

Are You Agile or Fragile?

An Agile Software Development Discussion

Michael J. Vizdos

Professional Services, Ronin International

michael.vizdos@ronin-intl.com

www.agilemodeling.com

www.agiledata.org





Questions?

Please ask any questions

The only dumb question
is the one that you don't ask





Discussion Goals

- Introduce you to agile software development values, principles, and practices
- To actively challenge your beliefs of how software development works
- To introduce you to leading agile development processes
- To get you thinking outside of the box





Examining the Basics of Software Development

What is a Software Process?

The Importance of Communication

Agile Software Development



What Is Software Process?

- The purpose of a software development process is to enable and enforce the repeatable delivery of working software in a timely and cost effective manner, supplying accurate and meaningful information to all key roles inside and outside a project with the minimum of disruption to developers and project stakeholders.
 - Modified from pg. 17 of *A Practical Guide to Feature Driven Development*





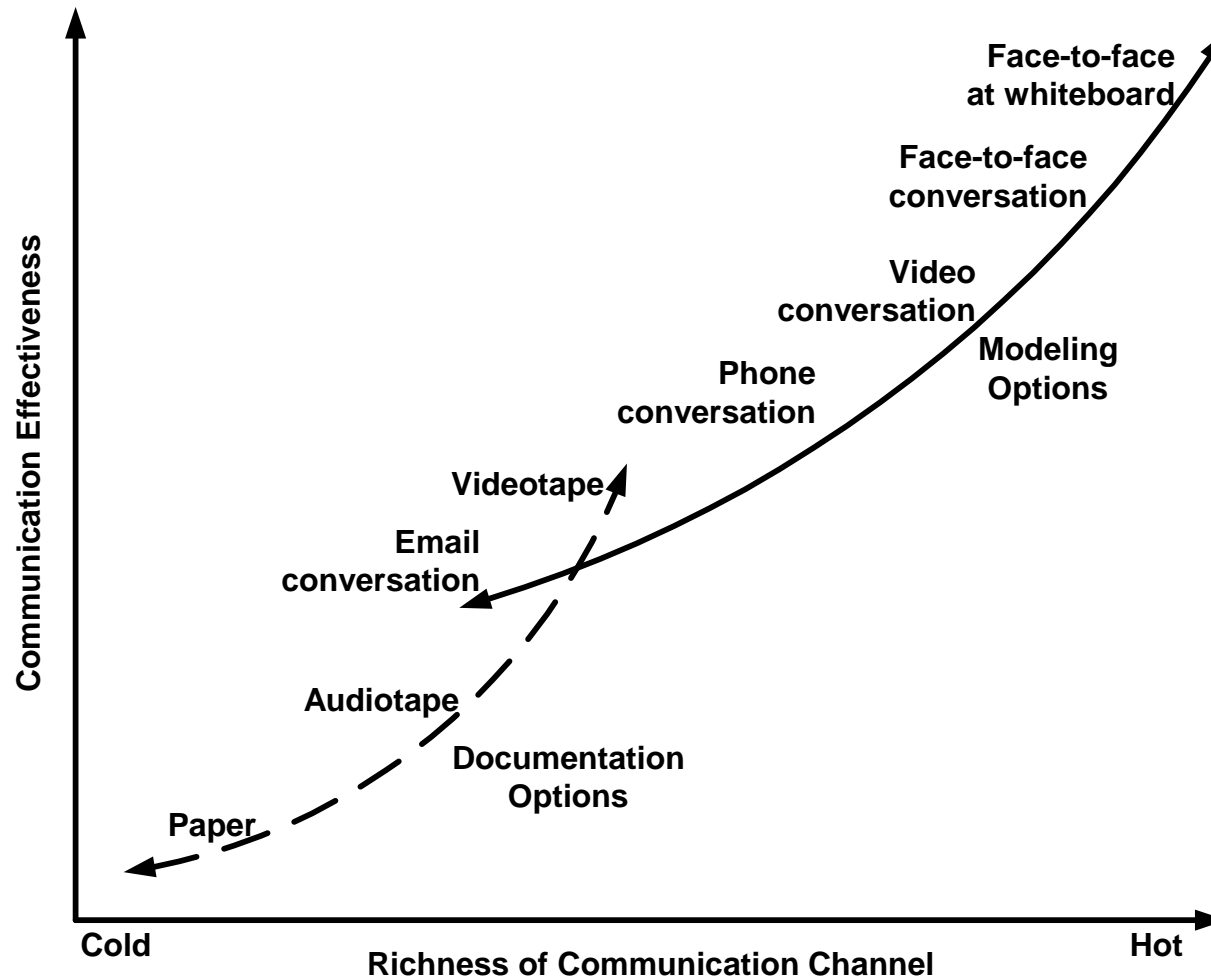
Communication

- Effective communication is a critical success factor for software development
- Inhibitors:
 - Fear of being wrong
 - Fear of revealing your own flaws
- Challenges:
 - Each time we translate from one language to another we risk information loss



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.





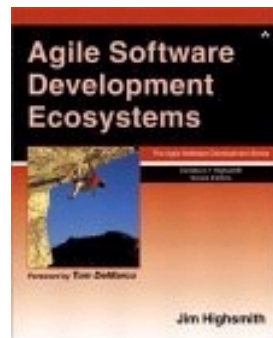
Agile Software Development is not

- “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





Suggested Reading





Quotation

- *Do these few things well, and projects will succeed. If you don't do these few things well, it doesn't matter how many other hundreds of things you do well, you won't succeed.*
 - Ken Schwaber and Mike Beedle





The Agile Alliance

A Bit of History
Agile Manifesto
Agile Principles
Interesting Observations
Quick Discussion



A Bit of History

- 17 methodologists met in Snowbird Utah in March 2001
- Discussed fundamental values and principles of effective software development
- Formed the Agile Alliance, a non-profit organization
- www.agilealliance.org





The Agile Manifesto

- Value:
 - *Individuals and interactions* over processes and tools
 - *Working software* over comprehensive documentation
 - *Customer collaboration* over contract negotiation
 - *Responding to change* over 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.





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





Small Group Discussion: Agile Software Development (ASD)

- Suggested discussion points:
 - Is the concept of ASD realistic? Why (not)?
 - What aspects of ASD are you currently doing? What aspects aren't you doing? Why?





Quotation

- *Sorry about your cow, I didn't know she was sacred.*
 - Ron Jeffries



A decorative graphic consisting of a vertical black line and a horizontal black line intersecting at a point. To the left of the intersection, there are several overlapping squares in blue, red, and yellow, with a gradient effect.

Agile Modeling (AM)

What is AM?

The History of AM

The Values, Principles, and Practices of AM

Is this Professional? (JIT)

AM and Other Processes

When Will AM Work For You?

When Won't AM Work For You?



Overview of AM

- AM is a chaotic, 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





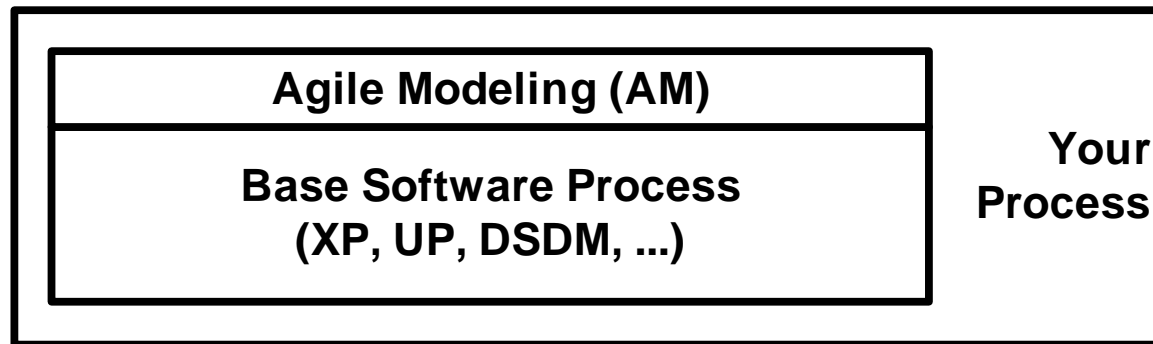
History of AM

- “eXtreme Modeling” published in Autumn of 2000 in Software Development
- Fantastic feedback right from the start
- www.extrememodeling.com created Jan 2001
- Topica.com discussion group created
- Evolved into Agile Modeling and www.agilemodeling.com in April 2001 to reflect actual scope of the methodology
- Principles and practices finalized at XP2001 in May 2001
- Materials posted on web site and updated based on feedback throughout Spring, Summer, and Autumn of 2001
- *Agile Modeling* book published in March 2002





Scope of AM





What Are Agile Models?

- Agile models:
 - Fulfill their purpose
 - Are understandable
 - Are sufficiently accurate
 - Are sufficiently consistent
 - Are sufficiently detailed
 - Provide positive value
 - Are as simple as possible
- Agile models are just barely enough!





Values of AM

- Communication
- Simplicity
- Feedback
- Courage
- Humility





Principles of AM

- 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
- Content is More Important Than Representation
- Everyone Can Learn From Everyone Else
- Know Your Models
- Know Your Tools
- Local Adaptation
- Open and Honest Communication
- Work With People's Instincts





Practices of AM

- 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
- Apply Modeling Standards
- Apply Patterns Gently
- Discard Temporary Models
- Formalize Contract Models
- Model to Communicate
- Model to Understand
- Reuse Existing Artifacts
- Update Only When It Hurts





AM and Other Agile Processes

- Modeling is an important aspect of all agile software processes, including XP
- The principles and practices of AM reflect the values and principles of the Agile Alliance
- AM has been successfully used on XP and RUP projects
- An FDD project team is currently “rumored” to be applying AM practices successfully





When Will AM Work For You?

- You are taking an agile approach to development
- You are working iteratively and incrementally
- Uncertain or volatile requirements
- Your primary goal is to develop software
- You have active stakeholder support and involvement
- The development team is in control of its destiny
- A true champion for AM exists
- You have responsible and motivated developers
- You have adequate resources available to you





When Won't AM Work for You?

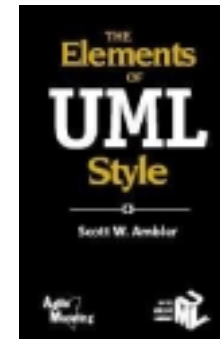
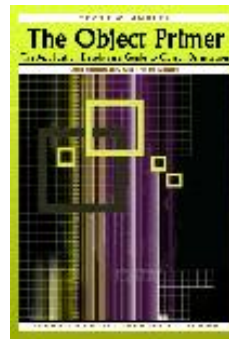
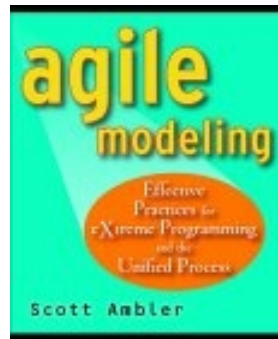
- One or more of the previous factors isn't there
- Your organizational culture is geared towards a prescriptive culture
- You have a large and/or distributed team





Suggested Reading

- www.agilemodeling.com
- www.agiledata.org
- www.modelingstyle.info





Small Group Discussion: Similarities and differences

- Discuss the **following**:
 - What common philosophies appear throughout agile software processes?
 - How different is this from your existing philosophies?
 - What challenges exist for your organization to adopt these philosophies?





Quotation

- *The U.S. army has a concept that it calls “on the ground.” That is, to really learn something you must be there where the action is – on the ground.*
 - Jim Highsmith



Adopting Agile Software Development Processes





Wouldn't You Like To...

- Actually look forward to going to work every day?
- Work on successful projects?
- Focus on activities that really matter?
- Develop software that meets the actual needs of your project stakeholders?
- Invest the resources of your project stakeholders wisely?
- Have your project stakeholders honestly extol the virtues of your work?
- Be able to proudly say "I built that"?



Questions?

Michael J. Vizdos

michael.vizdos@ronin-intl.com

www.agilemodeling.com

www.modelingstyle.info

www.agiledata.org





Extra Discussion / References



Leading Agile Software Processes



Extreme Programming (XP)

Scrum

Dynamic System Development Method (DSDM)

Feature Driven Development (FDD)

Crystal Clear

“Agile RUP”

Agile Modeling (AM)



Common Adoption Issues

- Stakeholder support
- Developer support
- Cultural fit
- Process champion
- Opportunity & resources to succeed





Extreme Programming (XP)

Overview

Suggested Reading



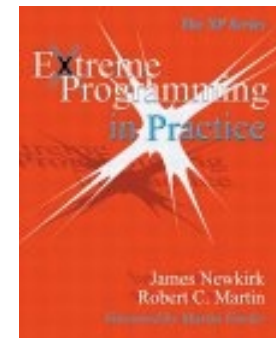
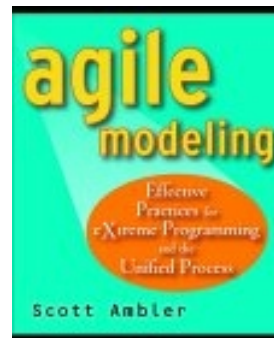
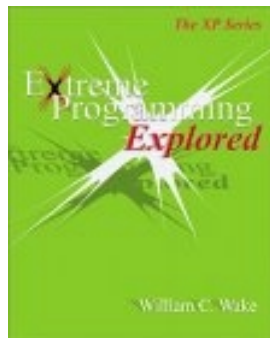
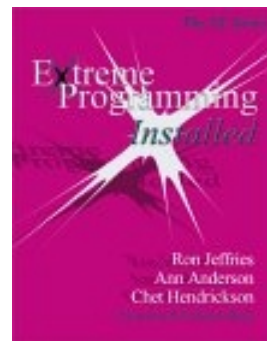
Overview of XP

- XP is a light-weight methodology for small-to-medium-sized teams developing software in the face of vague or rapidly changing requirements.
- XP is designed to work with projects that:
 - Can be built by teams of two to ten programmers
 - That aren't sharply constrained by the computing environment
 - Tests can be reasonably executed within a fraction of a day
- XP promises:
 - To developers that they will be able to work on things that really matter, every day
 - To project stakeholders that they will get the most possible value out of every programming week



Suggested Reading

- www.xprogramming.com
- www.extremeprogramming.org





Quotation

- *A decade ago, one of us wrote a 110-page process for a large development team. No matter how hard he tried to defend every word of his process as something of great value, the team members ALWAYS looked at the 4-page summary in the back and ignored the rest.*
 - Peter Coad, Eric Lefebvre, and Jeff DeLuca





Scrum

Overview

Suggested Reading



Overview of Scrum

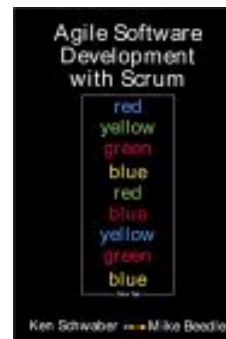
- Scrum starts with the premise that we live in a complicated world, that you can't predict or definitively plan what you will deliver, when you will deliver it, or what the quality and cost will be.
- Assumes that people do the best that they can under the circumstances, therefore management emphasis should be on improving the circumstances, monitoring the features to be delivered, and constantly adjusting.
- Scrum relies on self-commitment, self-organization, and emergence rather than authoritarian measures.
- Scrum teams are measured on meeting goals, not on how many hours it took to meet that goal.
- Works on a wide range of projects, from small to large teams, from co-located to distributed, in various domains.





Suggested Reading

- www.controlchaos.com





Quotation

- *Think of the project manager as operating the deflector shield on the front of Star Trek's U.S.S. Enterprise so that the project team behind can go about it's work, safe from random elements floating in its direction.*
 - Stephen Palmer and John Felsing





DSDM

Overview
Suggested Reading



Overview of DSDM

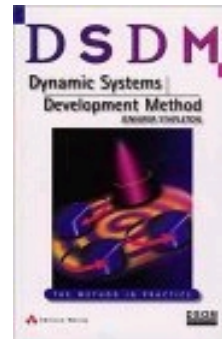
- Originally called Dynamic Systems Development Method, now considered by some to be Dynamic Solutions Delivery Model
- Formalization of Rapid Application Development (RAD)
- Works very well for small to medium sized projects with a large user interface aspect to them
- Emphasizes the use of facilitated workshops with project stakeholders
- Focused on business benefits and avoiding redundant work
- Recognizes that nothing is built right the first time





Suggested Reading

- www.dsdm.org





Quotation

- *The quality of the people on a project, and their organization and management, are much more important factors in success than are the tools they use or the technical approaches they take.*
 - Fred Brooks



Feature Driven Development (FDD)



Overview

Suggested Reading



Overview of FDD

- FDD is a client/customer-centric, architecture-centric, and pragmatic.
- It contains just enough process to ensure scalability and repeatability and encourage creativity and innovation all along the way.
- A feature is a small, client-valued function expressed in the form <action><result><object>
 - Calculate the total of a sale
 - Validate the password of a user
 - Authorize the sales transaction of a customer





Suggested Reading

- www.thecoadletter.com





Quotation

- *I see software development as primarily a sociological or cultural phenomenon.*
 - Luke Hohmann





Crystal Clear

Overview

Suggested Reading



Overview of Crystal Clear

- Domain is up to six people, co-located in a room, on a non-life critical project.
- One of a family of software processes – Clear, Yellow, Orange, Red.
- Crystal Clear is minimalist.
- Two absolute rules of the Crystal Family:
 - Use of incremental cycles not to exceed four months.
 - Use of reflection workshops (retrospectives) so the methodology may be self adapting.





Suggested Reading

- crystalmethodologies.org





Quotation

- *We think most process initiatives are silly. Well-intentioned managers and teams get so wrapped up in executing processes that they forget that they are being paid for results, not process execution.*
- Peter Coad, Eric Lefebvre, and Jeff De Luca





The Unified Process (UP)

Overview

History

The RUP Lifecycle

Can the UP be Agile?

The EUP Lifecycle

Should You Adopt the UP?

Suggested Reading



Overview of the UP

- The Unified Process (UP) is a framework from which software processes are instantiated.
- The Rational Unified Process (RUP) is an instantiation of the UP. It defines the roles that people take on a software engineering project, the responsibilities of those roles, the activities that people in them perform.
- The Enterprise Unified Process (EUP) is an extension to the RUP.



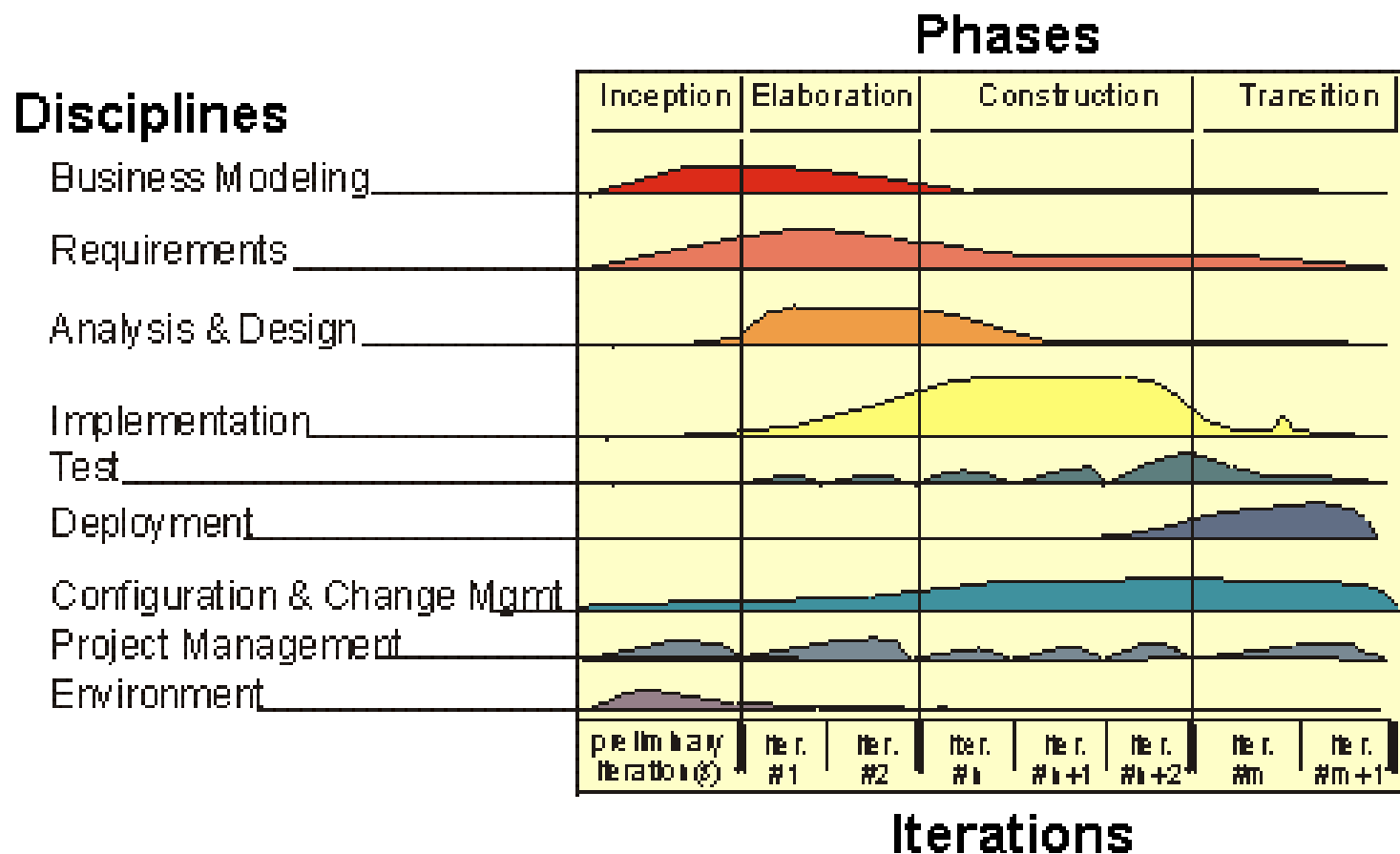


History of the UP

- Foundations in Jacobson's Objectory process
- Processes supported by the various tool vendors that Rational Corporation purchased were used to flesh out Objectory
- Name "Unified Process" coined to foster perceived link to the UML
- Unified Process book written by Three Amigos published in the Spring of 1999
- Kruchten's book soon followed
- RUP product first introduced in 1999, popularized in 2000
- Unified Process series from CMP books followed in 2000 through 2000 to define the Enterprise Unified Process (EUP)
- "Extreme RUP" article by Gary Evans published in Software Development, Autumn 2001
- Agile RUP, at least modeling disciplines, described by Craig Larman in *Applying UML and Patterns 2/e* in late 2001



The Rational Unified Process (RUP) Lifecycle



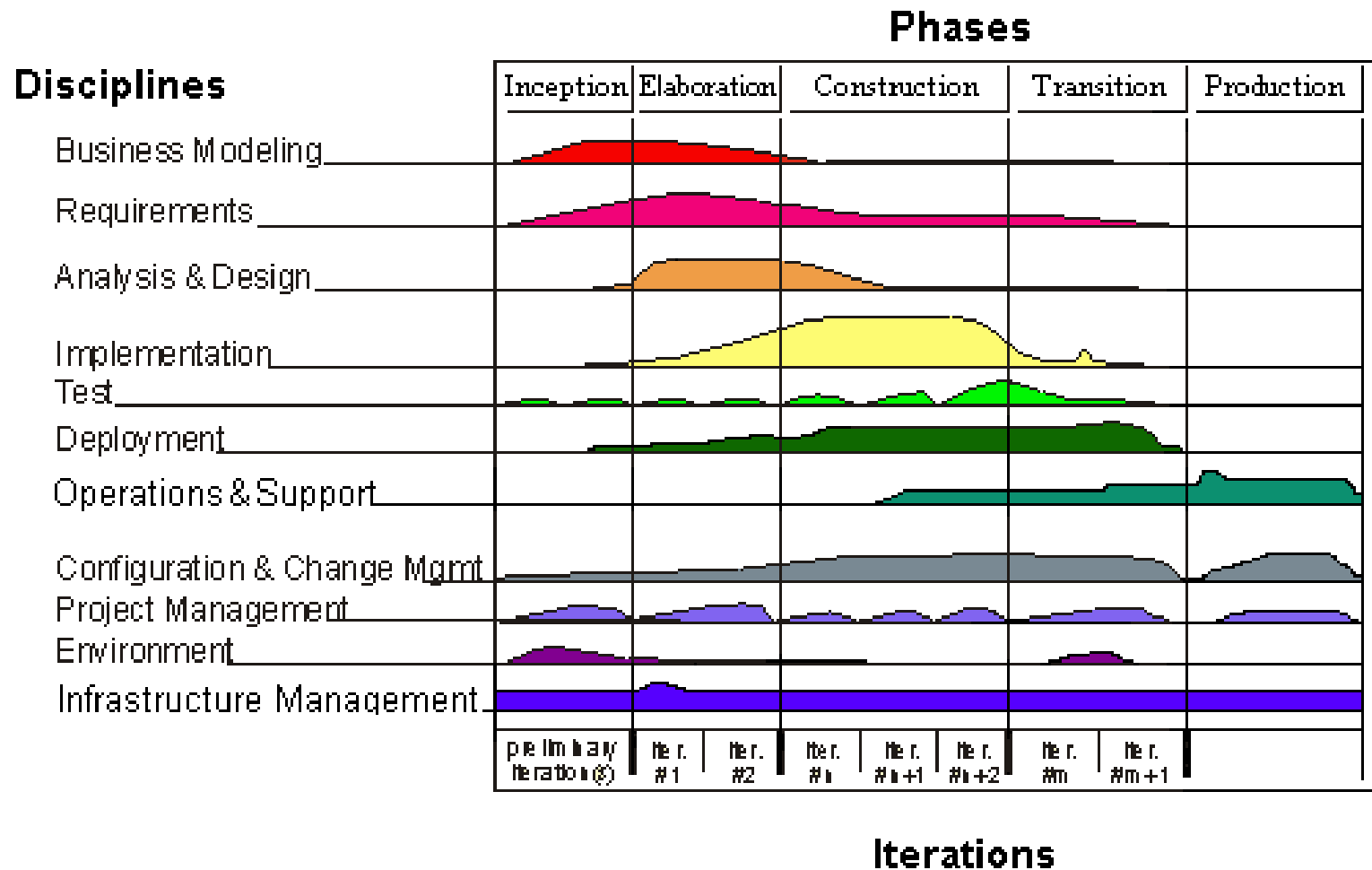
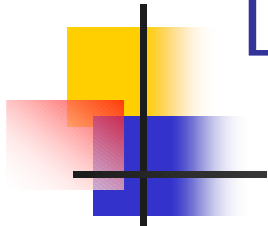


Can the RUP be Agile?

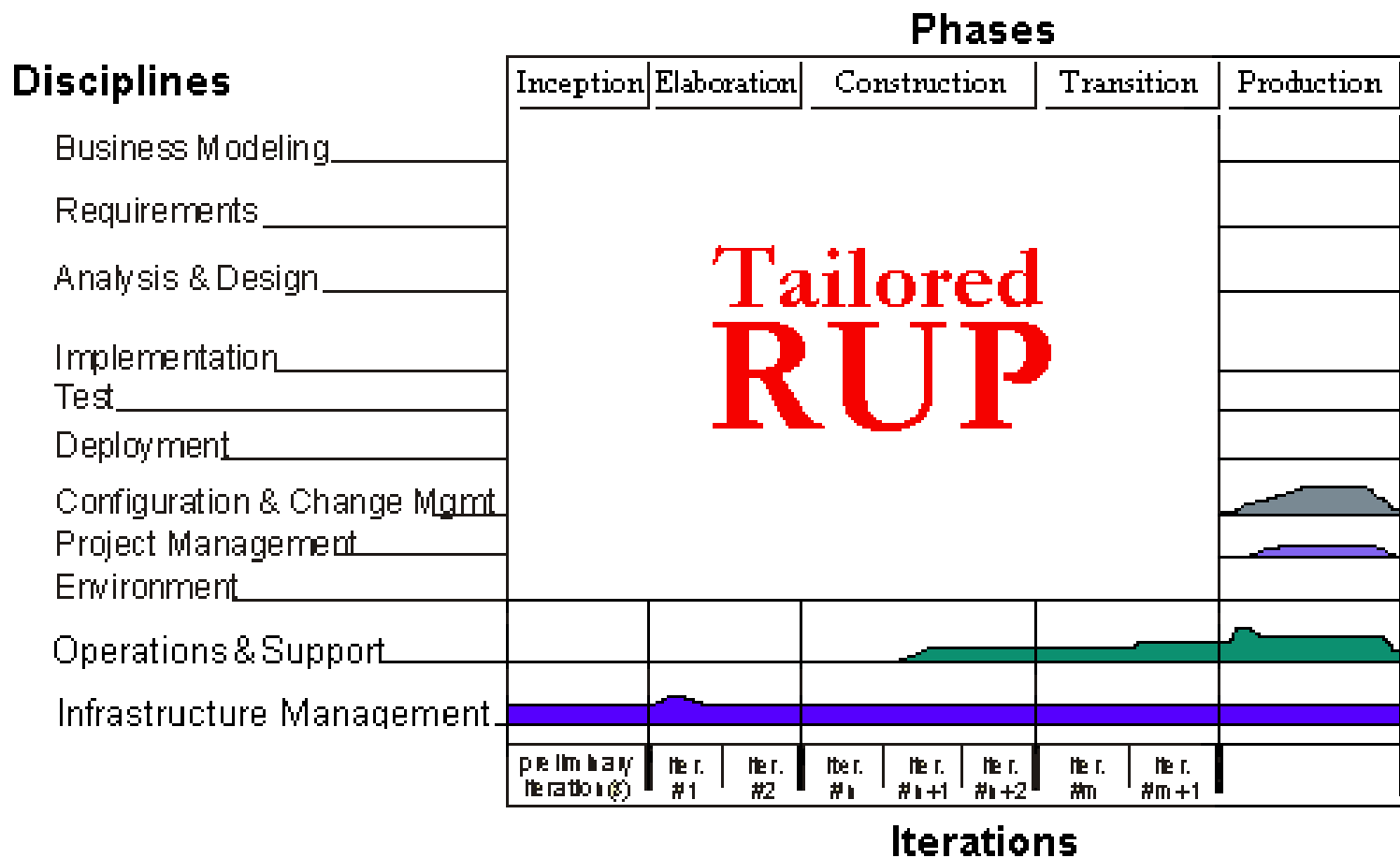
- Yes, but...
- The organizational cultural to which RUP is attractive often doesn't find the principles of agile development attractive
- There are many agile software development processes to choose from, why invest the effort to tailor the RUP?



The Enterprise Unified Process (EUP) Lifecycle



The EUP Wraps the RUP





Should You Adopt UP?

Strengths

- Process oriented
- Well defined
- Significant material exists
- Can be tailored to be agile
- Attractive to management
- Significant market mind share
- Proven to work in practice

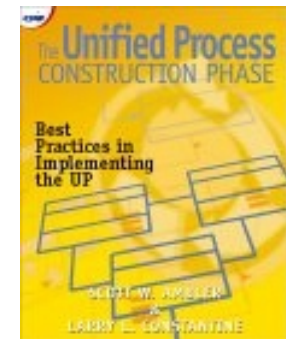
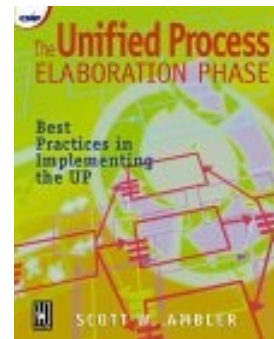
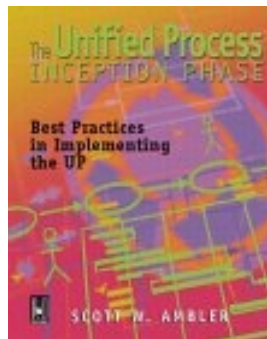
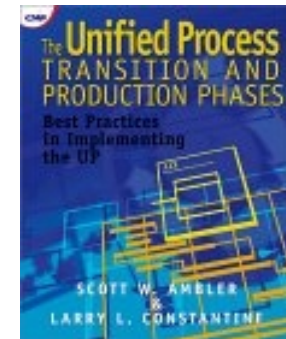
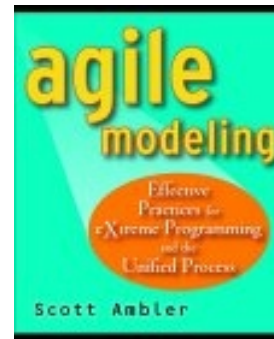
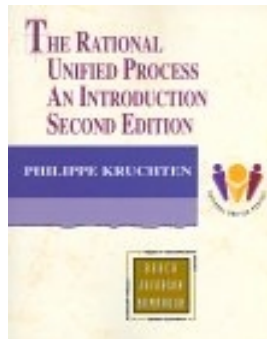
Weaknesses

- Process oriented
- Attractive to bureaucrats
- Dominated by a tool vendor
- Typically instantiated as a rigorous process
- Developers do not seem to like it



Suggested Reading

- www.ronin-intl.com/publications/unifiedProcess.html
- www.rational.com/products/rup/index.jsp



Questions?

Michael J. Vizdos

michael.vizdos@ronin-intl.com

www.agilemodeling.com

www.modelingstyle.info

www.agiledata.org





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. (2001). *The Object Primer 2nd Edition*. New York: Cambridge University Press. www.ambysoft.com/theObjectPrimer.html
- 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. (2003). *Agile Data*. New York: John Wiley & Sons.



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.



References and Recommended Reading

- Evans, G. (2001). *Palm Sized Process: Point of Sale Gets Agile*. Software Development, September 2001. www.sdmagazine.com
- Fowler, M. (1997). *Analysis Patterns: Reusable Object Models*. Menlo Park, California: Addison Wesley Longman, Inc.
- Fowler, M. & Scott, K. (1999). *UML Distilled: Applying the Standard Object Modeling Language*. Reading, MA: Addison Wesley Longman, Inc.
- Gane, C., Sarson, T. (1979). *Structured Systems Analysis: Tools and Techniques*. Englewood Cliffs, New Jersey: Prentice Hall, Inc.
- Highsmith, Jim (2002). *Agile Software Development Ecosystems*. Boston: Addison Wesley.
- Jacobson, I., Booch, G., & Rumbaugh, J., (1999). *The Unified Software Development Process*. Reading, MA: Addison Wesley Longman, Inc.
- Jefferies, R., Anderson, A., & Hendrickson, C. (2001). *Extreme Programming Installed*. Reading, MA: Addison Wesley Longman, Inc.
- Kruchten, P. (1999). *The Rational Unified Process: An Introduction*. Reading, MA: Addison Wesley Longman, Inc.
- Larman, C. (2002). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process*. Upper Saddle River, NJ: Prentice Hall PTR.



References and Recommended Reading

- 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. (1997). *DSDM: 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>

