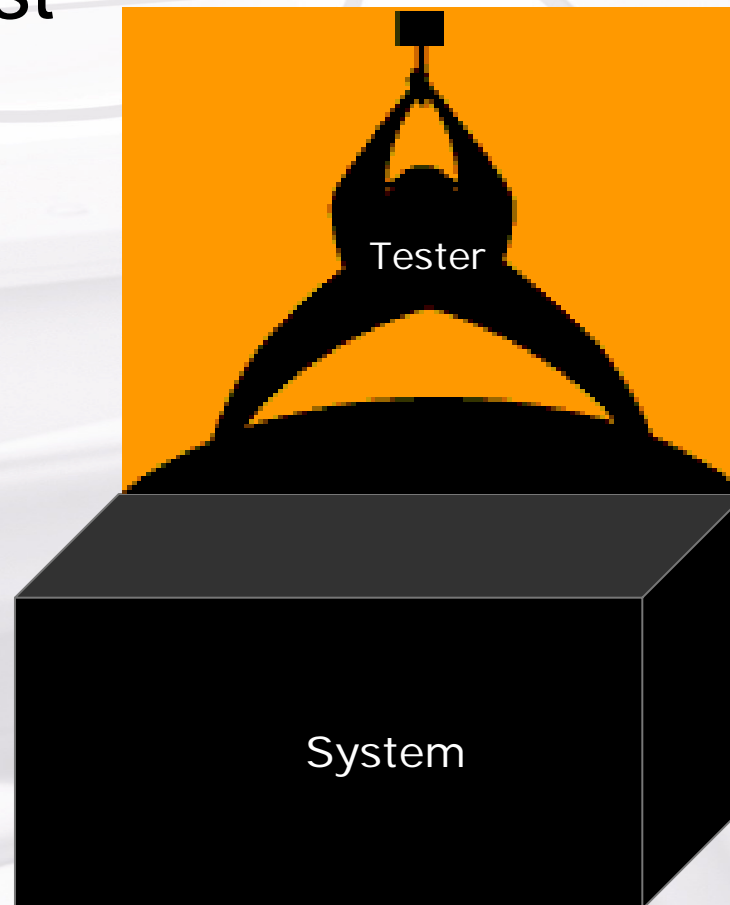


Smoke Test



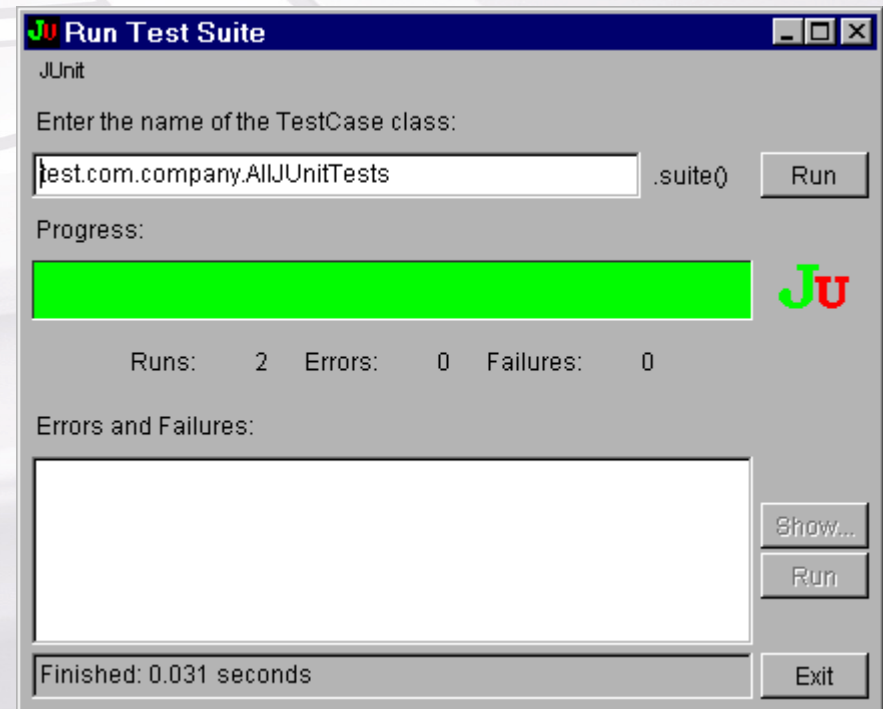
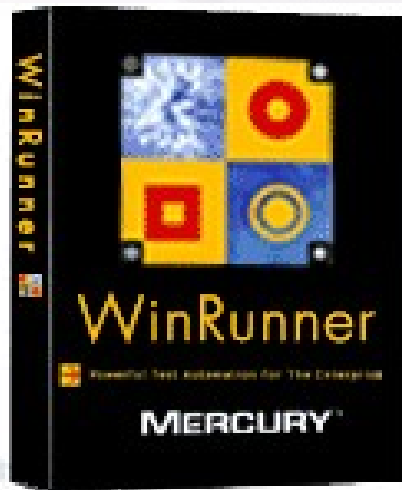
# Regression Suites

- Build smoke test
- Daily test
- Weekly test
- Increment test



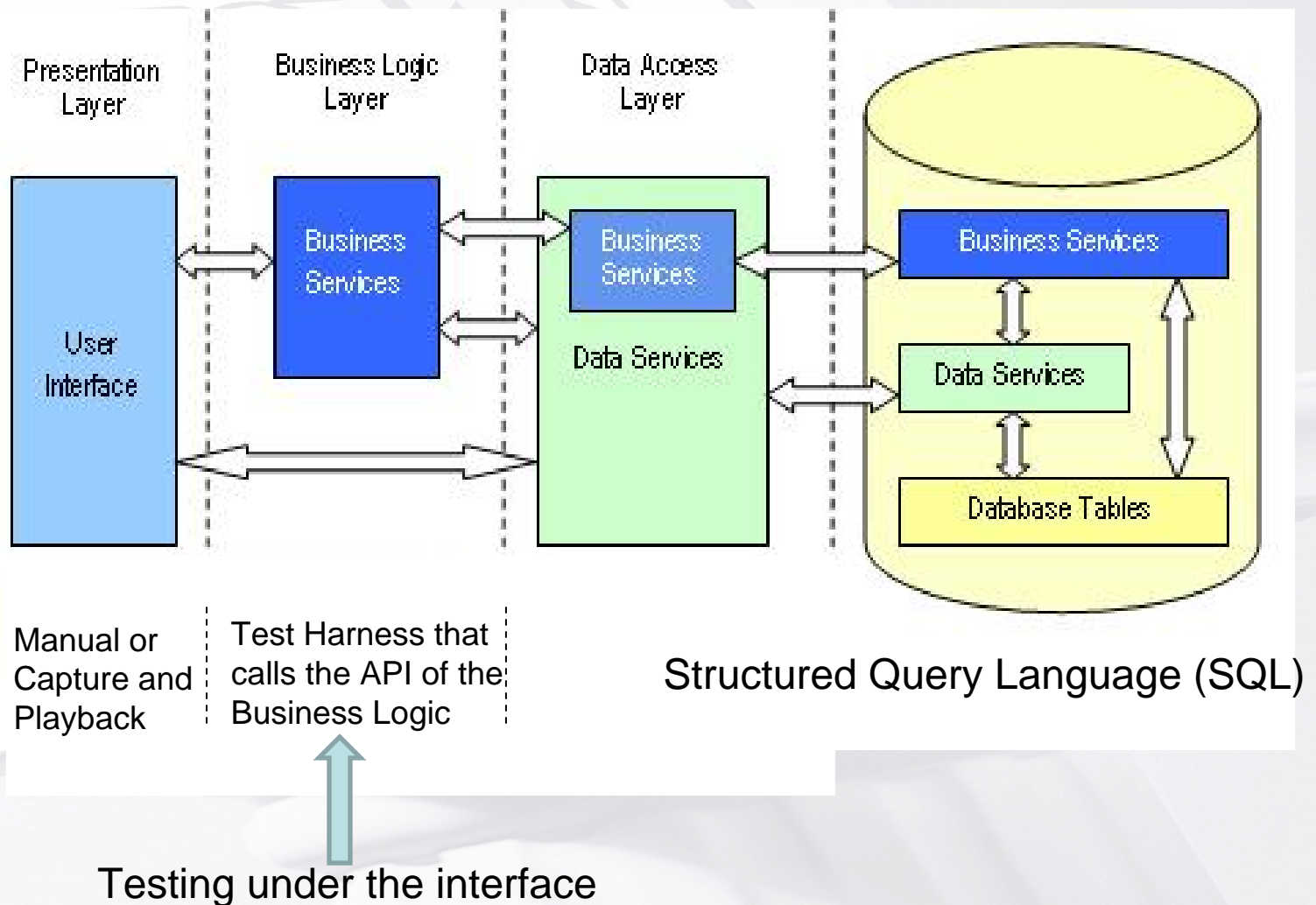
# Automation Tools

- Record and Playback
- Testing “Under the Interface”



## JUnit

# How Does One Automate?





# How Can I Cope?

- Process
- Organization
- Collaboration
- Automation
- **Participate in Planning**
- Test first development



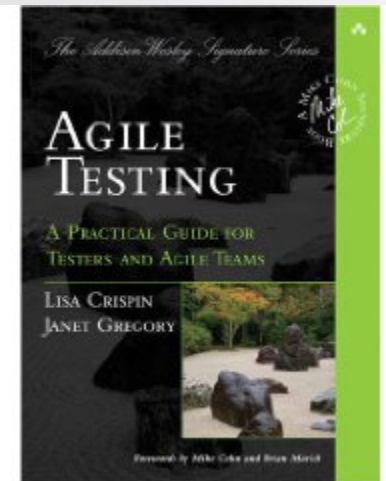
# Including Time for Acceptance Test in Planning the Estimates





# Up-Front Tester activities

- Identifying and making explicit hidden assumptions
- Defining acceptance tests for each story
- Including time for acceptance test in planning the estimates
- Enabling accurate estimates of time and velocity





# How Can I Cope?

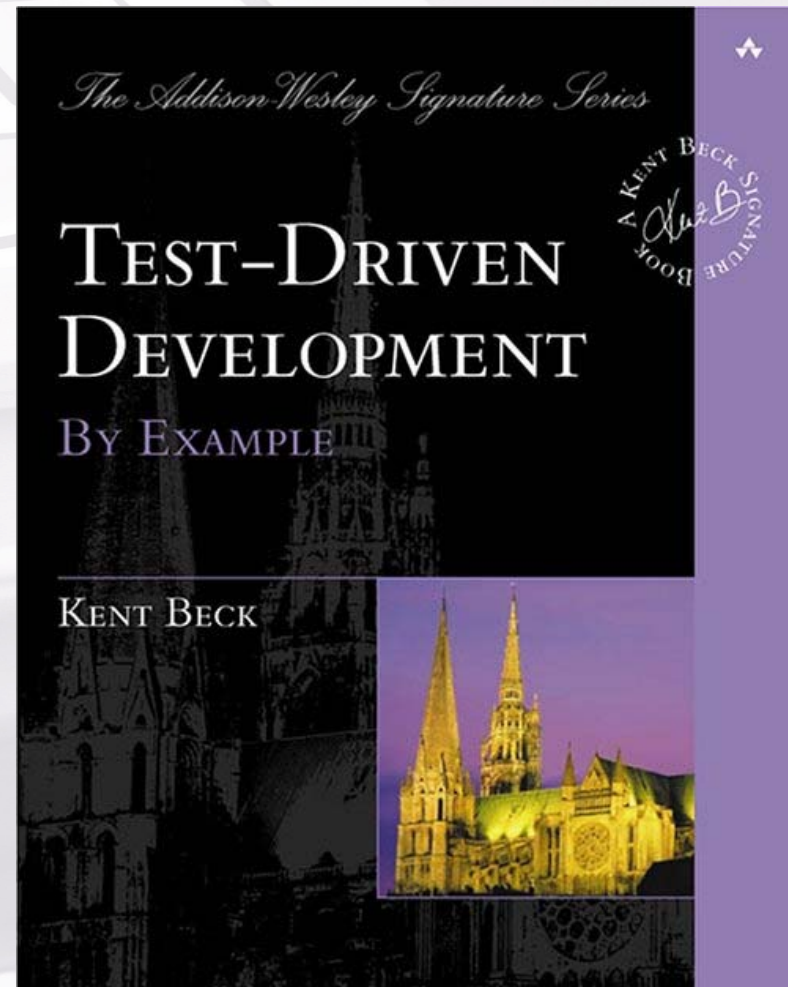
- Process
- Organization
- Collaboration
- Automation
- Participate in Planning
- **Test first development**



# Test First

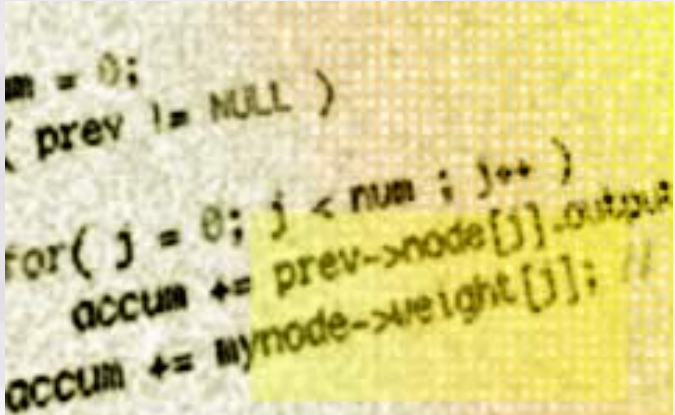
- Many programmers have adopted this approach with near religious zeal
- “Test-first design is infectious! Developers swear by it. We have yet to meet a developer who abandons test-first design after giving it an honest trial.”

Robert C. Martin



# Getting the Code Right vs. Getting the Right Code

- Test first development ensures the code is doing what the developer intended it to do
- Acceptance testing can now focus on determining how well the developer's intent matches client's expectations



```
num = 0;  
( prev != NULL )  
  
for( j = 0; j < num; j++ )  
    accum += prev->node[j].output  
    accum += mynode->weight[j]; //
```



# Agile in Action

- Process
- Organization
- Collaboration
- Automation
- Participate in Planning
- Test first development



# Thanks for coming

- Let us know about your testing work. We'd like to hear about your successes and your difficulties.
- My e-mail address is:  
tim@qualsys.org

