Are You Agile or Fragile?

Rethinking Software Development

Scott W. Ambler

President, Ronin International

scott.ambler@ronin-intl.com

www.agilemodeling.com

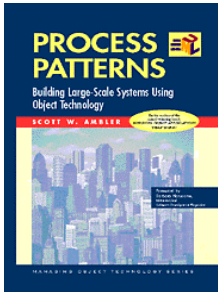
www.agiledata.org

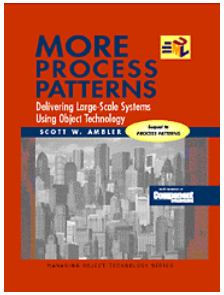
Scott W. Ambler

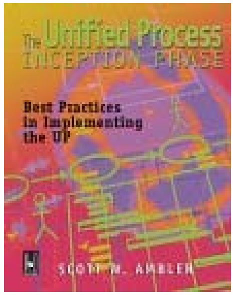
- **Consultant:**
  - Agile Modeling
  - Software Process Mentoring
  - Object/Component Development Mentoring
- **Author:**
  - Agile Modeling
  - Agile Database Techniques
  - The Object Primer 3rd Edition
  - The Unified Process Series (CMP Books)
  - Process Patterns & More Process Patterns
  - www.ambysoft.com/booksAmbler.html
- **Contributing Editor/Writer:**
  - Software Development
  - Computing Canada
  - IBM DeveloperWorks

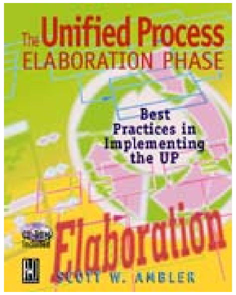
My Process Background

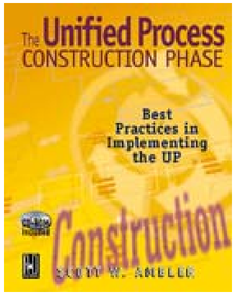
- The Pinball SDLC
- The Object-Oriented Software Process
- The Rational Unified Process (RUP)
- The Enterprise Unified Process (EUP)
- Agile Modeling
- Agile Data

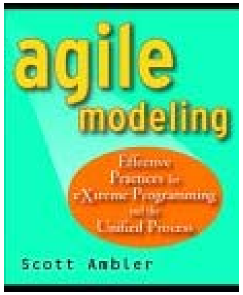


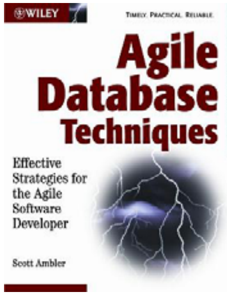


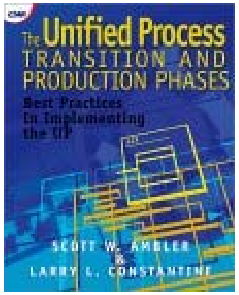







agile modeling
Effective Practices for eXtreme Programming and the Unified Process
Scott Ambler


WILEY
Timely. Practical. Reliable.
Agile Database Techniques
Effective Strategies for the Agile Software Developer
Scott Ambler


The Unified Process
TRANSITION AND PRODUCTION PHASES
Best Practices in Implementing the UP
SCOTT W. AMBLER & LARRY L. CONSTANTINE

Questions?

Don’t be shy.

I’ll take questions during the presentation and at the end.

Overview

- Conversation
- Warning!
- Modern software development
- Leading Agile Processes
- Interesting Observations

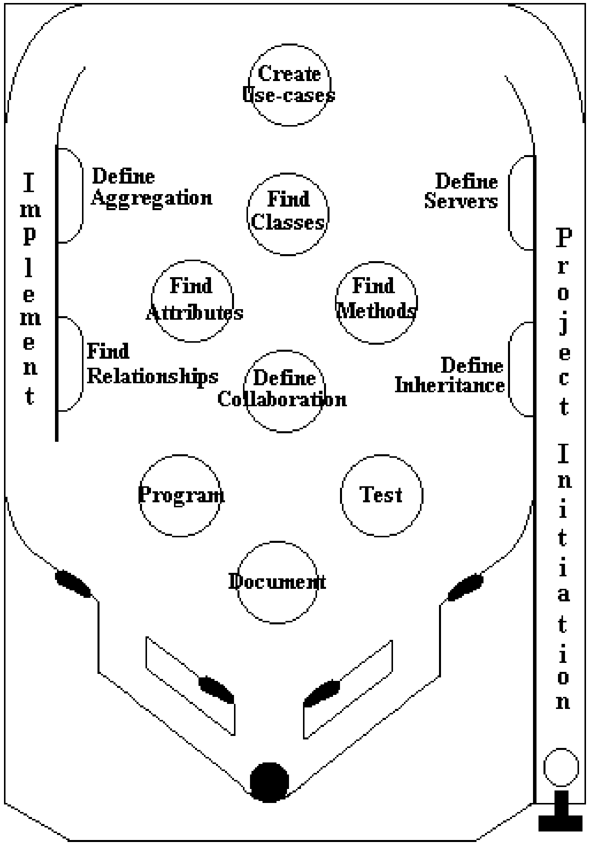
Conversation

- What have you heard about agile software development?

Warning!

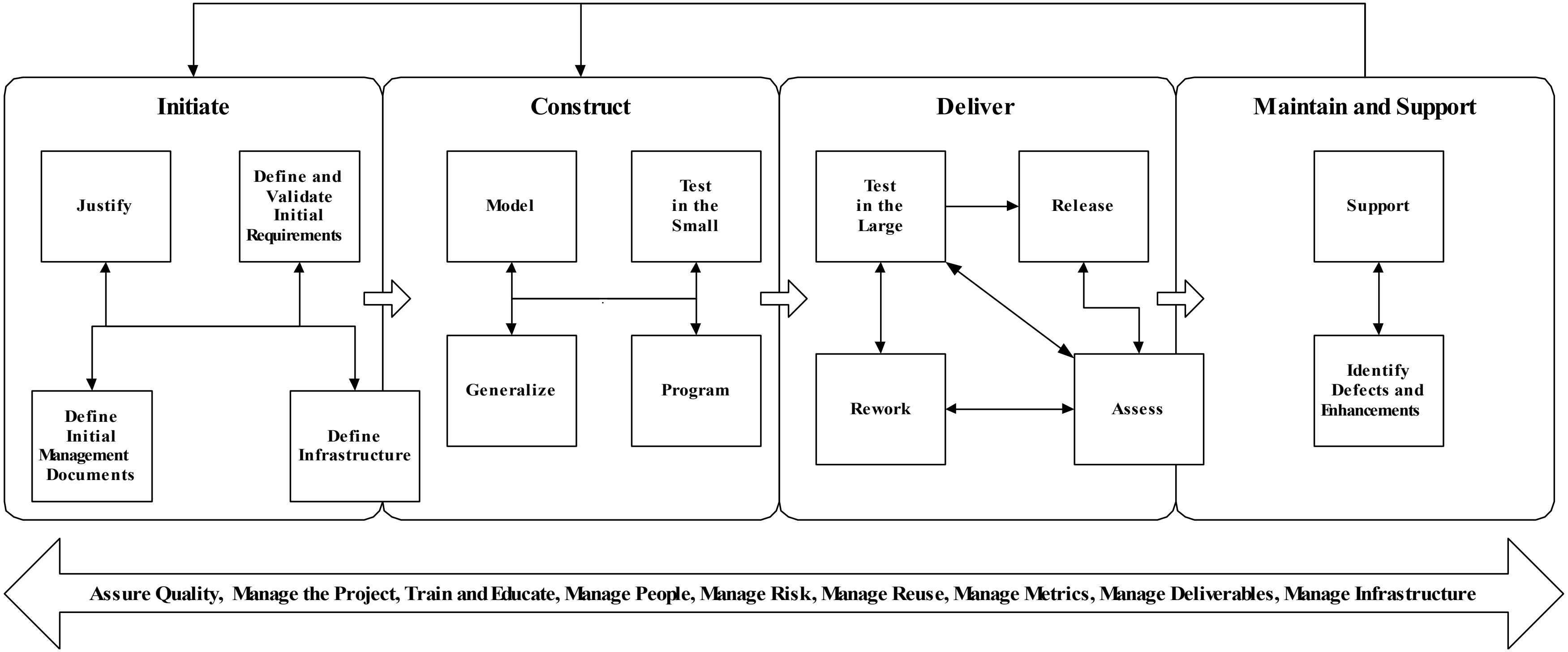
- I’m spectacularly blunt at times
- Many new ideas will be presented
- Some may not fit well into your existing environment
- Some will challenge your existing notions about software development
- Some will confirm your unvoiced suspicions
- Don’t make any “career-ending moves”
- Be skeptical but open minded

The Pinball SDLC (1995)

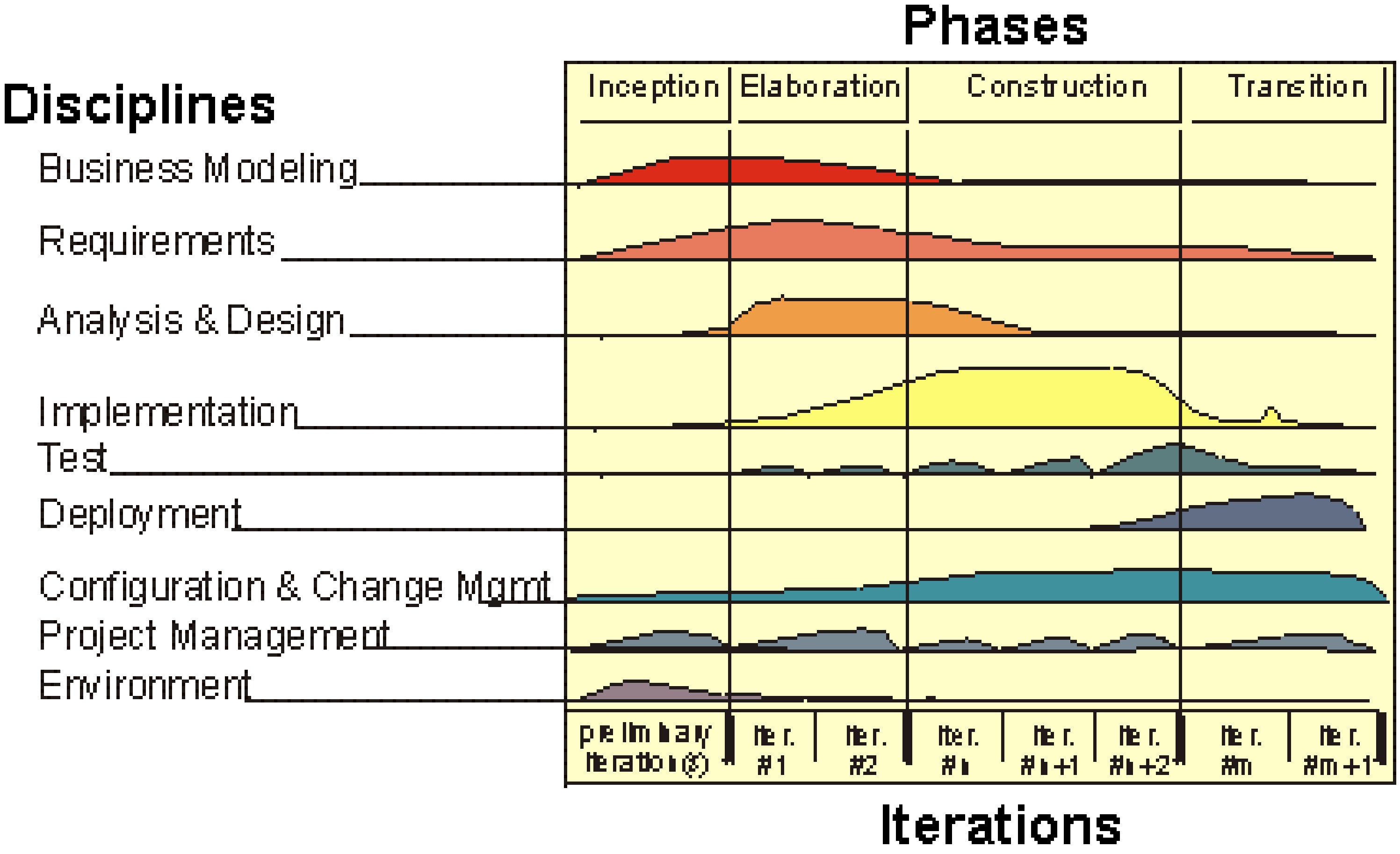


Object-Oriented Software Process (OOSP)

www.ambysoft.com/processPatterns.html

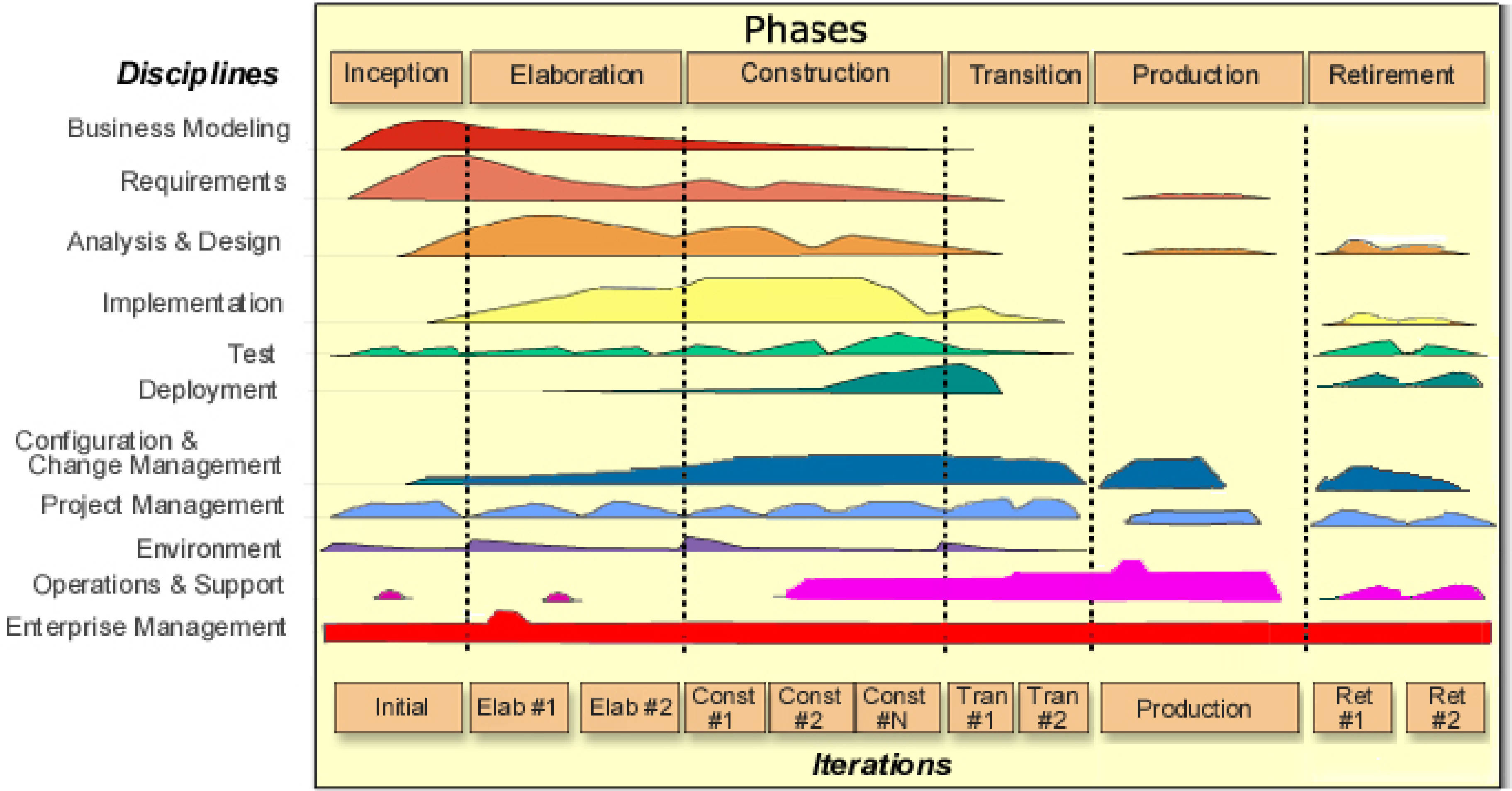

Initiate
Justify
Define and Validate Initial Requirements
Define Initial Management Documents
Define Infrastructure
Construct
Model
Test in the Small
Generalize
Program
Deliver
Test in the Large
Release
Rework
Assess
Maintain and Support
Support
Identify Defects and Enhancements
Assure Quality, Manage the Project, Train and Educate, Manage People, Manage Risk, Manage Reuse, Manage Metrics, Manage Deliverables, Manage Infrastructure

Rational Unified Process (RUP) Contributions



The Enterprise Unified Process (EUP)

www.enterpriseunifiedprocess.info



Agile Modeling (AM)
www.agilemodeling.com

- AM is a chaordic, practices-based process for modeling and documentation.
- AM is a collection of *practices* based on several *values* and proven software engineering *principles*
- AM is a light-weight approach for enhancing modeling and documentation efforts for other software processes such as XP and RUP

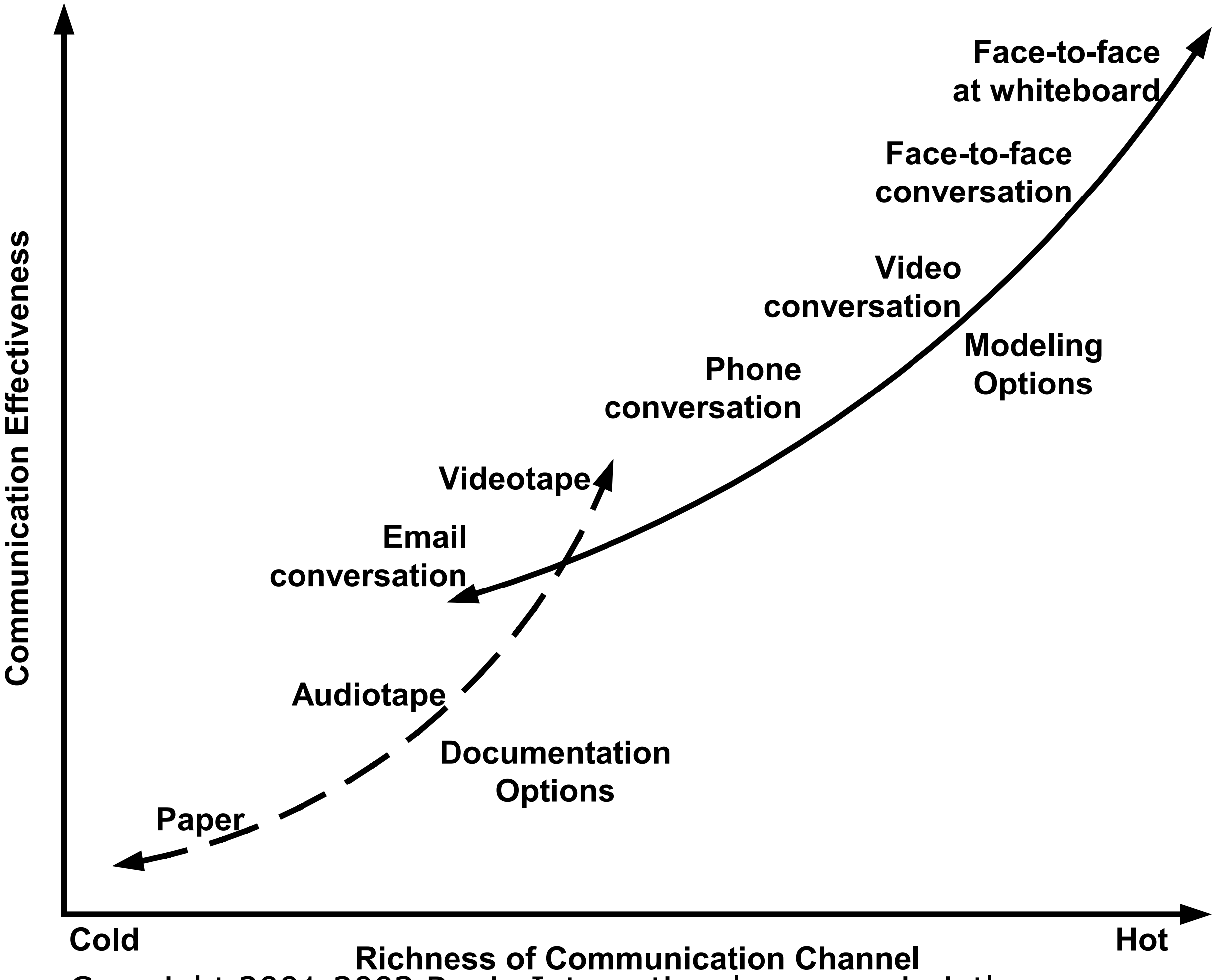
Agile Data
www.agiledata.org

- The Agile Data (AD) method is a collection of philosophies that will enable IT professionals within your organization to work together effectively when it comes to the data aspects of software-based systems.
- Six philosophies:
  - **Data**. Data is one of several important aspects of software-based systems.
  - **Enterprise issues**. Development teams must consider and act appropriately regarding enterprise issues.
  - **Enterprise Groups**. Enterprise groups exist to nurture enterprise assets and to support other groups, such as development teams, within your organization.
  - **Unique situation**. Each development project is unique, requiring a flexible approach tailored to its needs. One software process does not fit all.
  - **Work together**. IT professionals must work together effectively, actively striving to overcome the challenges that make it difficult to do so.
  - **Sweet spot**. Avoid the black and white extremes to find the gray that works best for your overall situation.

Modern Software Development

- Communication
- What is Agile Software Development (ASD)?
- What isn't ASD?
- Agile Values
- Agile Principles
- Signs That You're Fragile

Communication Modes
(Alistair Cockburn)



Agile Software Development

- Agility is the ability to both create and respond to change in order to profit in a turbulent business environment.
- Agile software development is an approach to software development that is people oriented, that enables people to respond effectively to change, and that results in the creation of working systems that meets the needs of its stakeholders.

Agile Software Development is not

- "Code and fix"
- An excuse not to document
- An excuse not to model
- An excuse to short-change quality
- An excuse to ignore enterprise concerns

Agile Values

We value:

1. Individuals and interactions
2. Working software
3. Customer collaboration
4. Responding to change

Over:

1. Processes and tools
2. Comprehensive documentation
3. Contract negotiation
4. Following a plan

Agile Principles

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity--the art of maximizing the amount of work not done--is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Signs That You're Fragile

- IT specialists
- Sign-offs
- Hand-offs between groups
- Desire to outsource
- Reviews
- Documentation focus
- Developers aren't primary champions of the process
- Teams need to "go rogue" to get anything done
- Becoming ISO, CMM, or 6 Sigma compliant is a primary goal
- Tool-focused process
- One "process size fits all" mentality

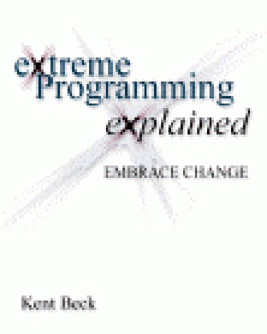
Leading Agile Software Processes

- Extreme Programming (XP)
- Scrum
- Dynamic System Development Method (DSDM)
- Feature Driven Development (FDD)
- Crystal Clear
- "Agile RUP"
- Agile Modeling
- Agile Data

Extreme Programming (XP)

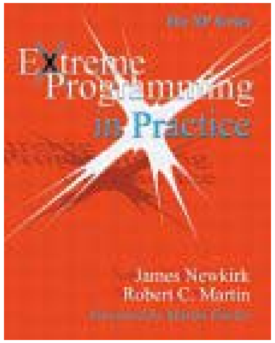
www.xprogramming.com

- The Planning Game
- Small Releases
- Metaphor
- Simple Design
- Testing
- Refactoring
- Pair Programming
- Collective Ownership
- Continuous Integration
- 40-Hour Week
- On-Site Customer
- Coding Standards
- Daily Stand Up Meeting


extreme Programming explained

EMBRACE CHANGE

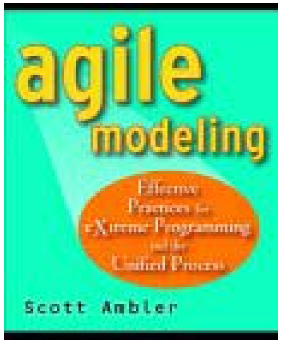
Kent Beck


The XP Series

Extreme Programming in Practice

James Newkirk

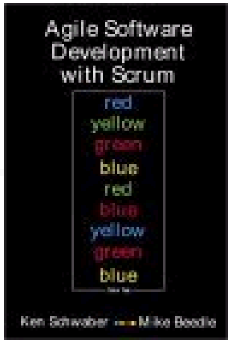
Robert C. Martin

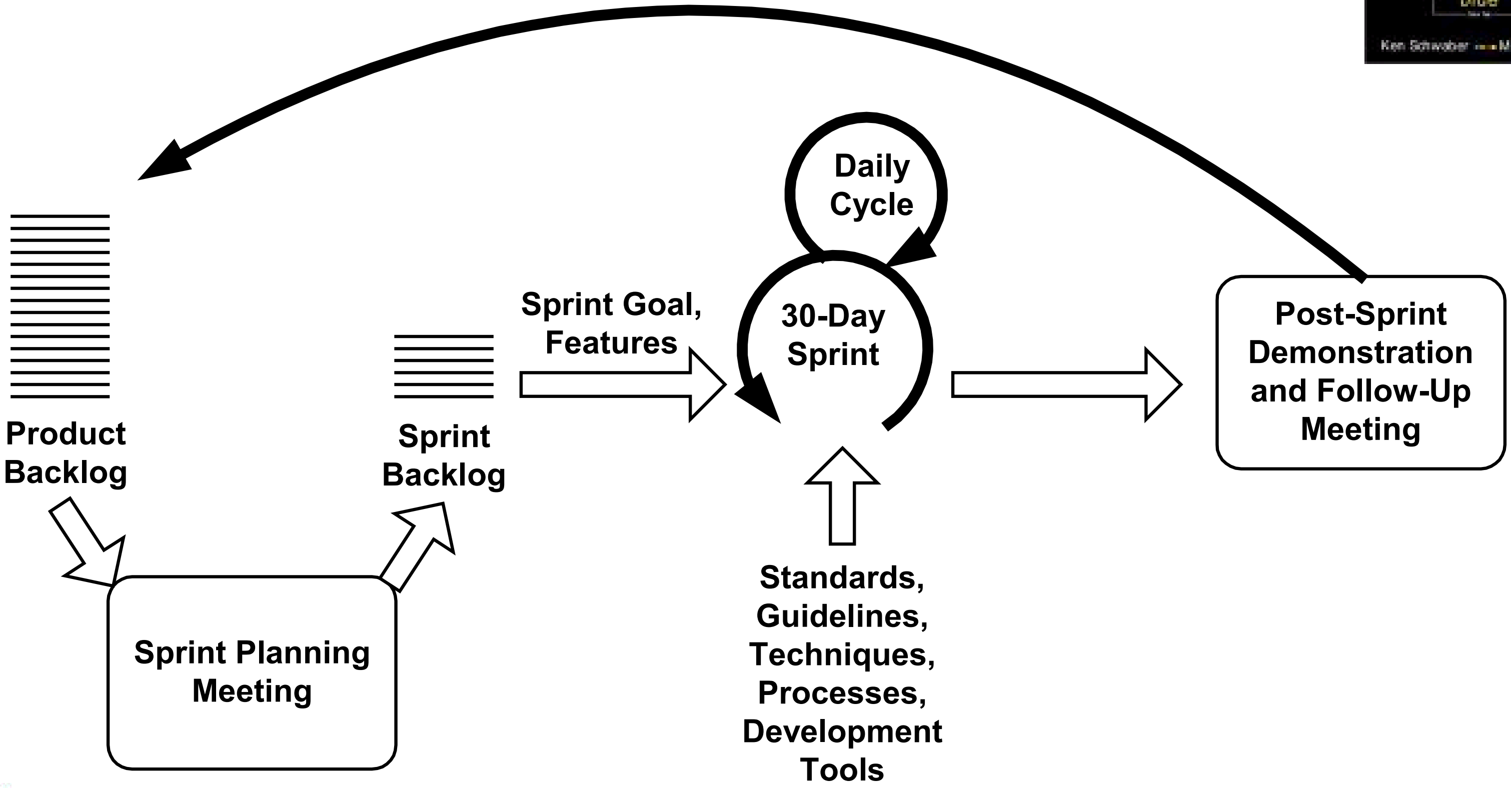

agile modeling

Effective Practices for eXtreme Programming and the Unified Process

Scott Ambler

Scrum
www.controlchaos.com





Daily Cycle

Product Backlog

Sprint Backlog

Sprint Goal, Features

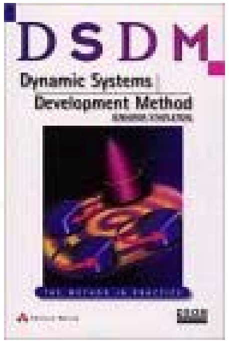
30-Day Sprint

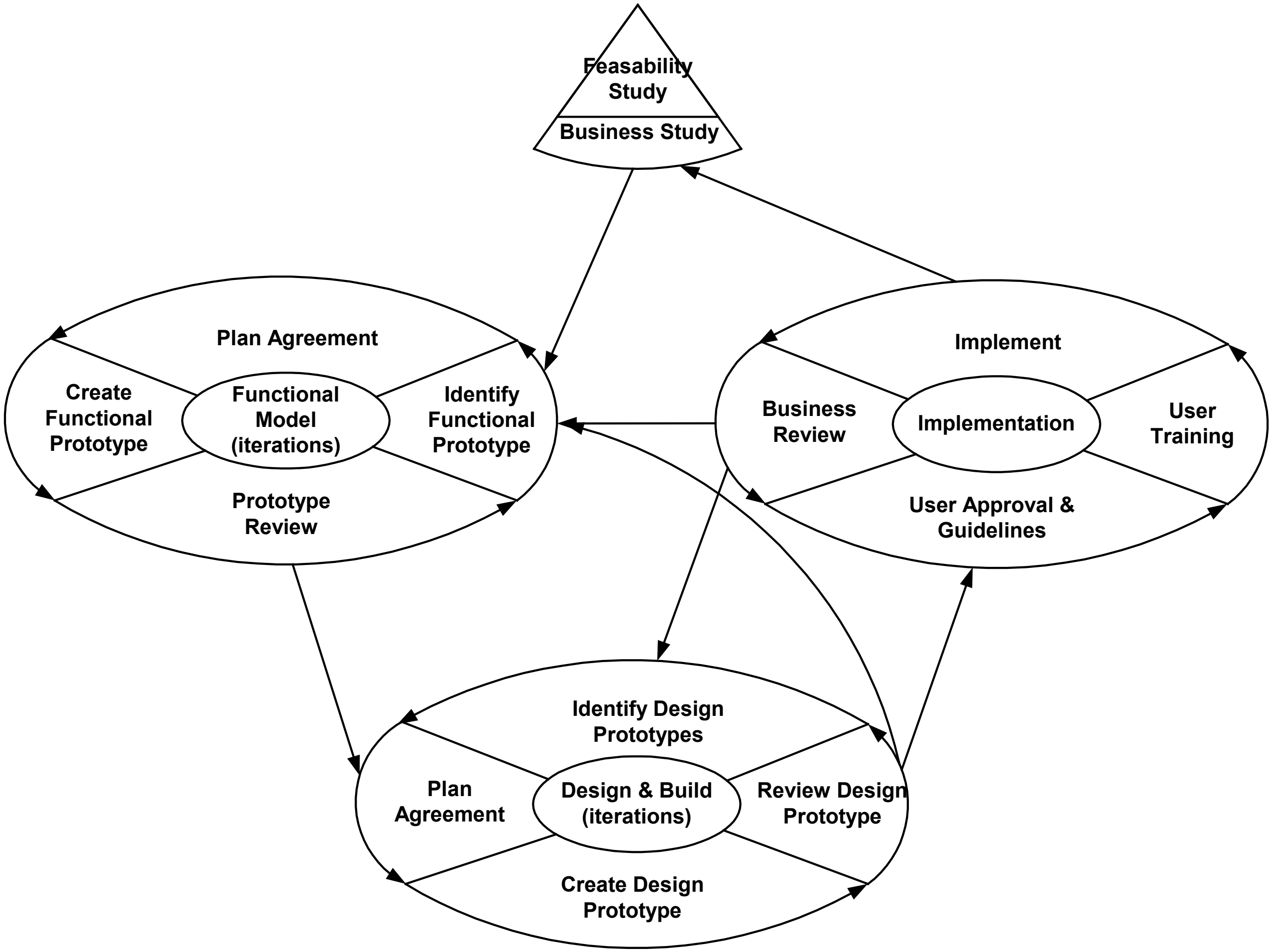
Post-Sprint Demonstration and Follow-Up Meeting

Sprint Planning Meeting

Standards, Guidelines, Techniques, Processes, Development Tools

Dynamic System Development Method (DSDM) www.dsdm.org


DSDM
Dynamic Systems
Development Method


Feasability Study
Business Study

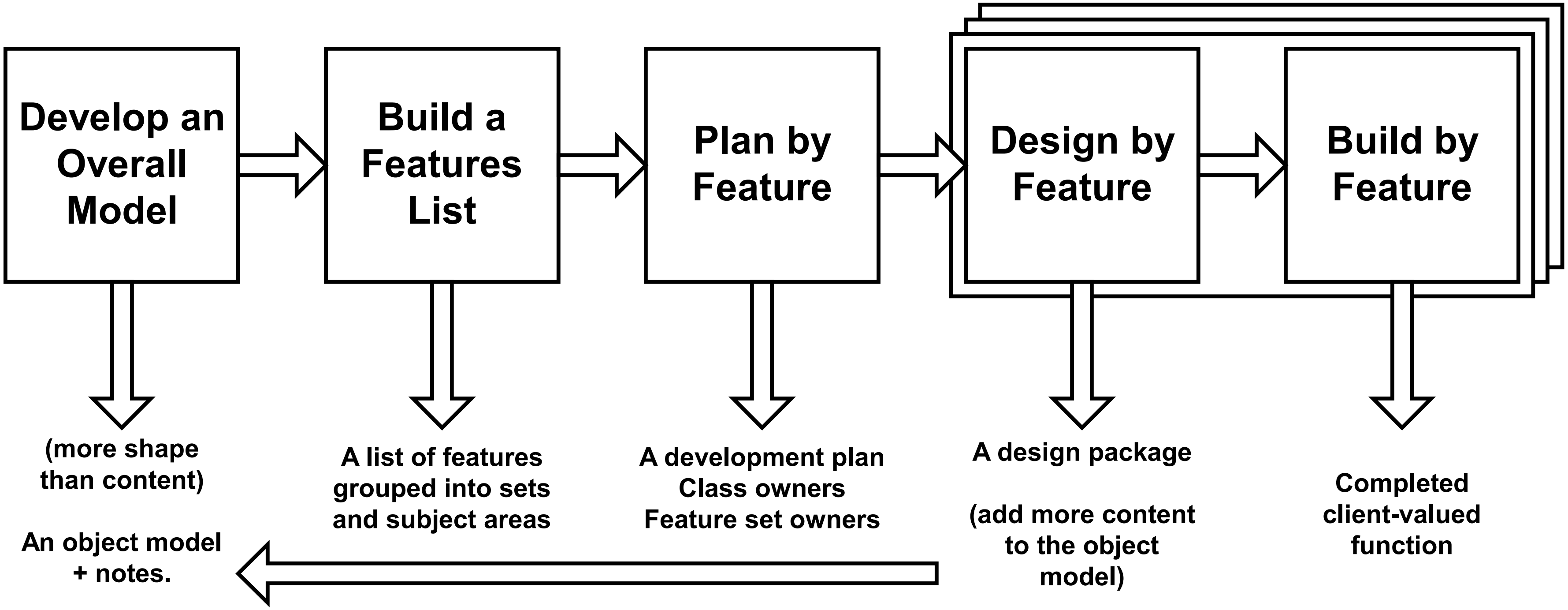
Plan Agreement
Create Functional Prototype
Functional Model (iterations)
Identify Functional Prototype
Prototype Review

Implement
Business Review
Implementation
User Training
User Approval & Guidelines

Identify Design Prototypes
Plan Agreement
Design & Build (iterations)
Review Design Prototype
Create Design Prototype

Feature Driven Development (FDD) www.thecoadletter.com

A Practical Guide to FEATURE-DRIVEN DEVELOPMENT



Develop an Overall Model

Build a Features List

Plan by Feature

Design by Feature

Build by Feature

(more shape than content)

An object model + notes.

A list of features grouped into sets and subject areas

A development plan
Class owners
Feature set owners

A design package

(add more content to the object model)

Completed client-valued function

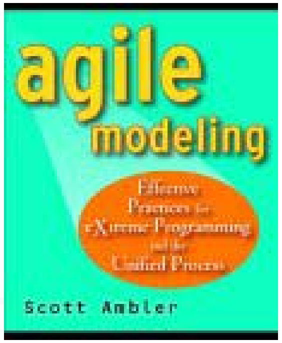
Crystal Clear crystalmethodologies.org

- Put your team in a room.
- Get out of their way.
- Policies the team must follow:
  - Incremental delivery every two or three months
  - Some automated testing
  - Direct user involvement
  - Two user reviews per release
  - Methodology tuning workshops

“Agile RUP”

- Theoretically possible, and some people are doing it
- RUP appeals to people with a prescriptive mindset
- The agile movement caught Rational by surprise
- Everyone doesn’t have to be agile
- If you want an agile method, consider something else
- www.agilemodeling.com/essays/agileModelingRUP.htm

Agile Modeling (AM)
www.agilemodeling.com


agile
modeling
Effective Practices for eXtreme Programming and the Unified Process
Scott Ambler

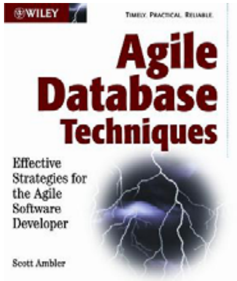
Core Principles

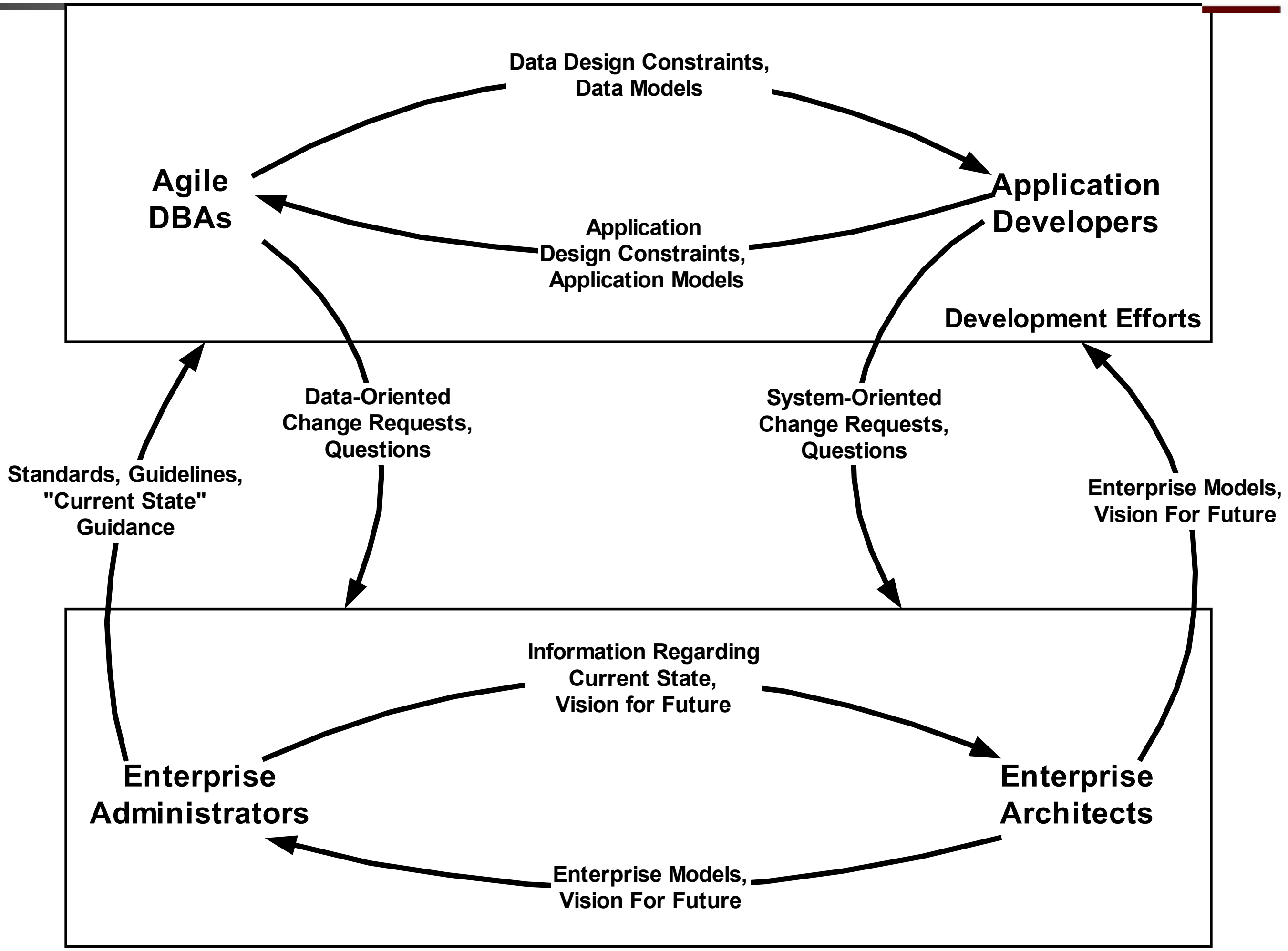
- Assume Simplicity
- Embrace Change
- Enabling the Next Effort is Your Secondary Goal
- Incremental Change
- Model With a Purpose
- Multiple Models
- Maximize Stakeholder Investment
- Quality Work
- Rapid Feedback
- Software Is Your Primary Goal
- Travel Light

Core Practices

- Active Stakeholder Participation
- Apply the Right Artifact(s)
- Collective Ownership
- Consider Testability
- Create Several Models in Parallel
- Create Simple Content
- Depict Models Simply
- Display Models Publicly
- Iterate to Another Artifact
- Model in Small Increments
- Model With Others
- Prove it With Code
- Use the Simplest Tools

Agile Data (AD)
www.agiledata.org


WILEY
Timely. Practical. Reliable.
Agile Database Techniques
Effective Strategies for the Agile Software Developer
Scott Ambler



Agile DBAs

Application Developers

Development Efforts

Data Design Constraints,
Data Models

Application
Design Constraints,
Application Models

Data-Oriented
Change Requests,
Questions

System-Oriented
Change Requests,
Questions

Standards, Guidelines,
"Current State"
Guidance

Enterprise Models,
Vision For Future

Enterprise Administrators

Enterprise Architects

Information Regarding
Current State,
Vision for Future

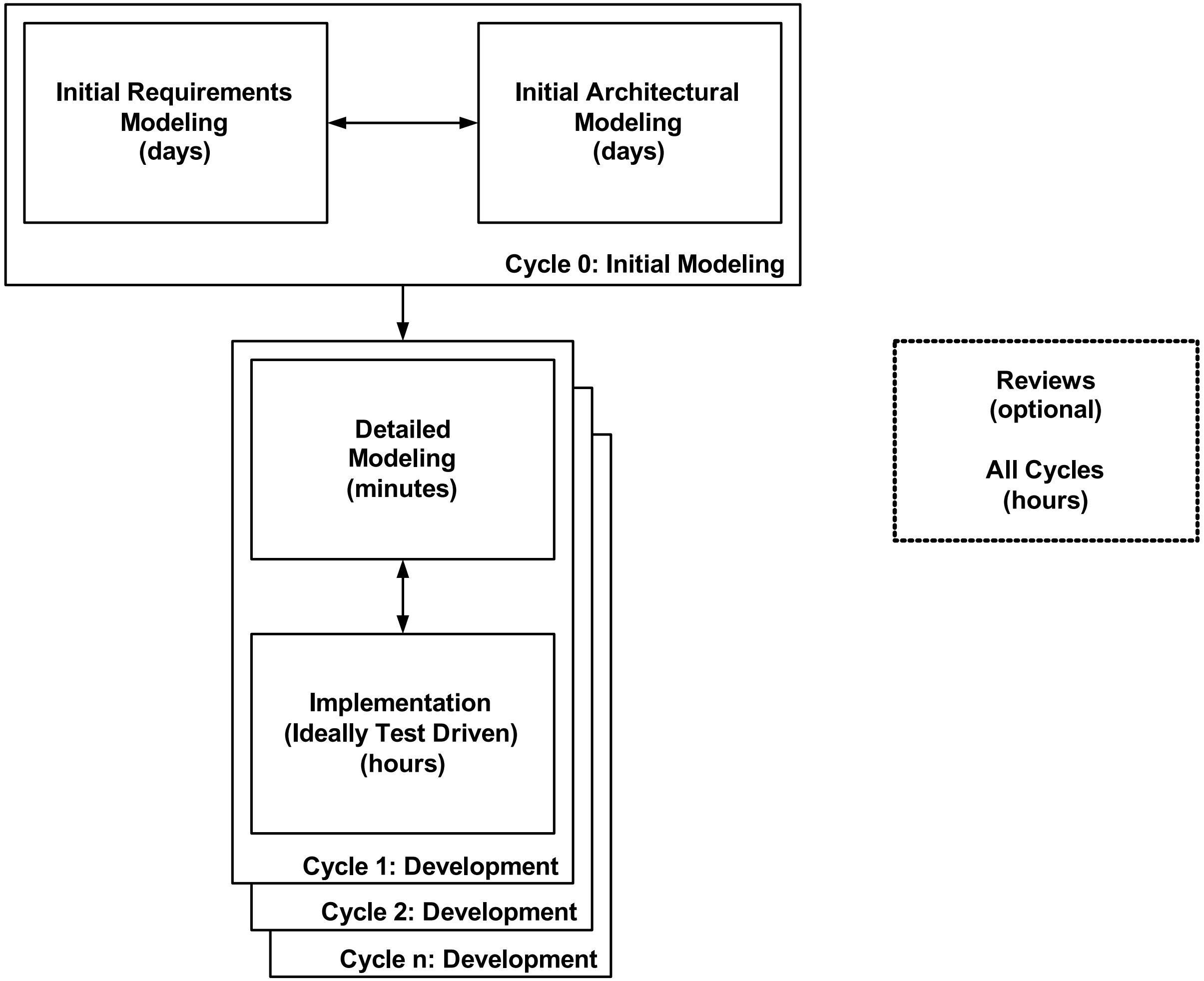
Enterprise Models,
Vision For Future

Leading Agile Software Techniques

- Agile Model Driven Development (AMDD)
- Database Refactoring
- Test Driven Development (TDD)

Agile Model Driven Development (AMDD)

www.agilemodeling.com/essays/amdd.htm

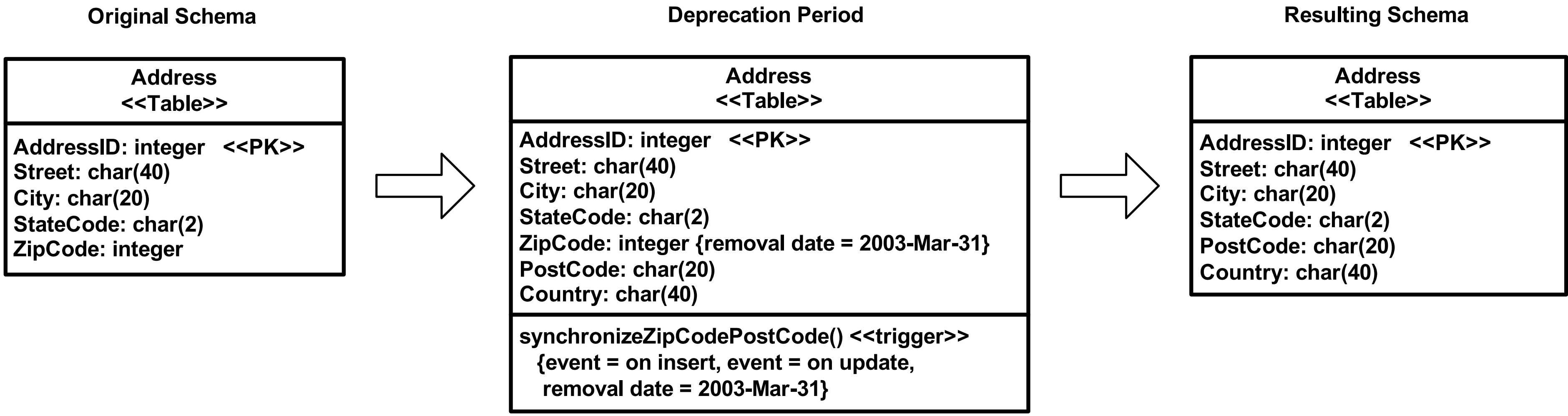

Initial Requirements Modeling (days)
Initial Architectural Modeling (days)
Cycle 0: Initial Modeling
Detailed Modeling (minutes)
Implementation (Ideally Test Driven) (hours)
Cycle 1: Development
Cycle 2: Development
Cycle n: Development
Reviews (optional)
All Cycles (hours)

Database Refactoring

www.agiledata.org/essays/databaseRefactoring.html

- A database refactoring is a simple change to a database schema that improves its design while retaining both its *behavioral and informational semantics*.
- A database schema includes both structural aspects such as table and view definitions as well as functional aspects such as stored procedures and triggers.
- In many ways database refactoring is simply normalization after the fact.
- Database refactorings are a subset of schema transformations, but they do not add functionality.

Database Refactoring Example
Replace Column



Original Schema

Address
<<Table>>

AddressID: integer <<PK>>
Street: char(40)
City: char(20)
StateCode: char(2)
ZipCode: integer

Deprecation Period

Address
<<Table>>

AddressID: integer <<PK>>
Street: char(40)
City: char(20)
StateCode: char(2)
ZipCode: integer {removal date = 2003-Mar-31}
PostCode: char(20)
Country: char(40)

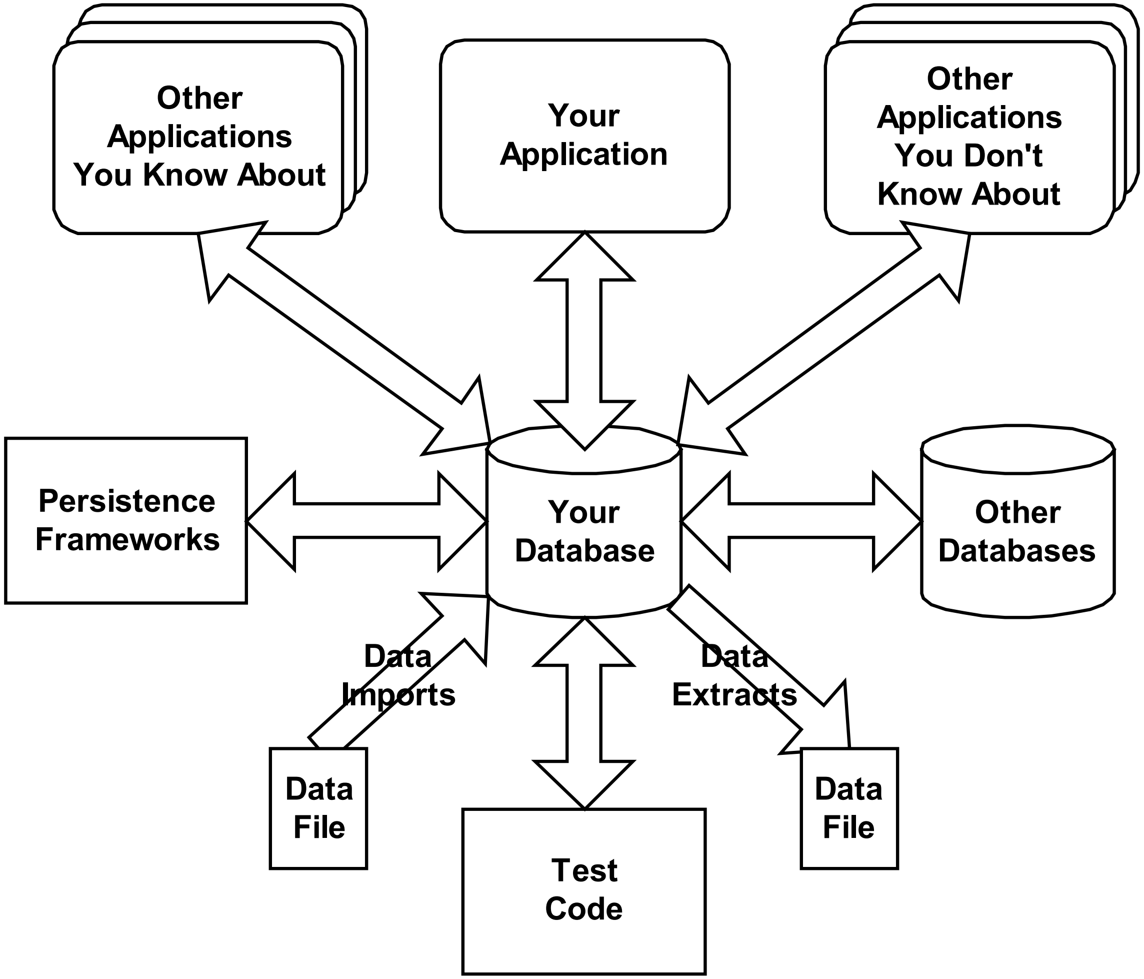
synchronizeZipCodePostCode() <<trigger>>
{event = on insert, event = on update,
removal date = 2003-Mar-31}

Resulting Schema

Address
<<Table>>

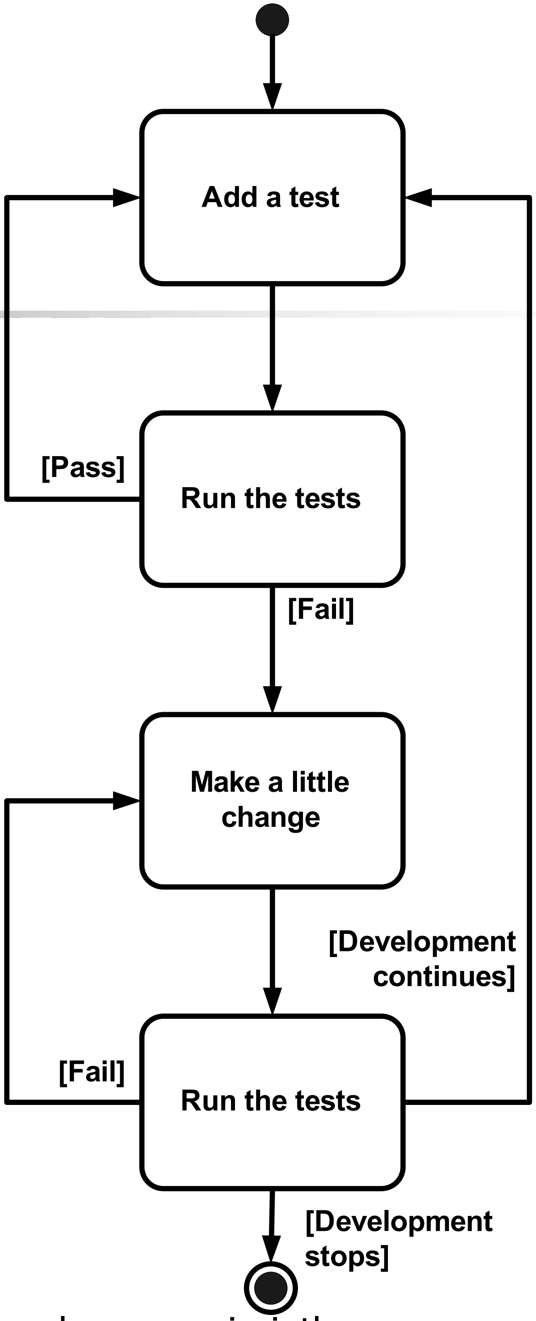
AddressID: integer <<PK>>
Street: char(40)
City: char(20)
StateCode: char(2)
PostCode: char(20)
Country: char(40)

Why DB Refactoring is Hard



Test Driven Development (TDD)

www.agiledata.org/essays/tdd.html



Interesting Observations

- The people involved with the Agile Alliance build software for a living – they are not academics
- Most members are already well known within the community – they're not simply doing this to become famous
- The alliance is made up of a diverse range of people, including competitors – yet they agreed on fundamental principles
- Agile software development is real
- Agile software development is not a fad
- Agile software development is supported by a wide range of industry luminaries
- Research evidence support agile techniques is beginning to emerge

Questions?

Scott W. Ambler

scott.ambler@ronin-intl.com

www.ronin-intl.com/company/scottAmbler.html

Important URLs

- www.agilealliance.org
- www.agilemodeling.com
- www.agiledata.org
- www.extremeprogramming.com
- www.xprogramming.com
- www.ambysoft.com/processPatterns.html
- www.modelingstyle.info
- www.enterpriseunifiedprocess.info

References and Recommended Reading

- Ambler, S.W. (1998a). *Building Object Applications That Work: Your Step-By-Step Handbook for Developing Robust Systems with Object Technology*. New York: Cambridge University Press.
- Ambler, S. W. (1998b). *Process Patterns – Building Large-Scale Systems Using Object Technology*. New York: Cambridge University Press.
- Ambler, S. W. (1999). *More Process Patterns – Delivering Large-Scale Systems Using Object Technology*. New York: Cambridge University Press.
- Ambler, S.W. (2002a). *Agile Modeling: Effective Practices for XP and the UP*. New York: John Wiley & Sons. www.ambysoft.com/agileModeling.html
- Ambler, S.W. (2002b). *The Elements of UML Style*. New York: Cambridge University Press.
- Ambler, S.W. (2004a). *Agile Database Techniques: Effective Strategies for the Agile Software Developer*. New York: John Wiley & Sons.
- Ambler, S.W. (2004b). *The Object Primer 3rd Edition: Agile Model Driven Development (AMDD) with UML 2*. New York: Cambridge University Press. www.ambysoft.com/theObjectPrimer.html

References and Recommended Reading

- Ambler, S.W. & Constantine L.L. (2000a). *The Unified Process Elaboration Phase*. Gilroy, CA: CMP Books.
- Ambler, S.W. & Constantine L.L. (2000b). *The Unified Process Construction Phase*. Gilroy, CA: CMP Books.
- Ambler, S.W. & Constantine L.L. (2001). *The Unified Process Inception Phase*. Gilroy, CA: CMP Books.
- Ambler, S.W. & Constantine L.L. (2002). *The Unified Process Transition and Production Phases*. Gilroy, CA: CMP Books.
- Beck, K. (2000). *Extreme Programming Explained – Embrace Change*. Reading, MA: Addison Wesley Longman, Inc.
- Beck, K. & Fowler, M. (2001). *Planning Extreme Programming*. Reading, MA: Addison Wesley Longman, Inc.
- Cockburn, A. (2002). *Agile Software Development*. Boston: Addison Wesley.
- Constantine, L.L. & Lockwood, L.A.D. (1999). *Software For Use: A Practical Guide to the Models and Methods of Usage-Centered Design*. New York: ACM Press.
- Evans, G. (2001). *Palm Sized Process: Point of Sale Gets Agile*. Software Development, September 2001. www.sdmagazine.com
- Highsmith, Jim (2002). *Agile Software Development Ecosystems*. Boston: Addison Wesley.

References and Recommended Reading

- Jefferies, R., Anderson, A., & Hendrickson, C. (2001). *Extreme Programming Installed*. Reading, MA: Addison Wesley Longman, Inc.
- Newkirk J. & Martin, R.C. (2001). *Extreme Programming in Practice*. Boston: Addison Wesley.
- Palmer, S.R. & Felsing, J.M. (2002). *A Practical Guide to Feature Driven Development*. Upper Saddle River, NJ: Prentice Hall PTR.
- Rational Corporation (2001). *Rational Unified Process Home Page*. http://www.rational.com/products/rup/index.jsp
- Roman, E., Ambler, S.W., & Jewell, T., (2002). *Mastering Enterprise Java Beans, 2/e*. New York: John Wiley & Sons.
- Stapleton, J. (2003). *Dynamic Systems Development Method*. Harlow, England: Addison Wesley.
- Vermeulen, A., Ambler, S.W., Bumgardner, G., Metz, E., Misfeldt, T., Shur, J., & Thompson, P. (2000). *The Elements of Java Style*. New York: Cambridge University Press.
- Wake, W.C. (2002). *Extreme Programming Explored*. Boston, MA: Addison Wesley.
- Wells, J.D. (2001). *Extreme Programming: A Gentle Introduction*. http://www.extremeprogramming.org