

Black Box Software Testing

(Academic Course - Fall 2001)

Cem Kaner, J.D., Ph.D.

Florida Institute of Technology

Section: 6 :

An Overview of Product Development

Contact Information:

kaner@kaner.com

www.kaner.com (testing practitioners)

www.badsoftware.com (software law)

www.testingeducation.org (education research)

Copyright (c) Cem Kaner 2001.

I grant permission to make digital or hard copies of this work for personal or classroom use, without fee, provided that (a) copies are not made or distributed for profit or commercial advantage, (b) copies bear this notice and full citation on the first page, and if you distribute the work in portions, the notice and citation must appear on the first page of each portion. Abstracting with credit is permitted. The proper citation for this work is Cem Kaner, *A Course in Black Box Software Testing (Academic Version)*, Fall 2001, www.testing-education.org.

To copy otherwise, to republish or post on servers, or to distribute to lists requires prior specific permission and a fee. Request permission to republish from kaner@kaner.com.

Black Box Software Testing

An Overview of Product Development

Assigned Reading:

Selection on waterfall model from Kruchten, Philippe, 2000. *The Rational Unified Process, an Introduction, 2nd edition.*

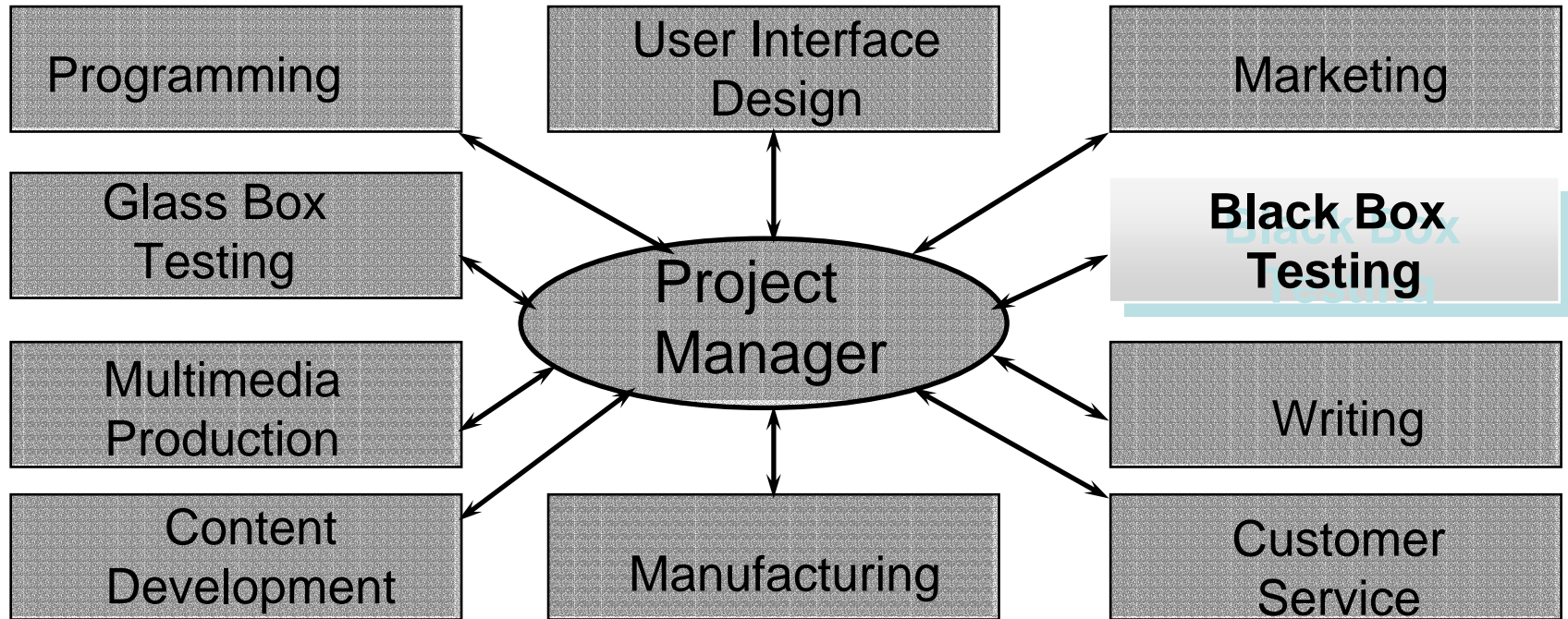
Supplementary Reading:

Beck, Kent, 1999. *Extreme Programming Explained.* Addison-Wesley.

**Coyote Valley Software's *Product Lifecycle Model:*
<http://www.coyotevalley.com/plc/builder.htm>**

Gilb, Tom. 1997. *Evo: The Evolutionary Project Managers Handbook.*
(<http://www.result-planning.com/>, click on Download Center.) If this document isn't available, check for related material at <http://www.stsc.hill.af.mil/swtesting/gilb.asp>

Product Development Organization



During development, several groups (or people) act as service providers to the Project Manager. They often play other roles beyond this, but during development, it pays to look at them as members of a managed group that seeks to ship a high quality, salable product on time.

Product Development Organization

- Project Manager: (Program Mgr.) *Responsible for shipping a salable product on time and within budget.*
- Architect: *Design the product's overall (code & data) structure, relationships among components, and relationships with other expected parts of the customers' system.*
- User Interface Designers: *Design to make the program usable and useful.*
- Programmers: (Development Mgr.) *Design and write/test the code.*
- Glass Box Testers: *Use knowledge of the internals of the code to drive the testing. Along with glass box test techniques, use code inspections and other code-aware methods to find errors.*

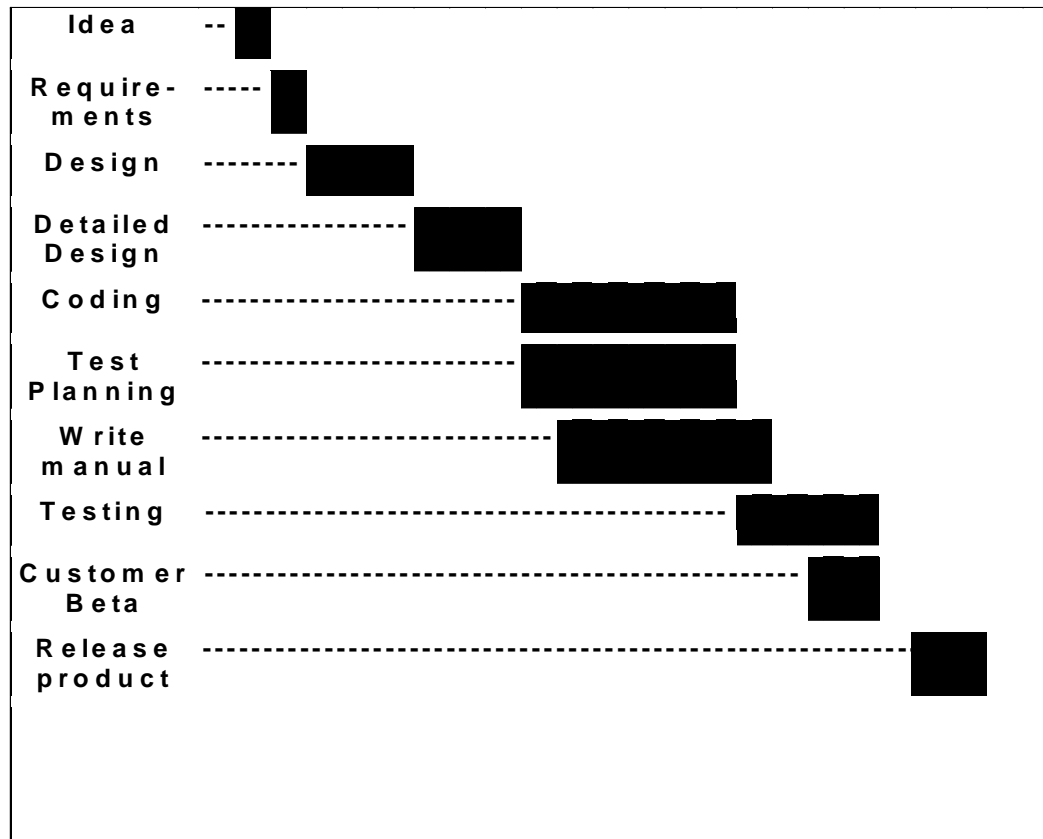
Product Development Organization

- Tech Writers: *Write the user manual, help, etc.*
- Content Developers: *There may be several different groups here, animating, filming, writing and performing music, writing, etc.*
- Multimedia Producer: *Responsible for managing content development, selecting content to include with the product, negotiating rights, etc.*
- Marketing: (Product Manager) *Manage the product's overall profit/loss plan. Evaluate competitors, appraise sub-markets (e.g. should we be compatible with a certain printer?), design packaging and advertisements, run trade shows, etc.*
- Black Box Testers: *Test from the customer's perspective.*

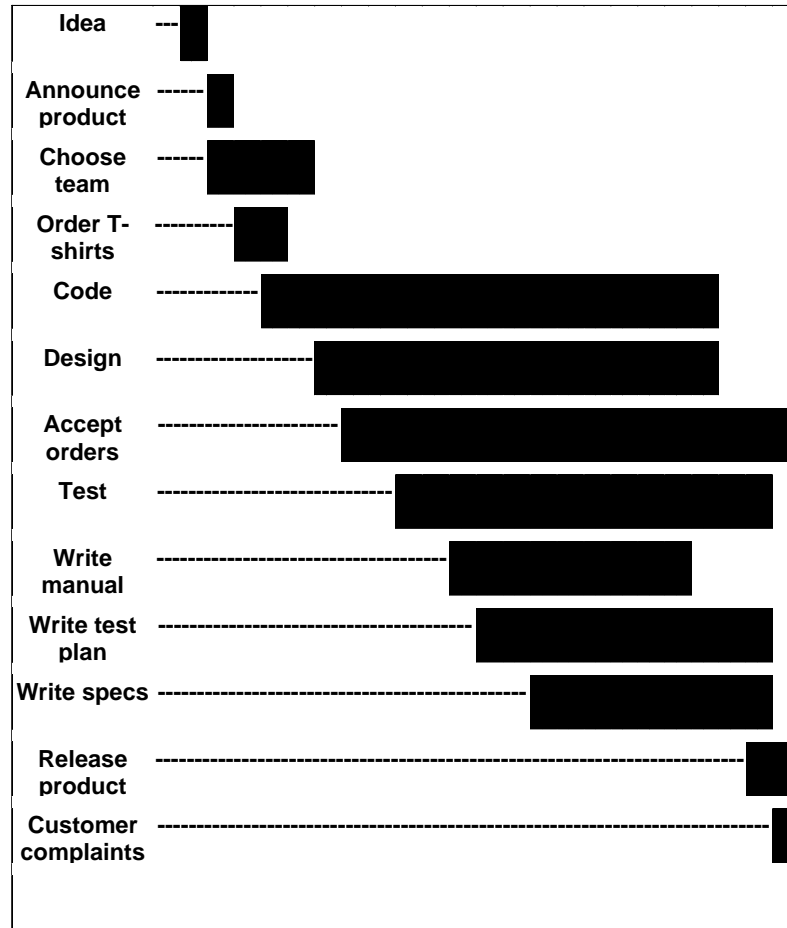
Product Development Organization

- **Customer Service:** *Help evaluate the effect of design decisions, bugs, and existing problems on customers.*
- **Manufacturing:** *Buy materials. Help everyone control their contribution to the cost of goods. Drive the late-stage schedule. Make the product.*

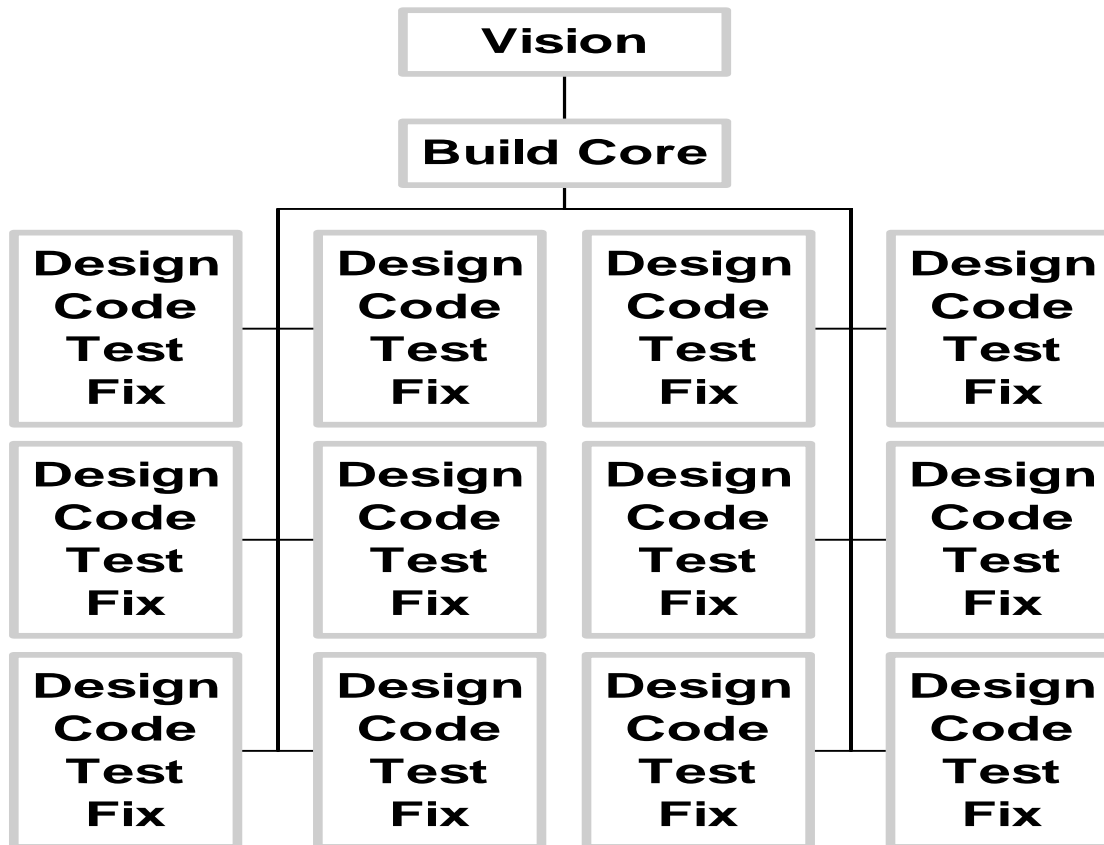
The Waterfall Model for Software Development



What the Waterfall Really Looks Like



Evolution



Evaluating a Life Cycle

- When you evaluate a life cycle model, consider the tradeoffs that the project manager must make among
 - Features
 - Reliability
 - Price
 - and Time.
- Ask yourself what life cycle model best complements the skills and weaknesses of this project's manager. Which cycle best supports this product's risk profile?

Product Development Milestones

- Milestones are moments of truth for a project. They define synchronization points that help groups work as a team. They do this by collecting lists of prerequisite tasks. (“Complete all these tasks in order to reach the milestone.”) Tasks that trigger events (e.g. UI freeze) become milestones.
 - Design and specification complete
 - First functionality
 - Almost alpha
 - Alpha
 - Pre-beta
 - Beta
 - User interface freeze
 - Pre-final
 - Final
 - Release

Product Development Milestones

- **Product design and specification complete**

We know what is to be built, what it will look like, what benefits it will provide, and how it will work.

- **First functionality**

The program has just enough capability that you can try it out.

- **Almost alpha**

The alpha milestone will hit in a few days or weeks. Before the program can be declared “alpha” the Testing Group has to check that it meets the alpha criteria.

- **Alpha**

A significant (and agreed-upon) portion of the product has been completed. (The “product” includes code, documentation, additional art or other content, etc.) The product is ready for first in-house use.

Product Development Milestones

- **Pre-beta**

A *beta candidate* has been submitted. If it meets the beta criteria (as verified by Testing), the software is beta.

- **User interface freeze**

Every aspect of the user interface of the software has been frozen. Some companies except error messages and help screens.

- **Beta**

Most or all of the product is complete. Some companies send “beta copies” of the product to customers at this milestone.

- **Pre-final**

A *final candidate* has been submitted. If all the pieces are there and other tests are met, Final.

- **Final**

Last wave of testing before it gets inflicted on customers.

- **Release**

Variation in Milestone Definitions

- *Alpha definitions:*
 - a. All code complete (but not necessarily working)**
 - b. Customer useful (but not necessarily fully coded)**
 - c. Core functionality complete (but not complete and not useful)**
- *Beta definitions:*
 - a. Reasonably predictable as X weeks (e.g. 6) before shipment.**
 - All known Level 1 errors corrected.
 - No bugs are known that look so complex that they might not be fixed before shipment
 - X% (such as 90%) of all Level 2 bugs reported have been closed and the total number of remaining bugs is less than K
 - b. Customer useful. All critical features coded. No known bugs that could damage an external beta user's data or embarrass the publisher**
 - c. Customer useful. Some features are still absent.**

The point is not to decide which definition is correct.

The point is to understand which definition is in use.

Variation in Milestones

- Milestones are defined quite differently by different development groups.
 - Refer to *Testing Computer Software*, “The Development Time Line” p. 266-302 for one set of milestone definitions and examples.
 - Another example of detailed definitions comes from Brian Lawrence and Bob Johnson of Coyote Valley Software
 - The testing group doesn’t define the life cycle model or the milestones. But you can do the project a huge service by listing a wide range of tasks and working with the project manager to sort the tasks into milestones (must be completed by Milestone X.)

A Product Life Cycle Model - Netscape
File Edit View Go Communicator Help **PLM LIVE**
Location: <http://www.coyotevalley.com/plc/builder.htm>

Coyote Valley Software

A Product Life Cycle (PLC) Model

Introduction

The term "PLC" has stood for both a **Product** life cycle and a **Project** life cycle. We believe that a project is many times a subset of a product. Thus, products consist of one or more projects.

Many projects fail because they do not consider the bigger product view, the smallest of projects may need to "sell" their results, announce their completion, and in general do many of the tasks required of larger products being delivered to customers. Therefore, we suggest you keep an open mind about your need to do the activities suggested by these checklists.

We are not blind to the fact that so many PLCs have been slayers of countless trees - behests tomes which end up gathering dust on engineers and managers bookshelves. We hope your effort is better received. Our approach was to follow a simple design objective:

It should be easier to complete my project using this PLC than without it!

With this idea in mind we constructed a set of tools we think is useful, but it's helpful to understand our philosophy when learning how to use them. There is a lot of material presented here. We do not believe every organization needs to implement this entire PLC. Think of this as a shopping list. Look at each item and ask "Is it right for us?" Pick and choose. Read the reasons. we've included these items. Remember you can always add or remove items later.

Views

There are many ways to build a PLC; here we present three approaches to the problem.

- A Minimal set. This will get you started with checklists for a simple Alpha, Beta, First Customer Ship (FCS) system.
- A Concise set. This is a complete Mini-PLC, with checklists for major and minor milestones.
- The Stearngasboard (A work in progress). This is all the Milestones in a beginning to end structure. Included are checklists plus templates of various deliverables.

Getting Help

If you want to save yourself and your organization time and trouble, Coyote Valley Software can provide you with one- or two-day seminars on this PLC model. Please read mail for rates and availability.

Credits

What's presented here is based on the hard work by lots of people over several years at least four different companies, spanning continents, and uncorked bottles of fine wine.

About the Authors

- Brian Lawrence
- Bob Johnson and more Bob

Document Done