# Required Level of Study

Reading: about 60 pages per week

Doing: some work each week

Writing: grammatically correct English documents

Individual Effort:
your grade is based on your effort and performance

Time Commitment:
9-12 hours per week outside of class

## Meetings for Discussion

Read assigned material *before* class each week.

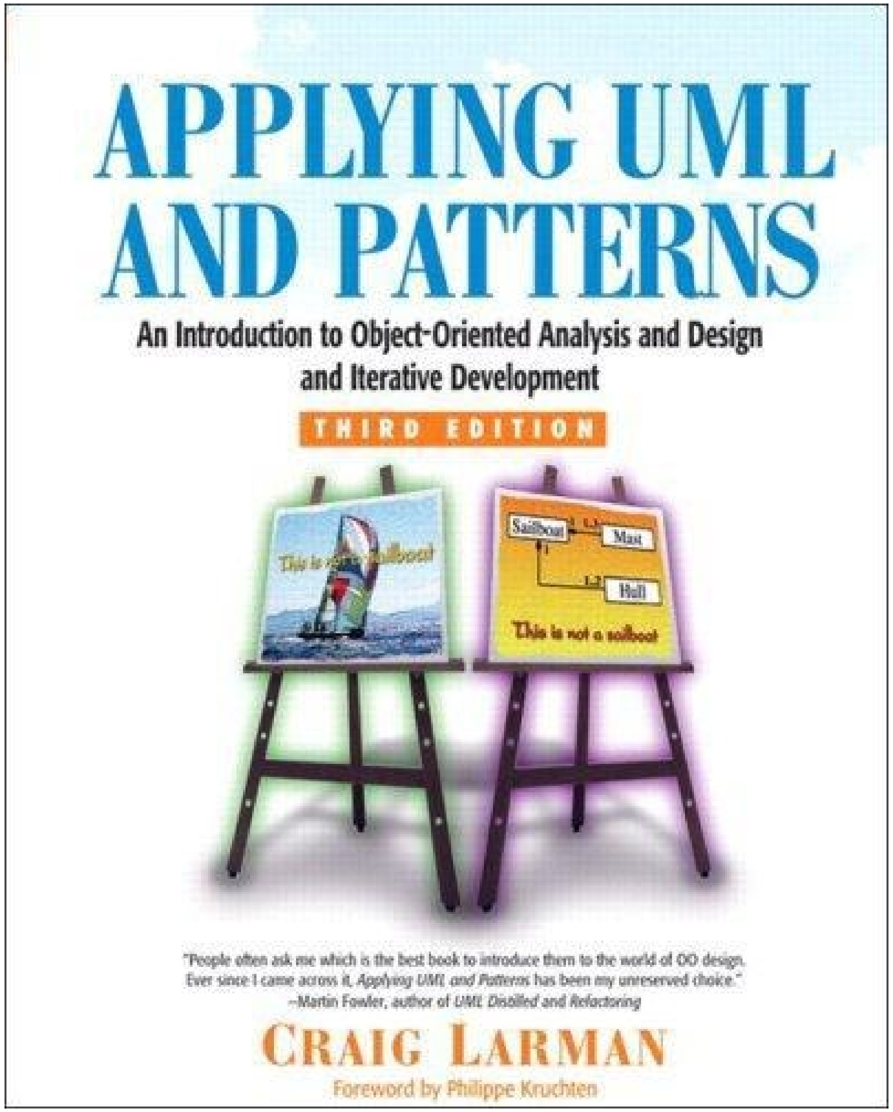
Take notes: summarize important points.

Participate in discussions.

# Main Textbook

*"People often ask me which is the best book to introduce them to OO design.*
*... Applying UML and Patterns has been my unreserved choice."*

-- Martin Fowler

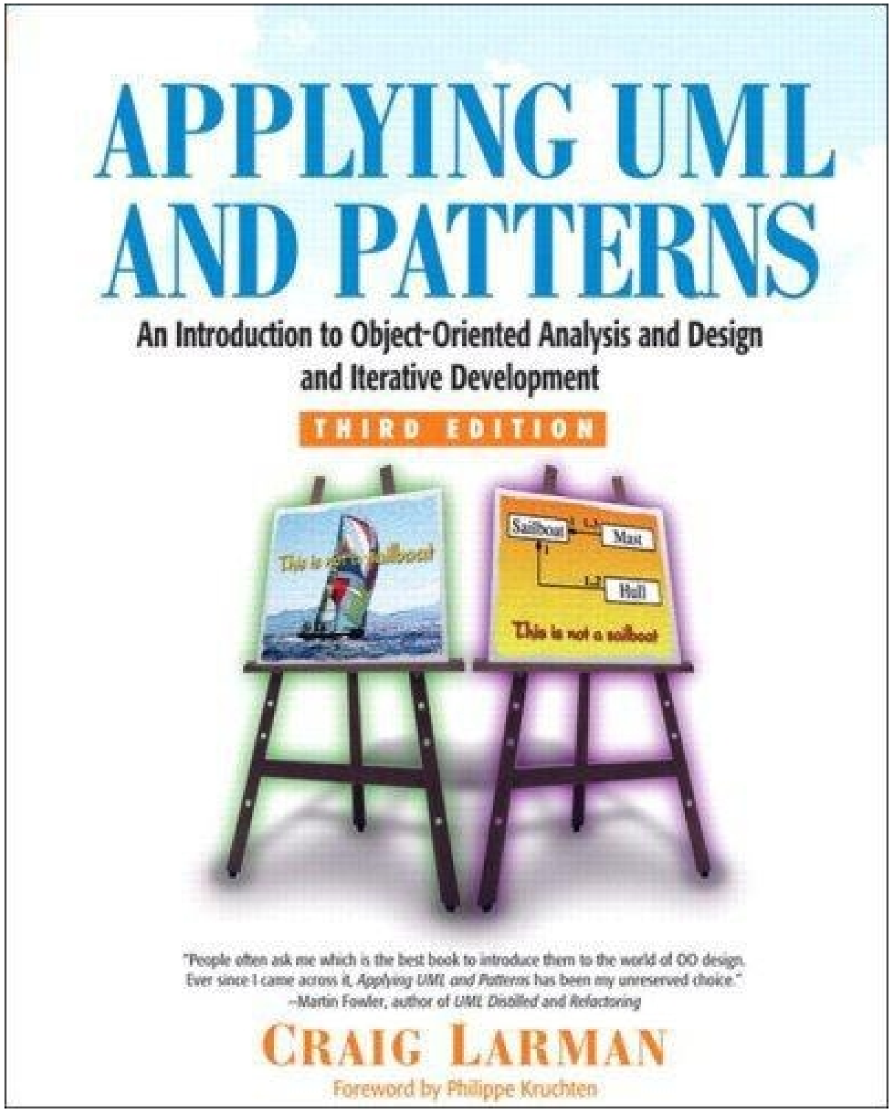

APPLYING UML AND PATTERNS
An Introduction to Object-Oriented Analysis and Design and Iterative Development
THIRD EDITION
This is not a sailboat
Sailboat
Mast
Hull
This is not a sailboat
"People often ask me which is the best book to introduce them to the world of OO design. Ever since I came across it, Applying UML and Patterns has been my unreserved choice." –Martin Fowler, author of UML Distilled and Refactoring
CRAIG LARMAN
Foreword by Philippe Kruchten

Was 700 Baht at KU Books (discount for this course).

# Main Textbook

*If you don't read the book,*

*you will not pass this course.*


APPLYING UML AND PATTERNS

An Introduction to Object-Oriented Analysis and Design and Iterative Development

THIRD EDITION

"People often ask me which is the best book to introduce them to the world of OO design. Ever since I came across it, *Applying UML and Patterns* has been my unreserved choice."
—Martin Fowler, author of *UML Distilled* and *Refactoring*

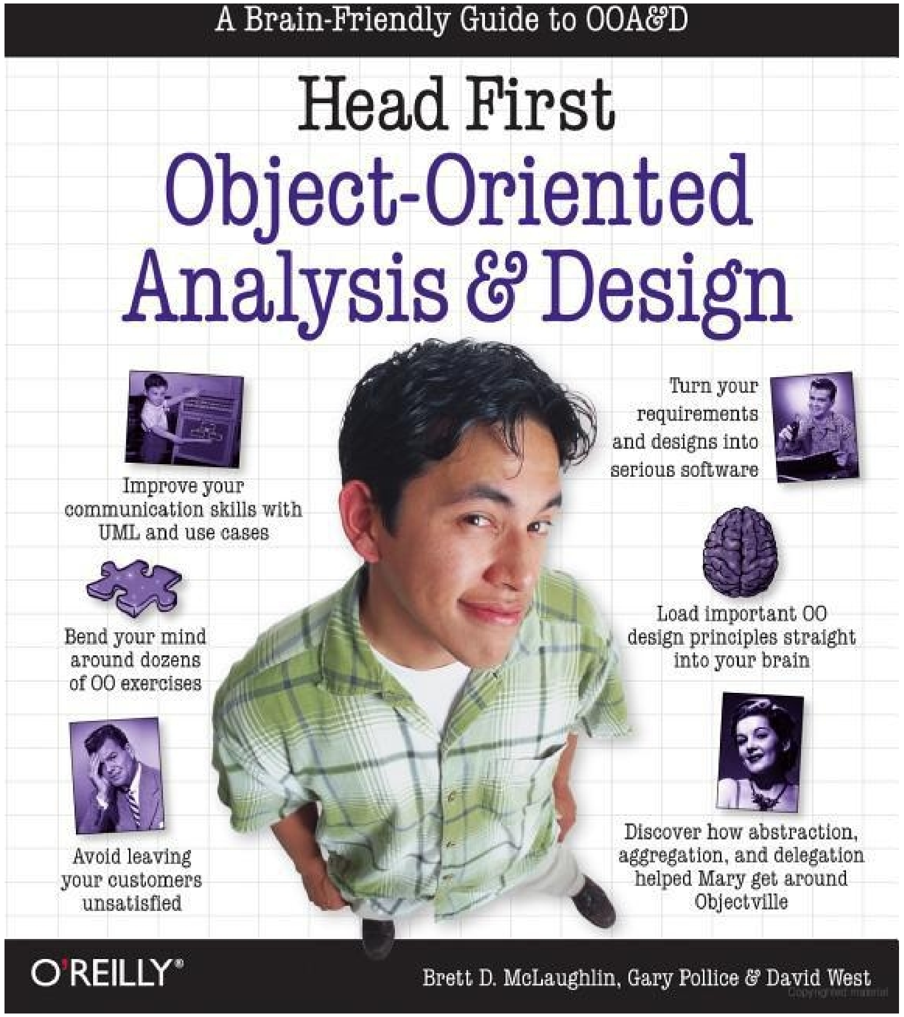
CRAIG LARMAN

Foreword by Philippe Kruchten

## Other Books

For OO Design Principles, we will use some chapters.

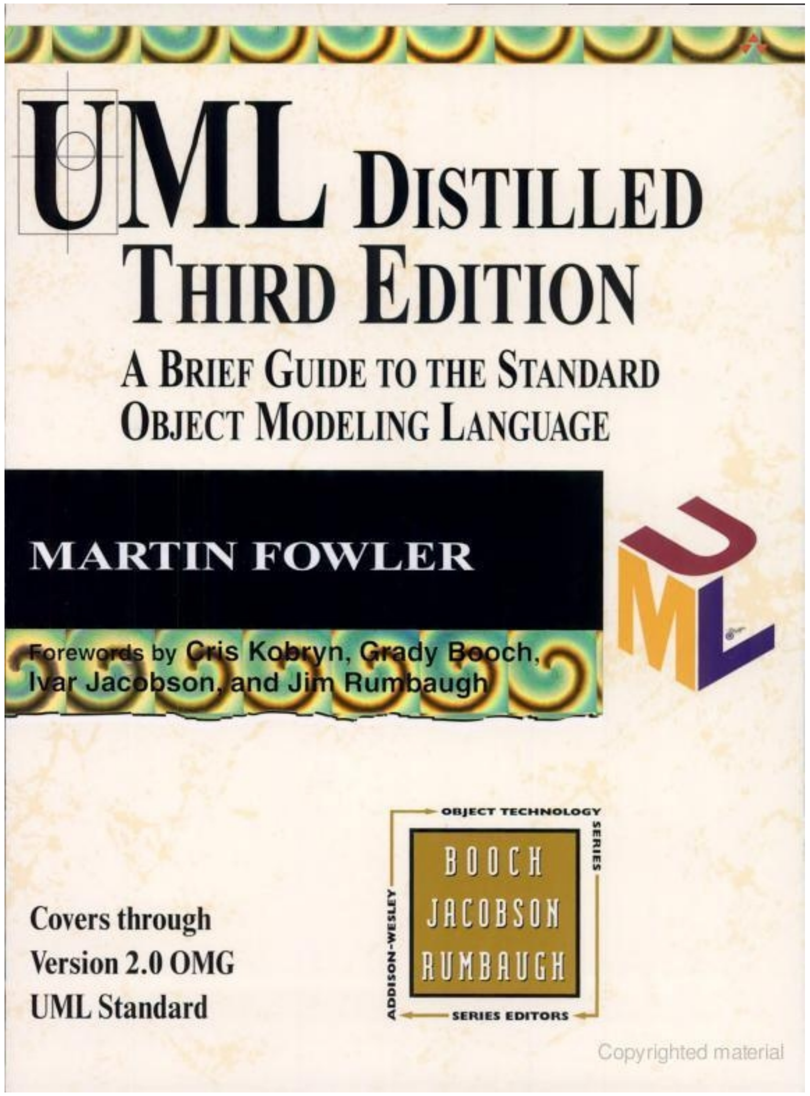
Amusing case studies make it easy to remember the principles.


A Brain-Friendly Guide to OOA&D
Head First
Object-Oriented Analysis & Design
Improve your communication skills with UML and use cases
Turn your requirements and designs into serious software
Bend your mind around dozens of OO exercises
Load important OO design principles straight into your brain
Avoid leaving your customers unsatisfied
Discover how abstraction, aggregation, and delegation helped Mary get around Objectville
O'REILLY
Brett D. McLaughlin, Gary Pollice & David West

# Other Books

For learning UML.

Chapter 2 is good intro to software process.



UML DISTILLED
THIRD EDITION
A BRIEF GUIDE TO THE STANDARD OBJECT MODELING LANGUAGE

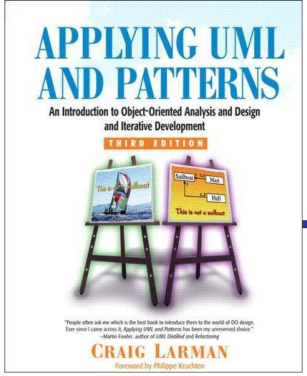
MARTIN FOWLER

Forewords by Cris Kobryn, Grady Booch, Ivar Jacobson, and Jim Rumbaugh

Covers through Version 2.0 OMG UML Standard

OBJECT TECHNOLOGY SERIES
BOOCH
JACOBSON
RUMBAUGH
SERIES EDITORS
ADDISON-WESLEY

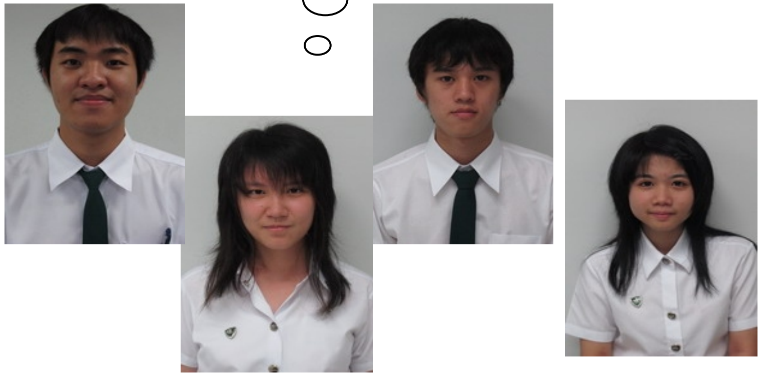
Copyrighted material


APPLYING UML AND PATTERNS
An Introduction to Object-Oriented Analysis and Design and Iterative Development
THIRD EDITION
CRAIG LARMAN

## Using the Textbook

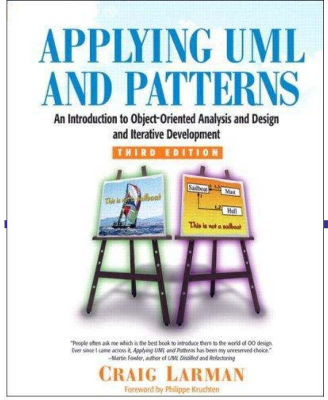
- Easy to read, but written for developers

what ???



Agile UP Inception is not Waterfall Requirements phase... involve Stakeholders ... Risk ...

Craig Larman


APPLYING UML AND PATTERNS
An Introduction to Object-Oriented Analysis and Design and Iterative Development
THIRD EDITION
CRAIG LARMAN
Foreword by Philippe Kruchten

## Textbook

Strengths:

- Excellent for O-O Analysis and Design
- Emphasizes iterative development
- Author conveys real-world experience

Weaknesses:

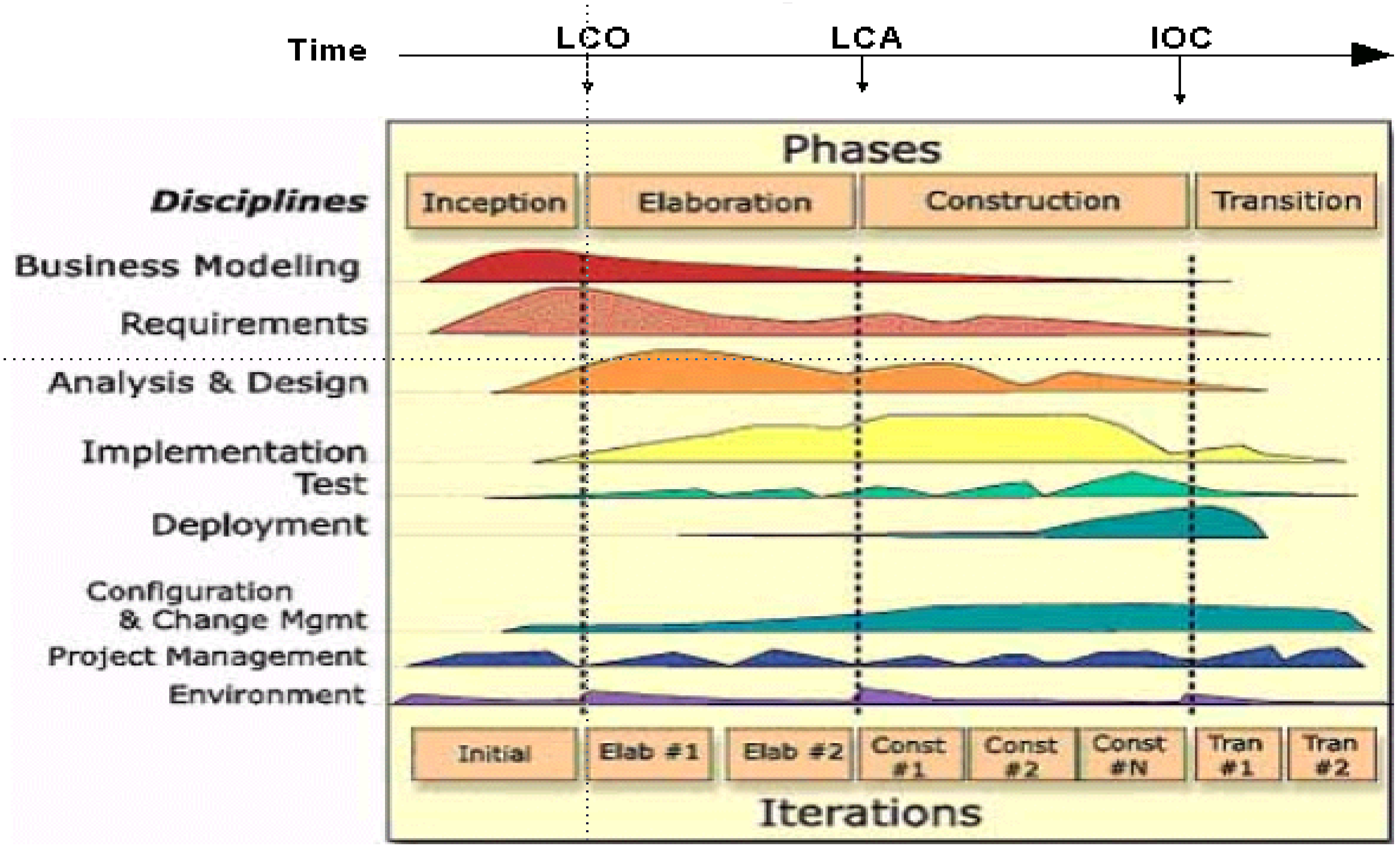
- Fuzzy about process and how docs fit together.
- Bias towards Agile methods.
- Some gaps you have to research for yourself.
- Redundant

## Software Process

- Requirements, Analysis, and Design are part of any *software process.*
- Hence, it helps to know where they fit in and when to do them.
- Larman favors an *Agile* UP (Unified Software Developement Process).

# UP Model

- Phases, iterations, 'disciplines' for kinds of activities

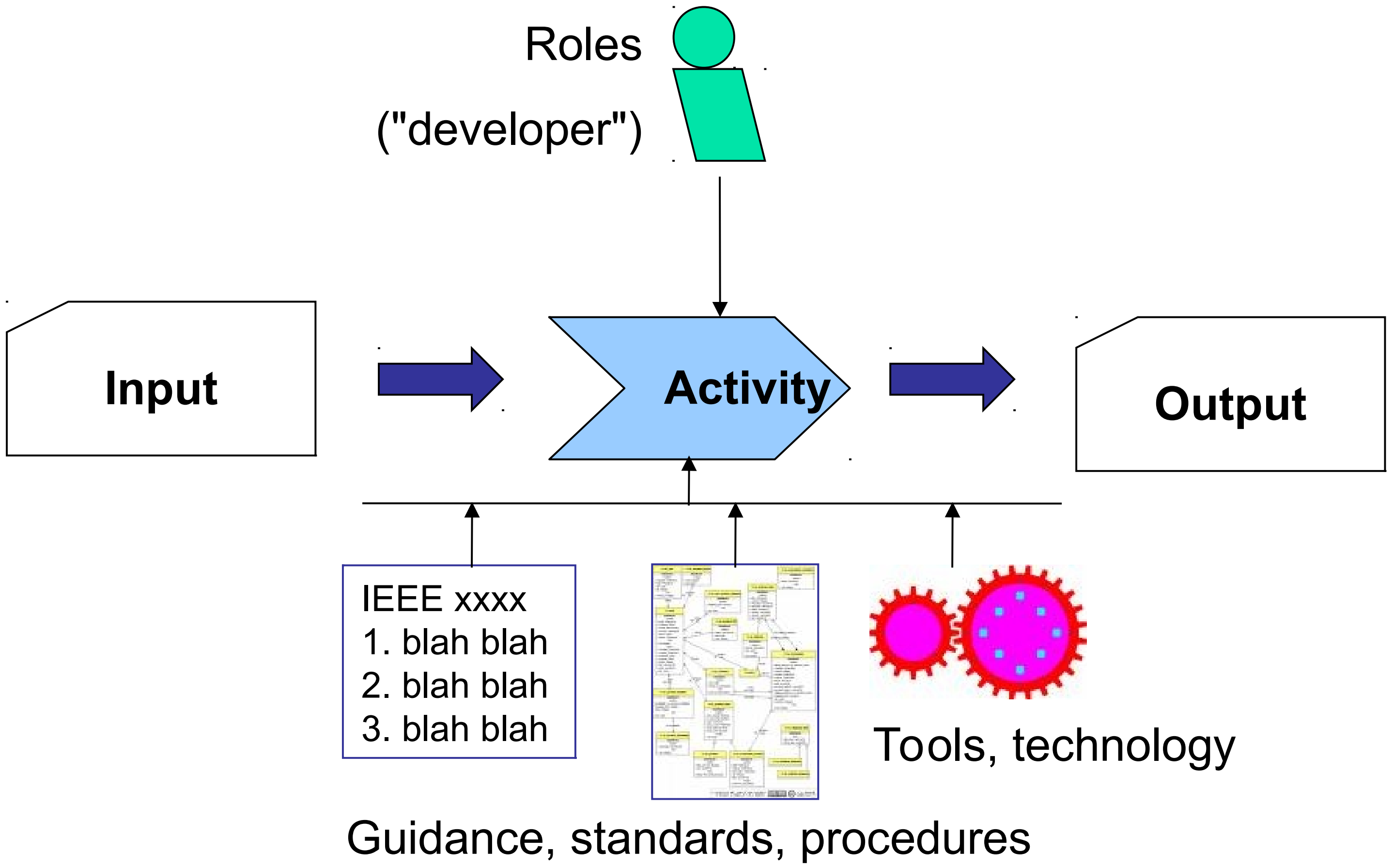


## Characteristics of U.P.

- Plan based, but accommodate change
- Architecture centric
- Identify & address risks early
- implement requirements based on business value, architecture, and risk
- handle risky requirements early
- prioritize requirements that have a big impact on software architecture
- time-boxed iterations with milestones
- UP is a "framework". Tailor it for your project.

# Activities and Artifacts

UP can be agile, but prefers tangible work products



## Key Artifacts

Vision Statement

Business Case

Software Requirements Specification (SRS)

- Functional Requirements (Use Cases)
- Other requirements, "URPS+"
- Acceptance Criteria

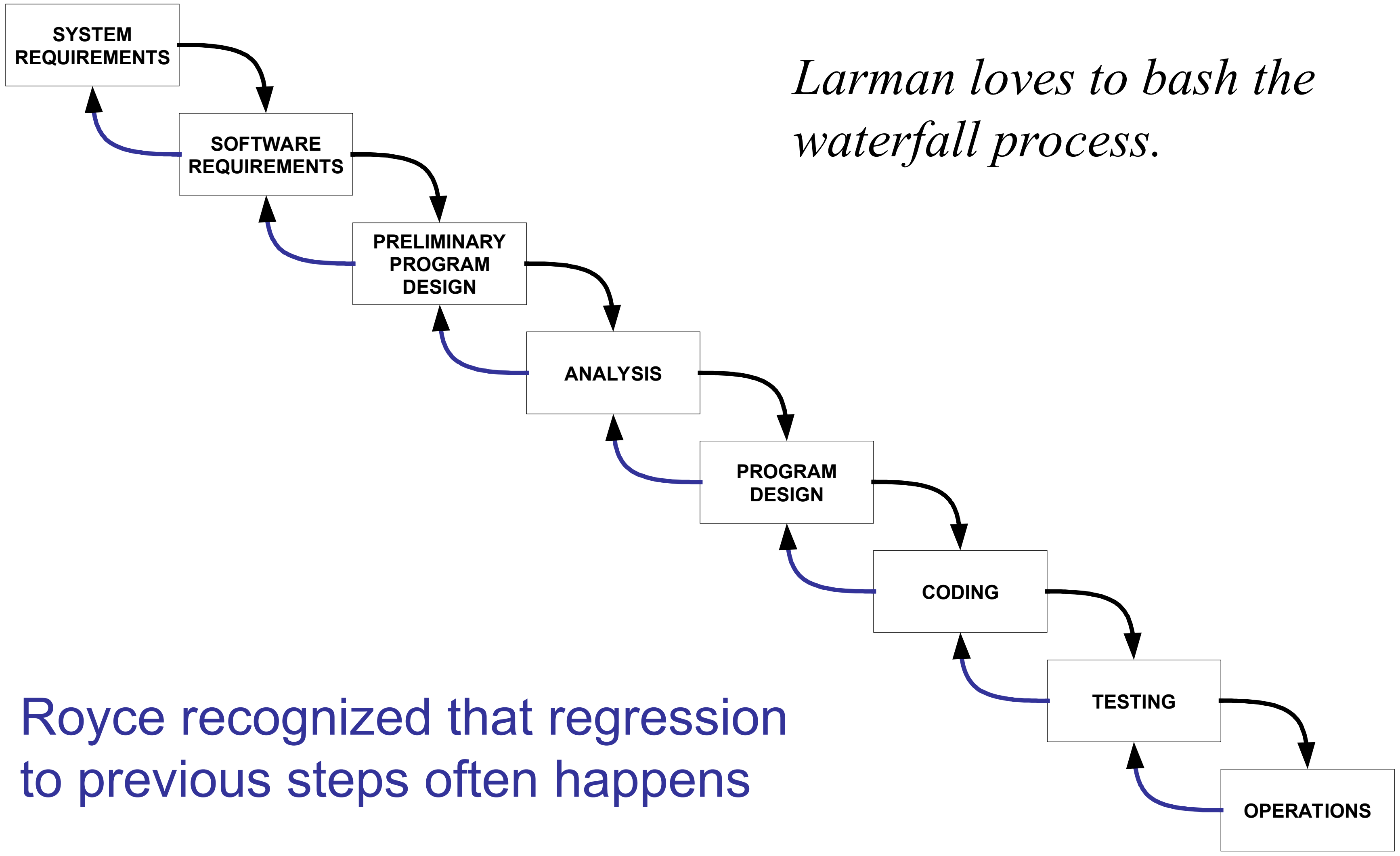
Software Development Plan

Iteration Plans

Project Measurements (many kinds)

Software (of course)

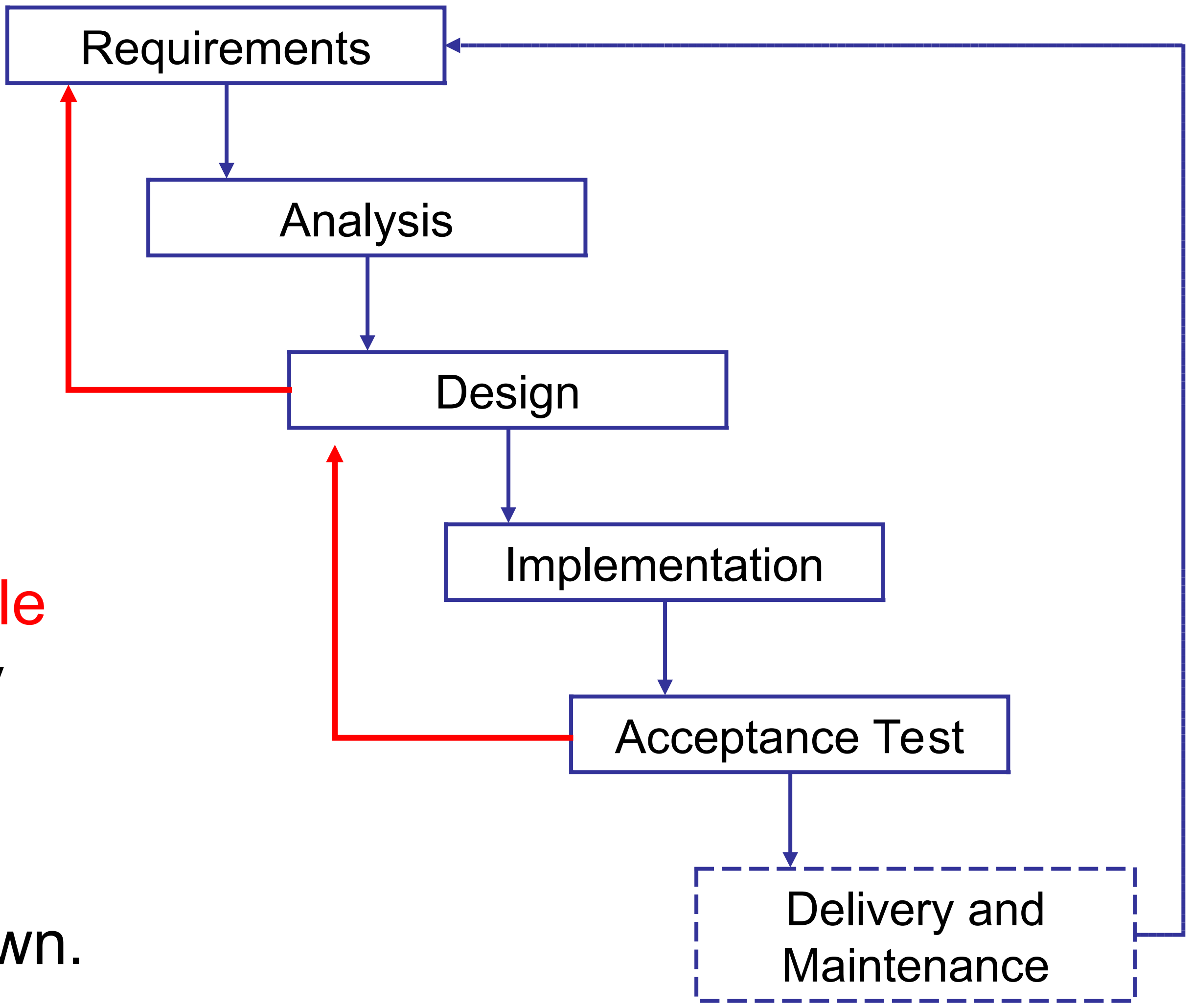
# The Royce Waterfall Model (1970)



*Larman loves to bash the waterfall process.*

Royce recognized that regression to previous steps often happens

# Waterfall Loopback in Reality



Royce found that backtracking multiple phases was usually necessary.

The most common backtracks are shown.

# Importance of Good Requirements

"*The hardest single part of building a software system is deciding what to build.*

*No other part of the conceptual work is as difficult as establishing the detailed technical requirements.... No other part of the work so cripples the resulting system if done wrong. No other part is as difficult to rectify later...*"

"*No Silver Bullet*" by Frederick Brooks. *Computer*, 1987

## Intrinsic Complexity of Software

"*There is no single development, in either technology or in management technique, that promises even one order of magnitude improvement in productivity, in reliability, in simplicity.*"

"*No Silver Bullet*" by Frederick Brooks. *Computer*, 1987

See also page 5-6 for Brooks comments about object-oriented approach.