



[Session S2]

AGILE SOFTWARE DEVELOPMENT: THE STATE OF THE ART

Scott Ambler

Senior Consultant, Ronin International Inc.

State of the Art

Modern Software Development Examined

Scott W. Ambler

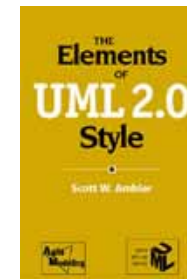
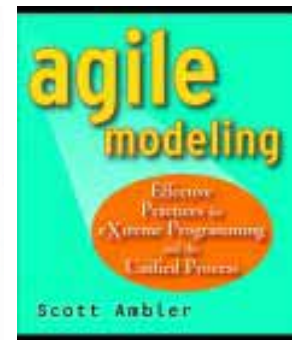
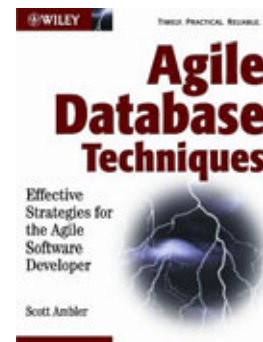
Senior Consultant, Ronin International

www.ambysoft.com/scottAmbler.html



Scott W. Ambler

- **Consultant:**
 - Agile Software Development
 - Software Process Improvement
- **Author:**
 - Agile Database Techniques
 - Agile Modeling
 - The Elements of UML 2.0 Style
 - The Object Primer 3rd Edition
 - Practical Guide to Enterprise Architecture
- **Contributing Editor/Writer:**
 - Software Development
 - IBM DeveloperWorks





Presentation Overview

- Warning!
- My Process Background
- Realities of Modern Software Development
- Agile Software Development
- “Traditional” Options
- Supporting Techniques
- Words of Advice





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





My Process Background

- Thought leader behind:
 - Pinball SDLC (1995)
 - Object-Oriented Software Process (1997)
 - Enterprise Unified Process (1998-2004)
 - Agile Modeling (2001-2004)
 - Agile Data (2002-2004)
- Actively developed software:
 - Following processes from very agile to very traditional
 - On small to very large projects (~\$100 million a year)
 - On short to very long projects (multi-year)





Realities of Modern Software Development

- The landscape:
 - Legacy technologies are here to stay
 - Legacy techniques are here to stay
 - Modern technologies are here to stay
 - Modern techniques are here to stay
- The implications:
 - Individuals need to be flexible
 - Large organizations need a wide range of techniques and technologies





Agile Software Development

Agile Software Development (ASD)

ASD Values

ASD Principles

Overview of Leading Agile Methods

Interesting Observations



Agile Software Development

- Agile software development is an approach to software development that is:
 1. People oriented
 2. That enables teams to respond effectively to change
 3. Results in the creation of working systems that meets the needs of its stakeholders

Agile Values

www.agilealliance.org

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

www.agilealliance.org

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



Agile Data Method

www.agiledata.org

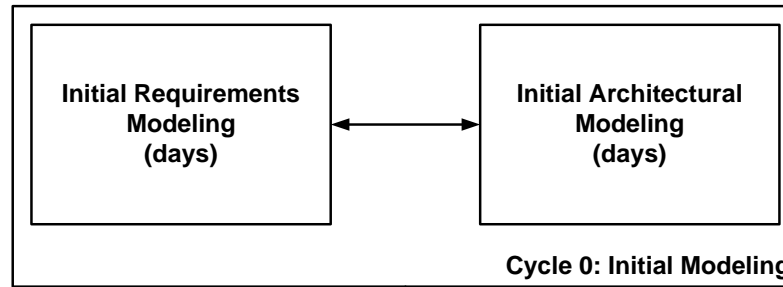


- Four roles:
 - Agile DBA
 - Developer
 - Enterprise Architect
 - Enterprise Administrator
- Six philosophies:
 - Data
 - Enterprise issues
 - Enterprise groups
 - Unique situation
 - Work together
 - Sweet spot

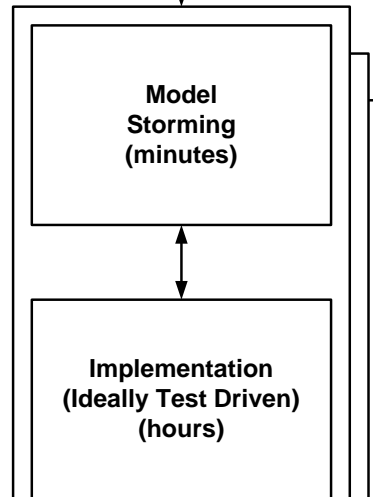


Agile Model Driven Development (AMDD) Project Level (www.agilemodeling.com/essays/amdd.htm)

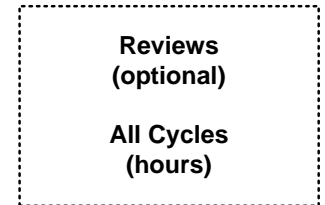
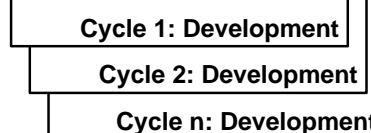
Goals: Gain an initial understanding of the scope, the business domain, and your overall approach.



Goal: Quickly explore in detail a specific issue before you implement it.



Goal: Develop working software in an evolutionary manner.



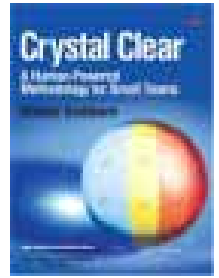


"Agile RUP"

- Theoretically possible, and some people are doing it
- RUP appeals to people with a prescriptive mindset, not an agile mindset
- The agile movement caught Rational by surprise, IMHO
- Everyone doesn't have to be agile
- If you truly want an agile method, consider something else
- www.agilemodeling.com/essays/agileModelingRUP.htm
- Skinnier RUP, Software Development, May 2004, www.sdmagazine.com

Crystal Clear

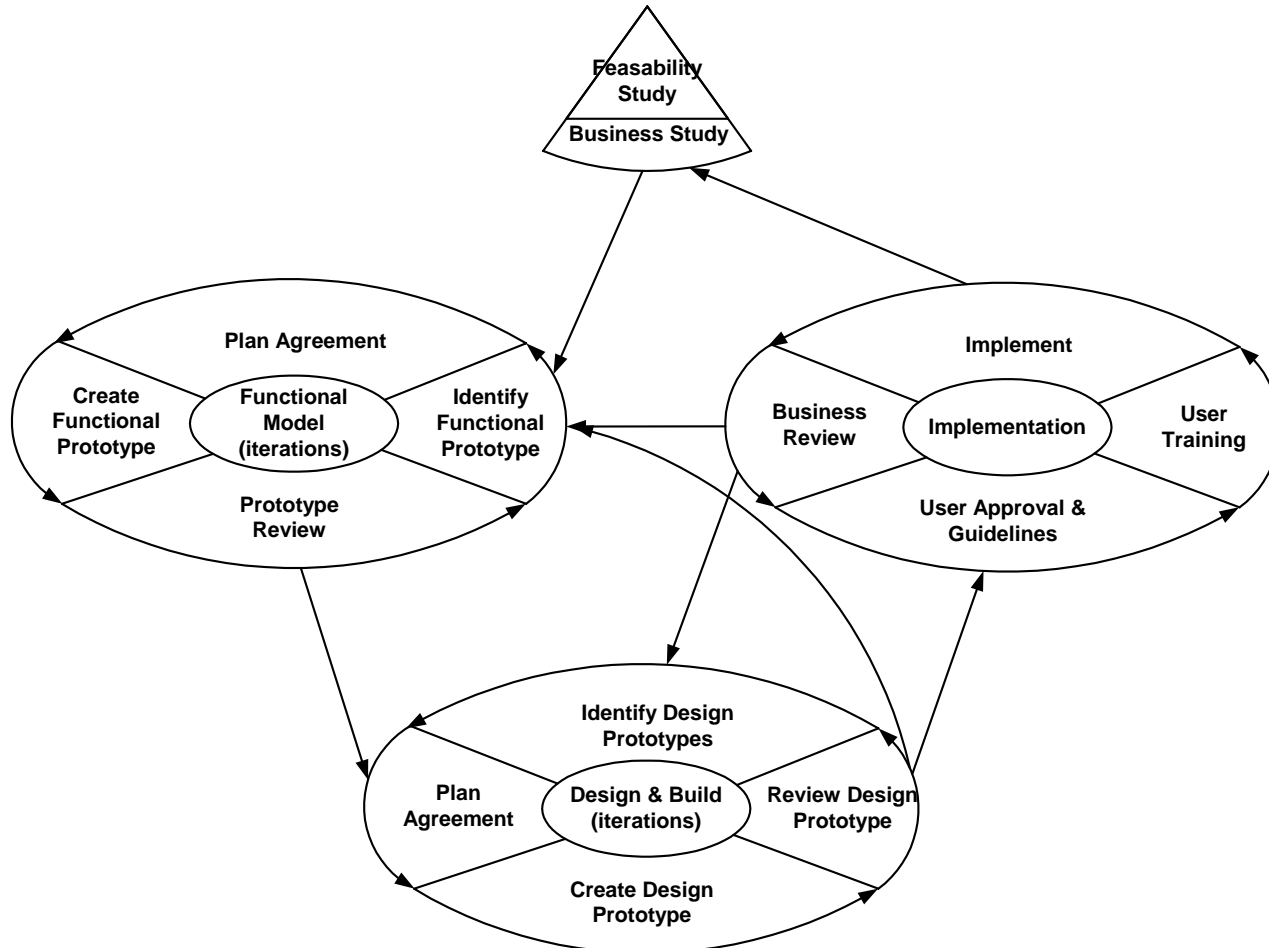
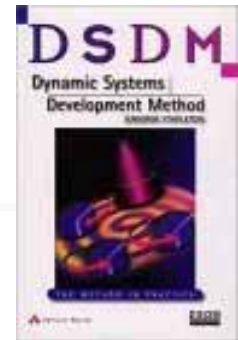
crystalmethodologies.org



- Seven properties:
 - Frequent delivery
 - Reflective improvement
 - Osmotic communication
 - Personal safety
 - Focus
 - Easy access to expert users
 - Technical environment with automated tests, configuration management, and frequent integration



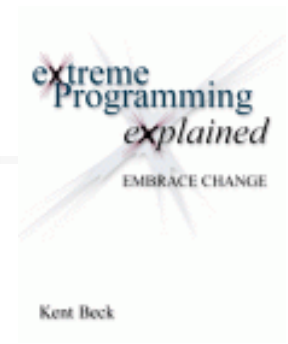
Dynamic System Development Method (DSDM) www.dsdm.org



Extreme Programming (XP)

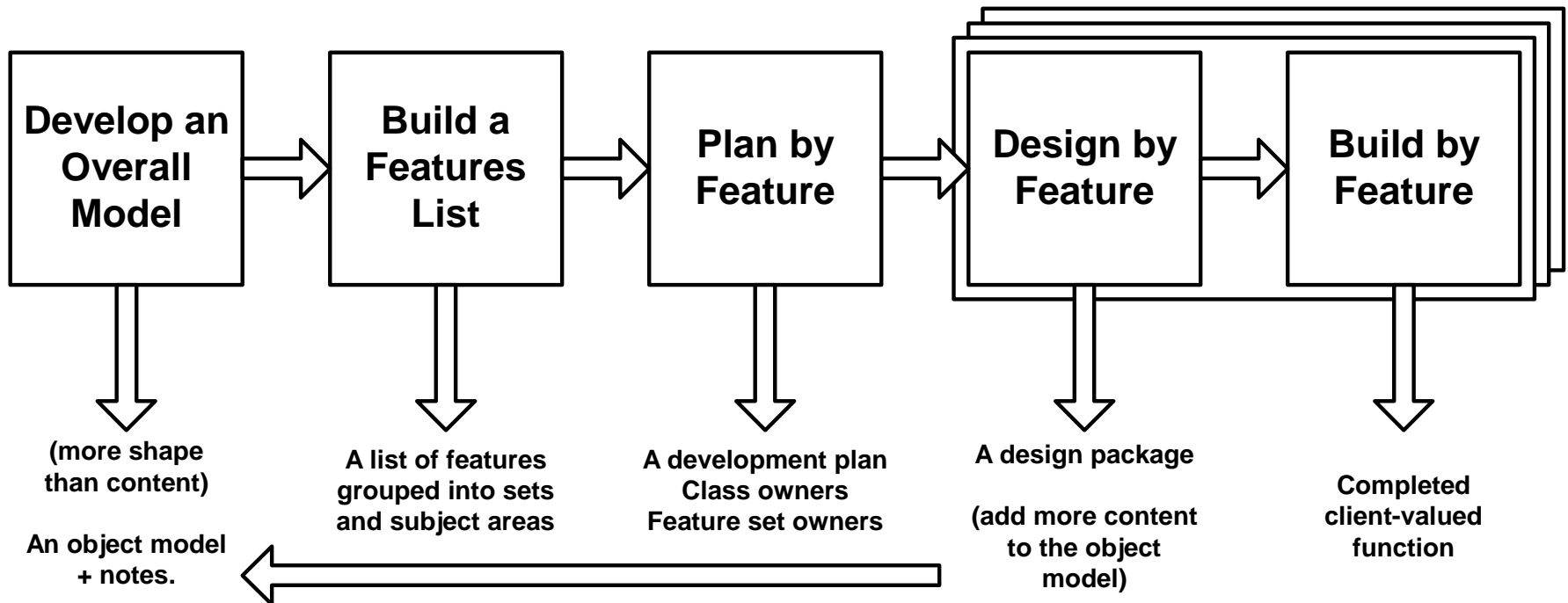
www.xprogramming.com

- 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



Feature Driven Development

(FDD) www.featuredrivendevelopment.com





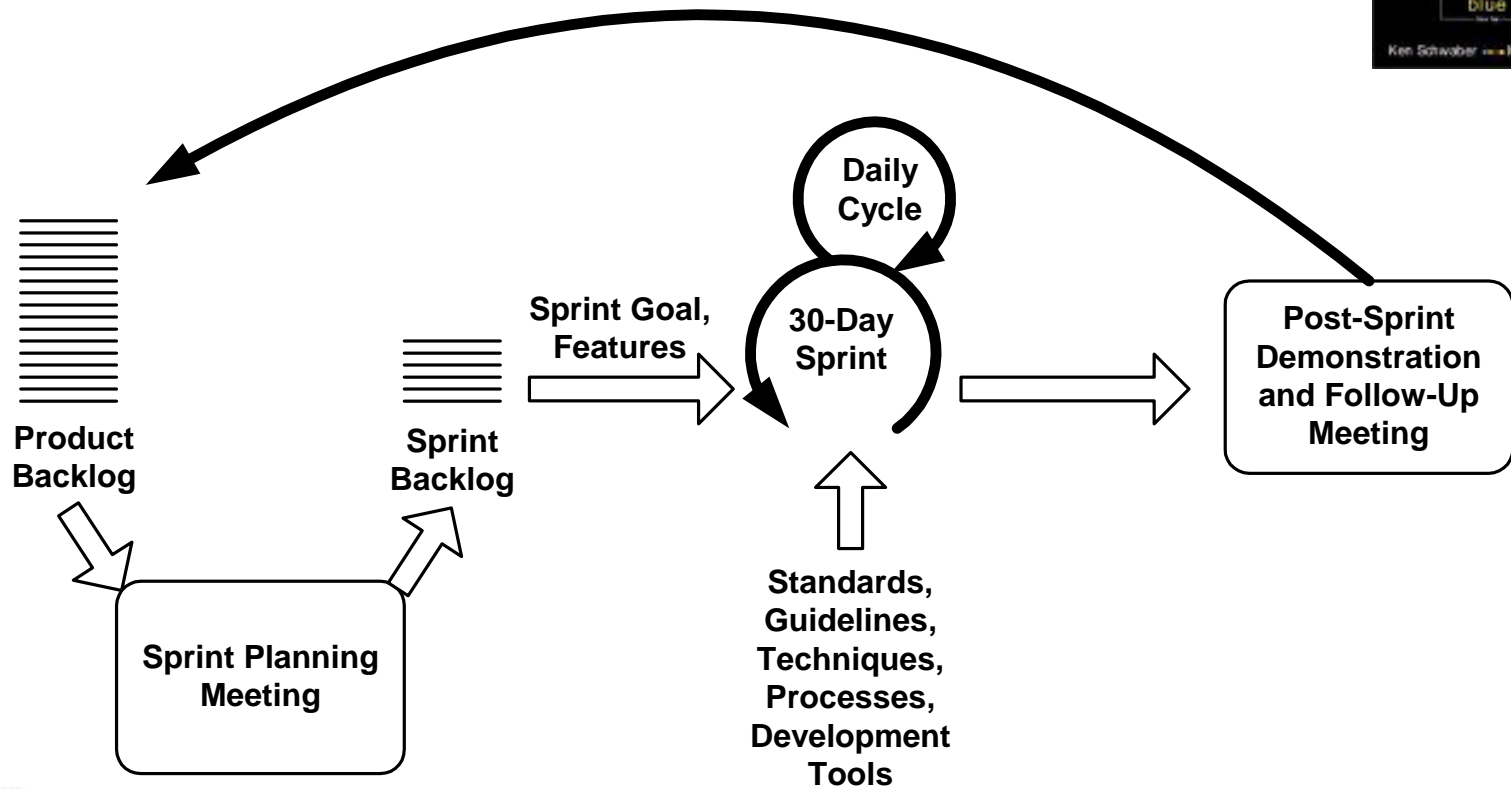
MSF Agile

- A new agile software development process from Microsoft.
- Scenario-driven development process for building .NET, Web, Web Service, and other object-oriented applications.
- Directly incorporates practices for handling quality of service requirements such as performance and security.
- Utilizes a context-driven approach (context-based) to determine how to operate the project.
- Builds upon the great work of the agile software development community



Scrum

www.controlchaos.com





Interesting Observations

- You need to become a generalizing specialist:
 - www.agilemodeling.com/essays/generalizingSpecialist.htm
- Agile software development is real and not a fad
- Agile software development is supported by a wide range of industry luminaries
- Research evidence support agile techniques is beginning to emerge
- Significant evidence exists showing that traditional techniques suffer from significant challenges
- Why is that the people saying agile doesn't work rarely seem to have tried it or even read a book about it?



“Traditional” Options

Capability Maturity Model (CMM)

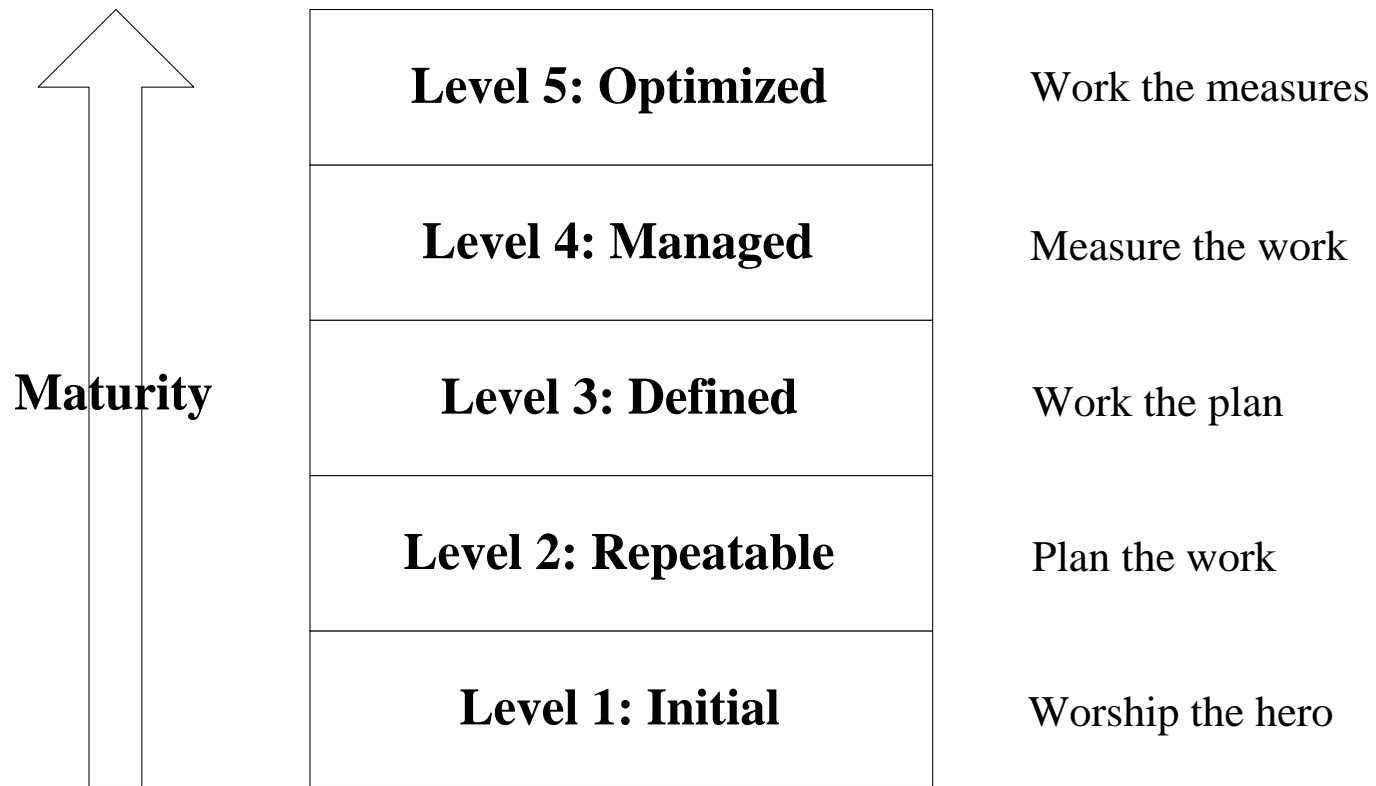
Rational Unified Process (RUP)

Enterprise Unified Process (EUP)

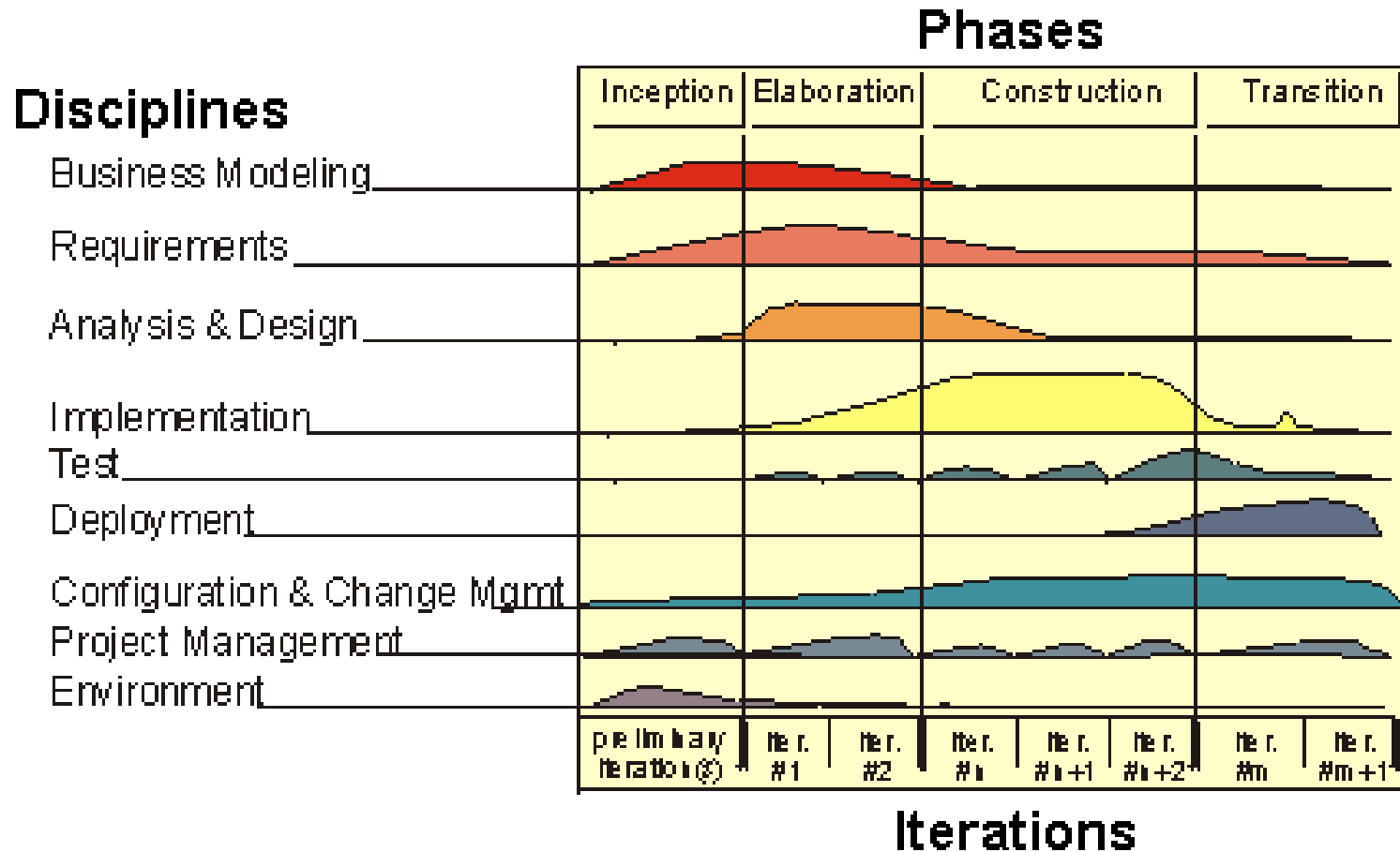
Six Sigma

Interesting Observations

Capability Maturity Model (CMM) Integration (CMMI)

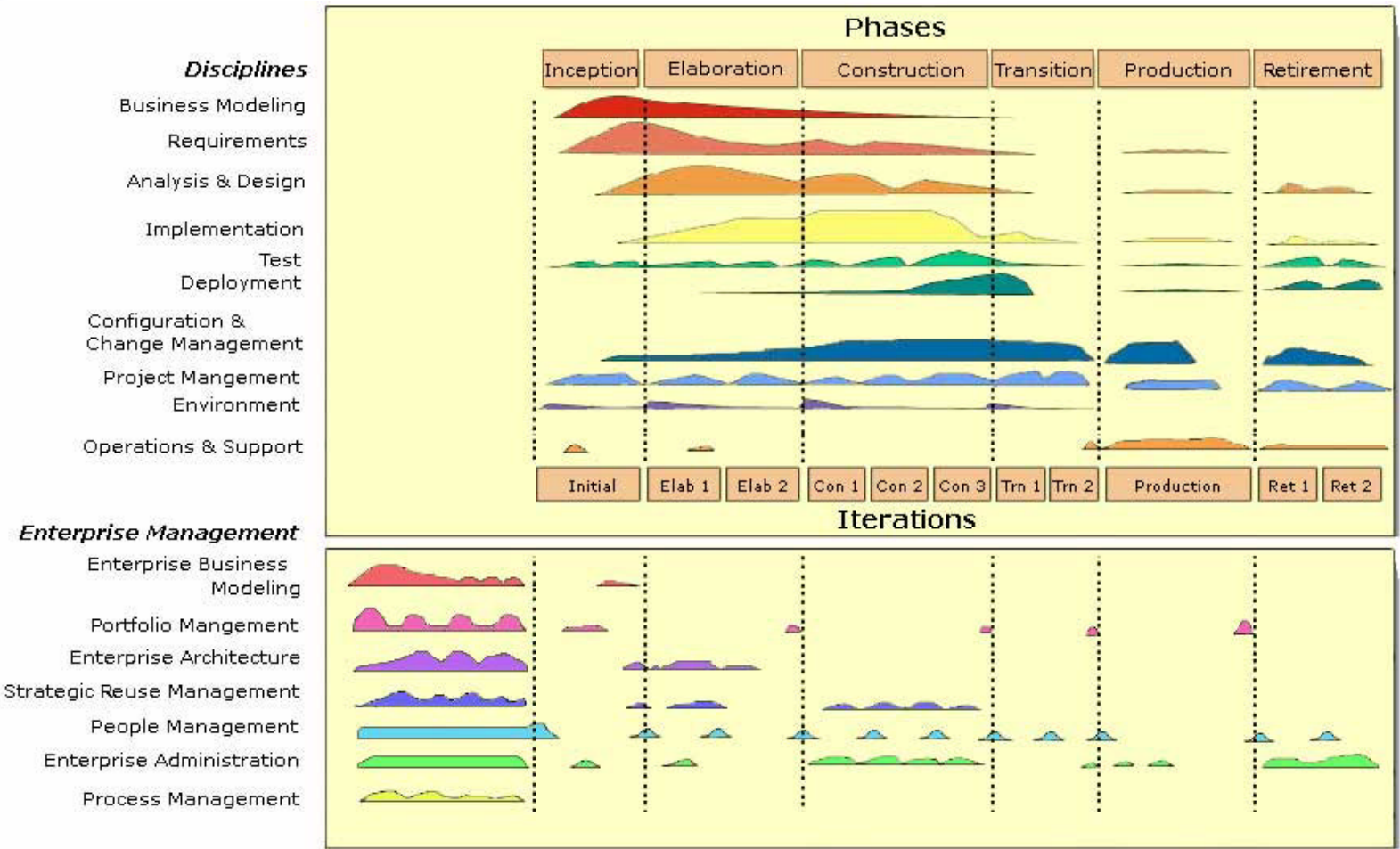
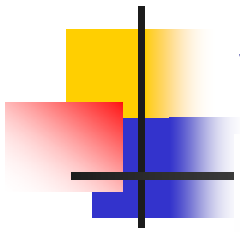
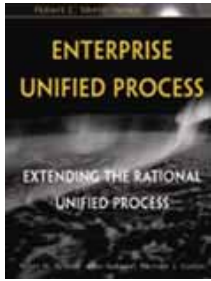


Rational Unified Process (RUP) v2003



Enterprise Unified Process (EUP)

v2004 www.enterpriseunifiedprocess.com





Six Sigma

- Things that make sense:
 - You should measure things and then improve based on that information
 - Don't let this get out of hand
- Things I have to question:
 - This is a manufacturing process improvement technique, not a software process technique
 - Originated within Motorola in the 1980s, when they dominated their industry, yet does Motorola dominate anything any more?
 - Green belts? Black belts? Yikes.





Interesting Observations

- Developers rarely seem to like these options
- The people that do like these options rarely seem to be directly involved with the development of software
- The “pro traditional” folks rarely seem to have taken an honest look at agile techniques

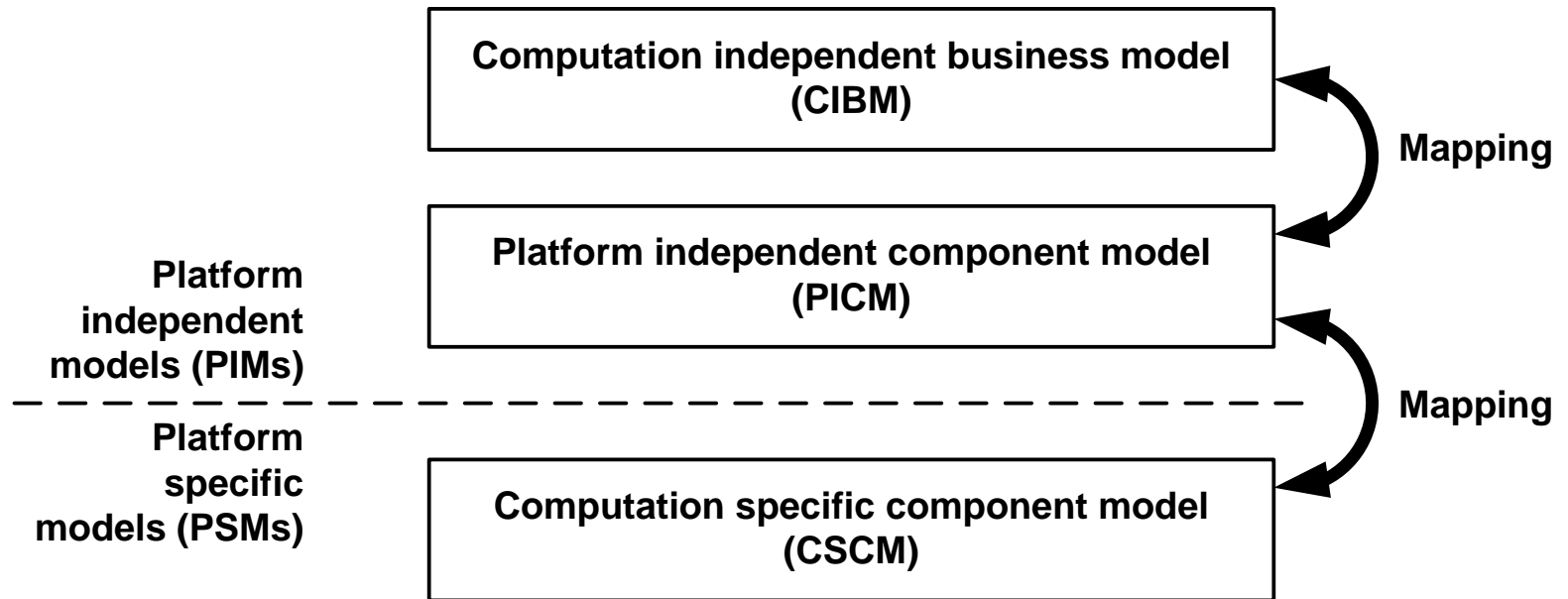


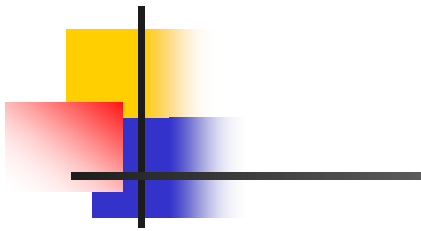
Supporting Techniques

Model Driven Architecture (MDA)
Unified Modeling Language (UML) 2.0
Interesting Observations

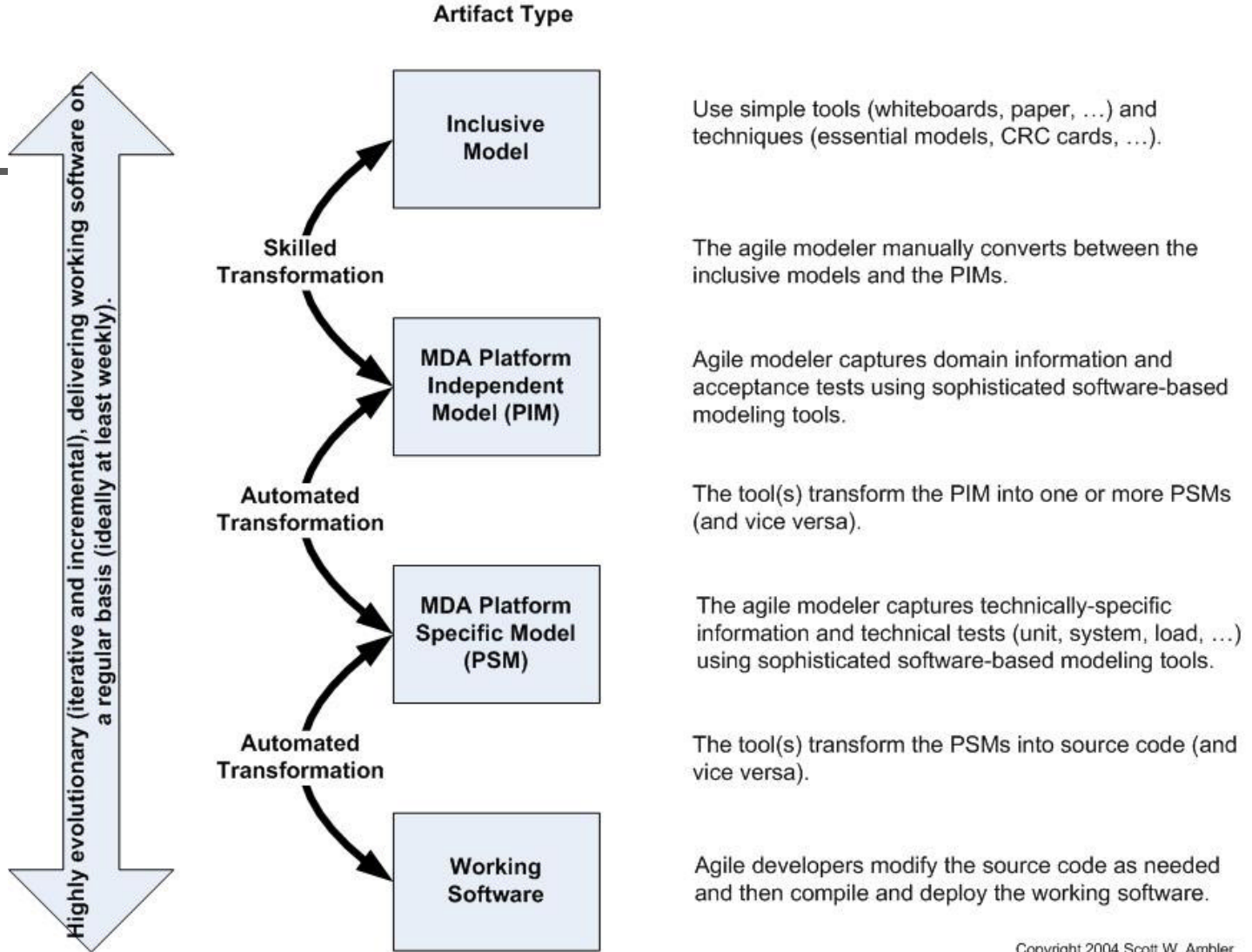


Model Driven Architecture (MDA)





Agile MDA Approach



Copyright 2004 Scott W. Ambler





UML 2: The Good News (IMHO)

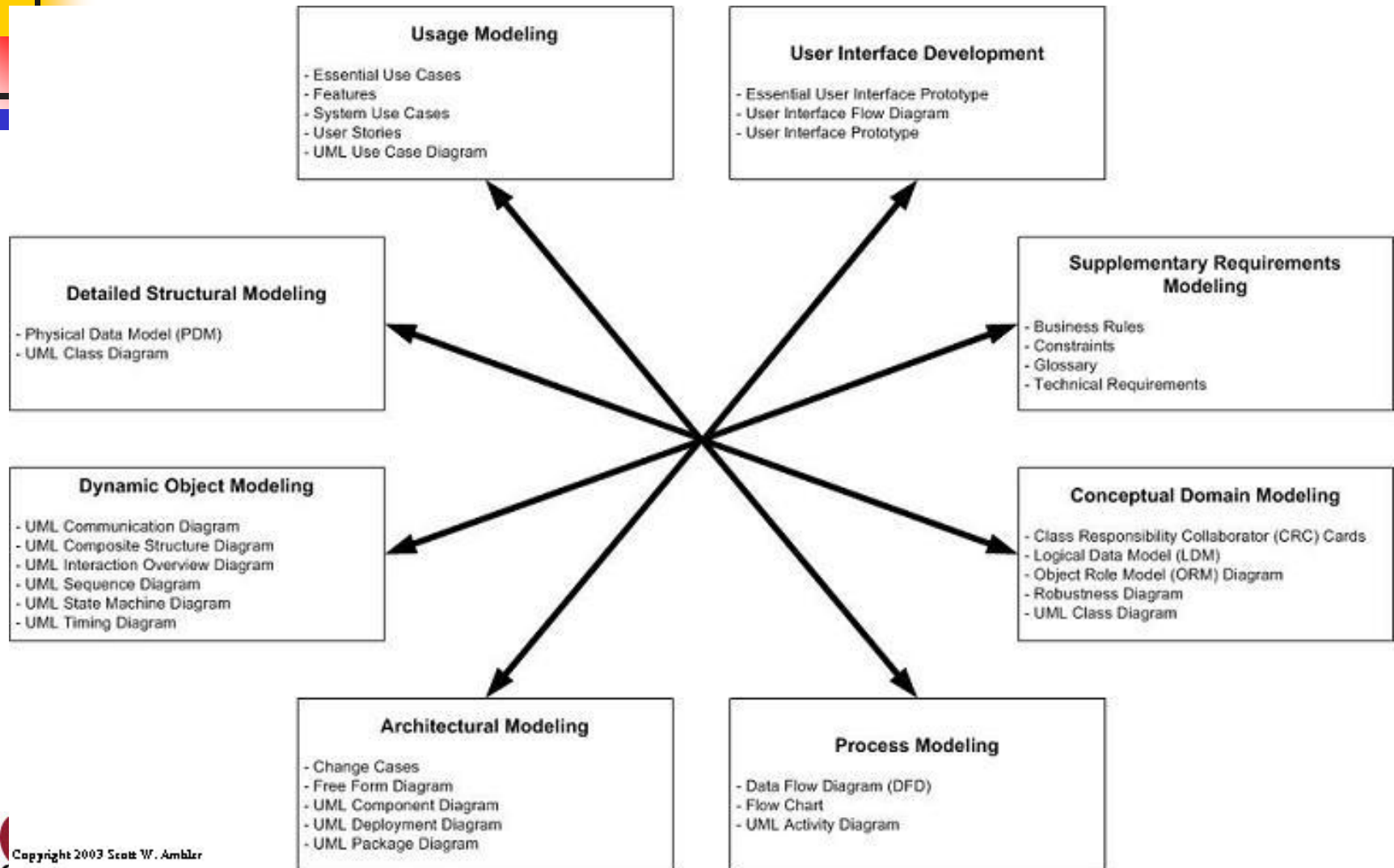
- Very consistent
- An evolutionary step forward from UML 1.x
- Much more precise than UML 1.x
- Reasonably easy to learn for existing UML practitioners
- Contains as much, and far more, detail than you need



UML 2: The Bad News (IMHO)

- UML 2 is still perceived by practitioners to come from IBM/Rational
 - UML 2 is still perceived by many to be tied into the RUP
- UML 2 appears to have been hijacked by the CASE tool vendors and to a lesser extent by academics
 - The primary motivation has been to support the OMG's Model Driven Architecture (MDA), a concept which very likely will fail in practice
 - UML 2 doesn't yet fully reflect the real-world needs of business developers
- Too many people are afraid of the "UML police" and thrash on "getting it right"
- UML 2 is both overkill and "underkill" for most projects

There is More to Modeling than UML





Interesting Observations

- MDA will likely work in a very small minority of organizations
- MDA is only a marketing brand, don't take it seriously
- The MDA is merely a rehash of the I-CASE vision of the late 1980s, and that failed miserably
- Every IT professional should understand the UML
- Why isn't there a use case model for the UML?



Words of Advice

- To individuals:
 - Seek to become a generalizing specialist
 - Focus on “soft skills” as well as hard ones
 - Be flexible
- To organizations:
 - One process size does not fit all
 - Focus on results, not bureaucracy
 - Be flexible

Questions?

Scott W. Ambler

www.ambysoft.com/scottAmbler.html

