




Sponsored by:




## Supporting eXtreme Programming with SCRUM

Joseph Pelrine  
C\*O, MetaProg GmbH  
jpelrine@metaprog.com



\* Copyright © 2003 MetaProg GmbH



”If we think to perpetuate the old ways, we should try to recall the last time evolution rang our number and asked consent.”

- Dee Hock, *Birth of the Chaordic Age*

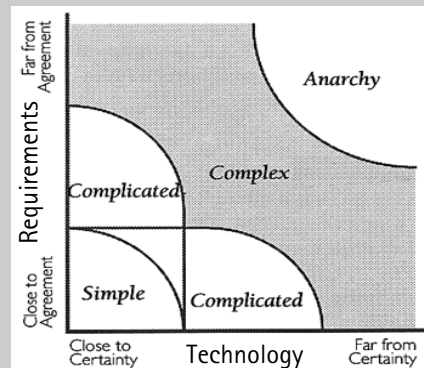
\* Copyright © 2003 MetaProg GmbH

## A Rough Overview

- ▶ Defined and empirical Processes
- ▶ Disentangling eXtreme Programming
- ▶ Introducing Scrum
- ▶ Combining XP and Scrum
- ▶ When and how to adapt processes

\* Copyright © 2003 MetaProg GmbH

## Categorization of complexity in development projects



- ▶ People are a third dimension!

- Graph by Ralph Stacey

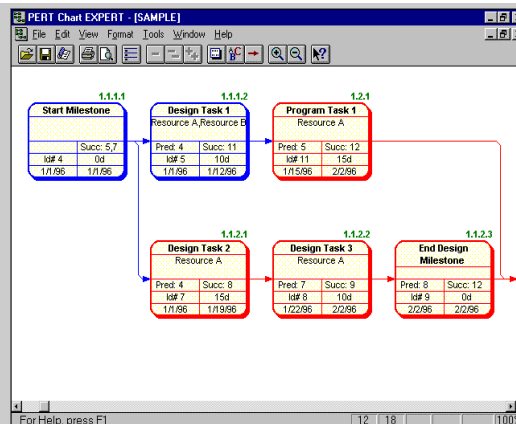
\* Copyright © 2003 MetaProg GmbH

"It is typical to adopt the defined (theoretical) modeling approach when the underlying mechanisms by which a process operates are reasonably well understood. When the process is too complicated for the defined approach, the empirical approach is the appropriate choice."

Ogunnaik and Ray, *Process Dynamics, Modeling, and Control*, Oxford University Press, 1992

\* Copyright © 2003 MetaProg GmbH

## Defined Processes



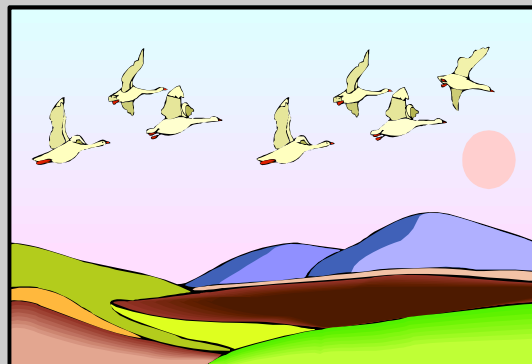
\* Copyright © 2003 MetaProg GmbH

## Defined Processes

- ▶ **Command and Control for simple projects**
- ▶ **Plan what you expect to happen**
- ▶ **Enforce that what happens is the same as what is planned**
- ▶ **Use change control to manage change**

\* Copyright © 2003 MetaProg GmbH

## Empirical Processes



\* Copyright © 2003 MetaProg GmbH

## Empirical Processes

- ▶ When you can't define things enough so that they run unattended and produce repeatable, acceptable quality output
- ▶ Empirical models are used when the activities are not predictable, are non-linear, and are too complex to define in repeatable detail
- ▶ Control is through inspection and adaptation.

\* Copyright © 2003 MetaProg GmbH

## Software Development is an Empirical Process

- ▶ Ziv's Uncertainty Principle in Software Engineering - uncertainty is inherent and inevitable in software development processes and products [Ziv, 1996].
- ▶ Humphrey's Requirements Uncertainty Principle - for a new software system, the requirements will not be completely known until after the users have used it.
- ▶ Wegner's Lemma - it is not possible to completely specify an interactive system [Wegner, 1995].

\* Copyright © 2003 MetaProg GmbH

## Agile Practices

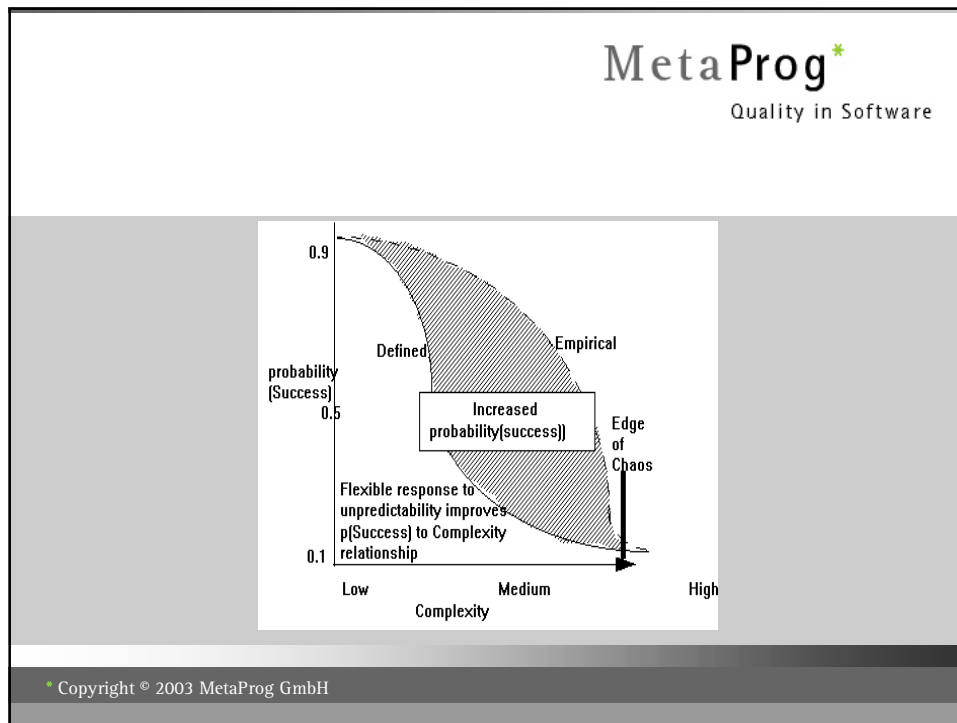
- ▶ **Agile lays out a vision and then nurtures project resources to do the best possible to achieve the plan.**
- ▶ **Agile is the "art of the possible."**
  
- ▶ **Agile employs the following practices:**
  - ▶ Frequent inspection
  - ▶ Emergence of requirements, technology, and team capabilities
  - ▶ Self-organization and adaptation in response to what emerges
  - ▶ Incremental emergence
  - ▶ Dealing with reality, not artefacts
  - ▶ Collaboration

\* Copyright © 2003 MetaProg GmbH

## We Use Agile Techniques to:

- ▶ **Increase control of a project**
- ▶ **Reduce the risk**
- ▶ **Maximize Return on Investment**
- ▶ **Increase probability of success**

\* Copyright © 2003 MetaProg GmbH



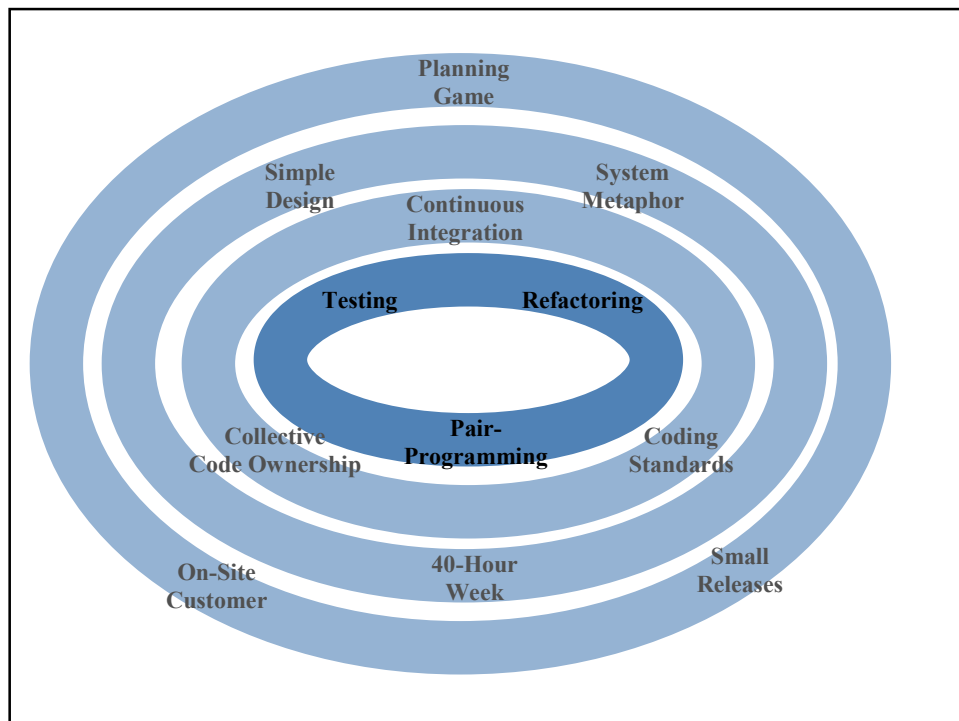
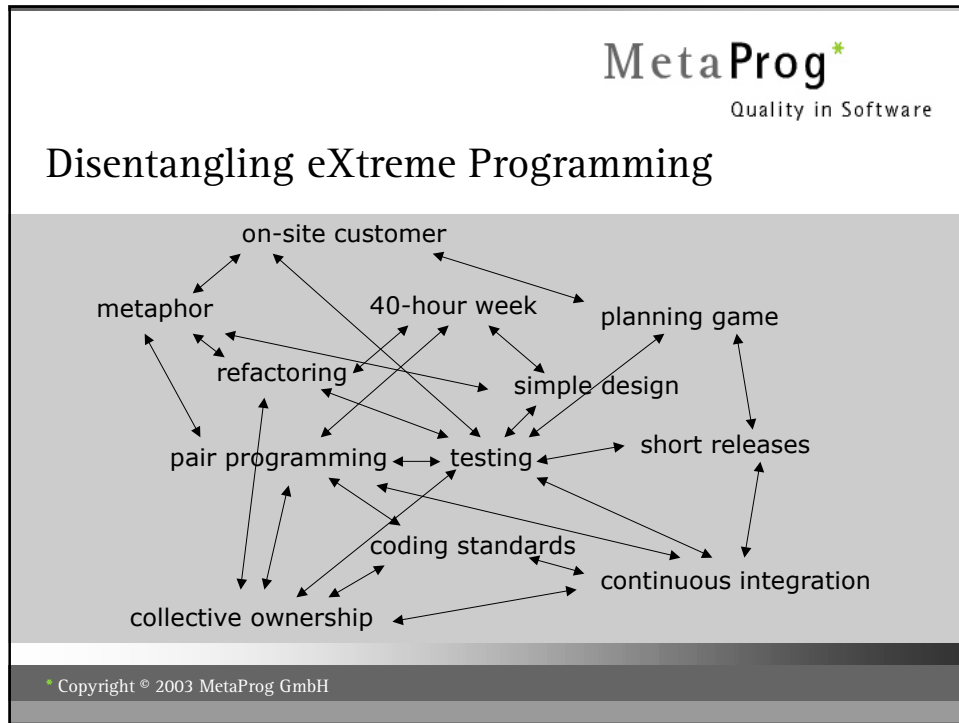
MetaProg\*  
Quality in Software

### Joseph's Basic Principle of XP

**"What would you do different if you knew that your customer could afford only one day of software development? ...**

**...especially if, by doing something different, he might be able to afford yet another day of software development"**

\* Copyright © 2003 MetaProg GmbH

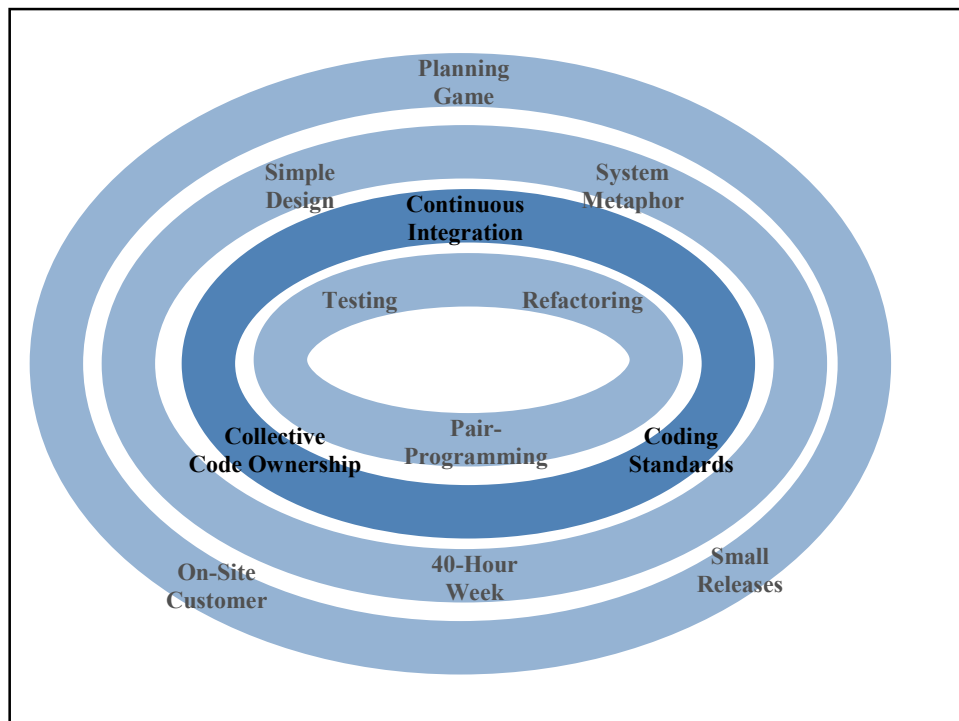


**MetaProg\***  
Quality in Software

## The Coding Circle

- ▶ Testing
- ▶ Refactoring
- ▶ Pair-Programming

\* Copyright © 2003 MetaProg GmbH

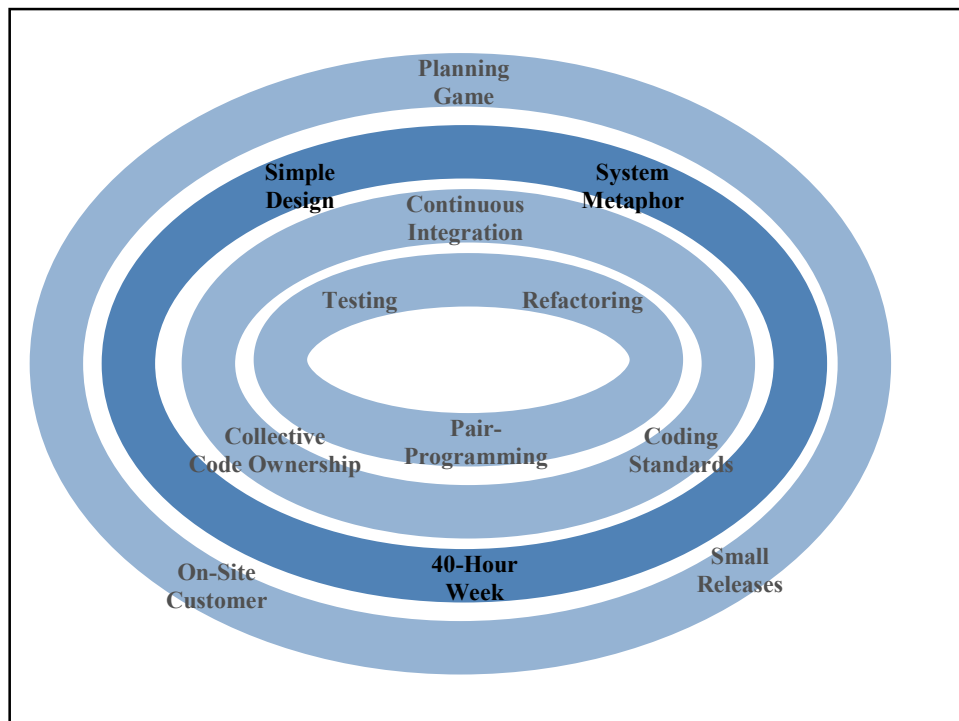


**MetaProg\***  
Quality in Software

## The Team Circle

- ▶ Coding Standards
- ▶ Collective Code Ownership
- ▶ Continuous Integration

\* Copyright © 2003 MetaProg GmbH

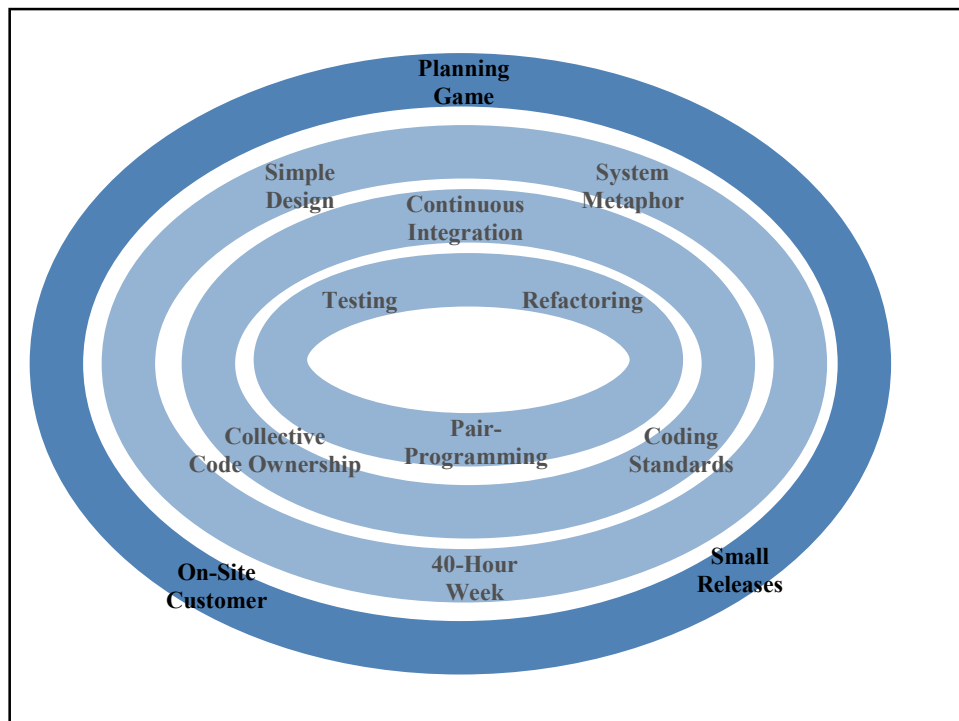


**MetaProg\***  
Quality in Software

## The Process Circle

- ▶ Simple Design
- ▶ System Metaphor
- ▶ 40-Hour Week

\* Copyright © 2003 MetaProg GmbH



**MetaProg\***  
Quality in Software

## The Product Circle

- ▶ **Planning Game**
- ▶ **Small Releases**
- ▶ **On-site Customer**

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## The Four Circles

- ▶ **Coding Circle**
- ▶ **Team Circle**
- ▶ **Process Circle**
- ▶ **Product Circle**

\* Copyright © 2003 MetaProg GmbH

## From Development to Organization...

- ▶ **eXtreme Programming is very good as a software development methodology**
- ▶ **The further out the practices get from the core circle, though, the less clearly defined they are**
- ▶ **What XP is missing is a compatible organizational methodology**
- ▶ **For this, we use Scrum**

\* Copyright © 2003 MetaProg GmbH

## Scrum Overview

- ▶ **Empirical management and control process for development efforts;**
- ▶ **Used at product companies and IT organizations since 1990;**
- ▶ **Wraps existing engineering practices;**
- ▶ **Extremely simple but very hard;**
- ▶ **CMM Level 2 compliant, partially Level 3**
- ▶ **Usually implements in 1 day, delivers business functionality in 30 days;**
- ▶ **Scalable; and**
- ▶ **Scrum feels completely different!**
  
- ▶ **First, cut out the noise...**

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## The Flow of Scrum

Scrum Flow

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## Scrum Practices – Product Planning Meeting

- ▶ Enough to drive first development Sprint to deliver product increment that provides business value;
- ▶ Requirements emerge as customer sees product increments;
- ▶ Systems architecture emerges as design emerges and is refactored; and
- ▶ Product architecture emerges as produce emerges and is refactored.

\* Copyright © 2003 MetaProg GmbH

## Scrum Practices – Scrum Master

- ▶ Responsible for establishing Scrum practices and rules
- ▶ Representative to management
- ▶ Representative to team
- ▶ A coach
- ▶ Engineering and development skills
- ▶ Agile version of IT project manager

\* Copyright © 2003 MetaProg GmbH

## Scrum Practices - Daily Scrum Meeting

- ▶ Daily 15 minute status meeting
- ▶ Same place and time every day
- ▶ Meeting room
- ▶ Chickens and pigs
- ▶ Three questions
- ▶ Impediments and Decisions

\* Copyright © 2003 MetaProg GmbH

## Scrum Practices - Scrum Teams

- ▶ **Self-organizing**
- ▶ **Cross-functional with no roles**
- ▶ **Seven plus or minus two**
- ▶ **Responsible for committing to work**
- ▶ **Authority to do whatever is needed to meet commitment**
- ▶ **Work environment**

\* Copyright © 2003 MetaProg GmbH

## Scrum Practices - Product Backlog

- ▶ **List of functionality, technology, issues**
- ▶ **Emergent, prioritized, estimated**
- ▶ **More detail on higher priority backlog**
- ▶ **One list for multiple teams**
- ▶ **Product Owner responsible for priority**
  - ▶ agile business project manager
- ▶ **Anyone can contribute**

\* Copyright © 2003 MetaProg GmbH

## Scrum Practices - Product Owner

- ▶ **One person;**
- ▶ **Sets development schedule by prioritizing backlog;**
- ▶ **Can be influenced by committees, management, customers, sales people, but is the only person that prioritizes;**
- ▶ **Responsible for ensuring that the most important business value is developed first;**
- ▶ **This mechanism ensures that only one set of requirements drives development; and**
- ▶ **Eliminates confusion of multiple bosses, different opinions, and interference.**

\* Copyright © 2003 MetaProg GmbH

## Scrum Practices - Sprint

- ▶ **Thirty calendar day iteration**
- ▶ **Team builds functionality that includes product backlog and meets Sprint goal**
- ▶ **Team self-organizes to do work**
- ▶ **Team conforms to existing standards and conventions**
- ▶ **Abnormal termination of Sprint**

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## Scrum Practices - End-of-Sprint Review

- ▶ **Analysis of**
  - ▶ Product backlog
  - ▶ Current product functionality
  - ▶ Current business and technology conditions
- ▶ **Review, consider and organize Info**
- ▶ **Set next Sprint goal**

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## Scrum Practices - Sprint Planning Meeting

Product Backlog

Team Capabilities

Business Conditions

Technology Stability

Executable Product Increment

→ **Review, Consider, Organize** →

**Next Sprint Goal**

**Product Backlog**

**Sprint Backlog**

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## Combining XP & Scrum

- ▶ Two complementary methodologies
- ▶ The XP Planning Game idea was taken directly from Scrum
- ▶ What happens if we turn the Scrum dials up to 10?

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## Scrum and eXtreme Programming

The diagram consists of two overlapping circles. The left circle is green and labeled 'Scrum'. The right circle is blue with horizontal lines and labeled 'xP'. The overlapping area in the center is labeled 'Planning'.

Sprint Sprint Planning End of Sprint Review Daily Scrum Product Owner Scrum Master Product Backlog Sprint Backlog	Simple Design Testing Test and Code Refactoring Pair Programming Collective Ownership Continuous Integration Coding Standards
--	--

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## Combining XP & Scrum

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## When and How to Adapt Processes

- ▶ The art of management and leadership is having an array of approaches and being aware of when to use which approach.
- ▶ Ralph Stacey proposed a matrix to help with this art by identifying management decisions on two dimensions:
  - ▶ the degree of certainty
  - ▶ the level of agreement

\* Copyright © 2003 MetaProg GmbH

## Close to Certainty

- ▶ Issues or decisions are close to certainty when cause and effect linkages can be determined.
- ▶ This is usually the case when a very similar issue or decision has been made in the past.
- ▶ One can then extrapolate from past experience to predict the outcome of an action with a good degree of certainty.

\* Copyright © 2003 MetaProg GmbH

## Far from Certainty

- ▶ At the other end of the certainty continuum are decisions that are far from certainty.
- ▶ These situations are often unique or at least new to the decision makers. The cause and effect linkages are not clear.
- ▶ Extrapolating from past experience is not a good method to predict outcomes in the far from certainty range.

\* Copyright © 2003 MetaProg GmbH

## Agreement

- ▶ **The vertical axis measures the level of agreement about an issue or decision within the group, team or organization.**
- ▶ **As you would expect, the management or leadership function varies depending on the level of agreement surrounding an issue.**

\* Copyright © 2003 MetaProg GmbH

## The Zones of the Matrix

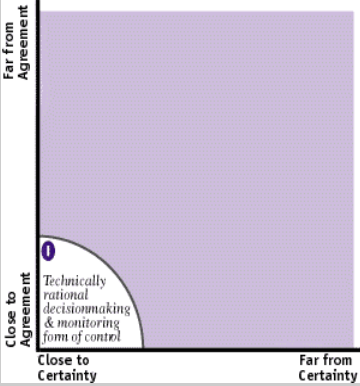
1. **Close to Agreement, Close to Certainty**
2. **Far from Agreement, Close to Certainty**
3. **Close to Agreement, Far from Certainty**
4. **Anarchy: Far from Agreement, Far from Certainty**
5. **The Edge of Chaos**

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## Close to Agreement, Close to Certainty

- ▶ **Much of management literature and theory addresses the region on the matrix which is close to certainty and close to agreement.**
  - ▶ use techniques which gather data from the past to predict the future.
  - ▶ plan specific paths of action to achieve outcomes and monitor the actual behavior by comparing it against these plans.
- ▶ **The goal is to repeat what works to improve efficiency and effectiveness.**

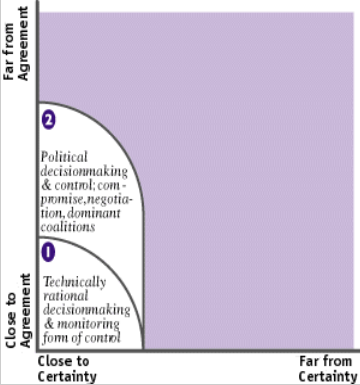


\* Copyright © 2003 MetaProg GmbH


**MetaProg\***  
Quality in Software

## Far from Agreement, Close to Certainty

- ▶ **Some issues have a great deal of certainty about how outcomes are created but high levels of disagreement about which outcomes are desirable.**
- ▶ **Neither plans nor shared mission are likely to work in this context.**
- ▶ **Instead, politics become more important.**
- ▶ **Coalition building, negotiation, and compromise are used to create the organization's agenda and direction.**



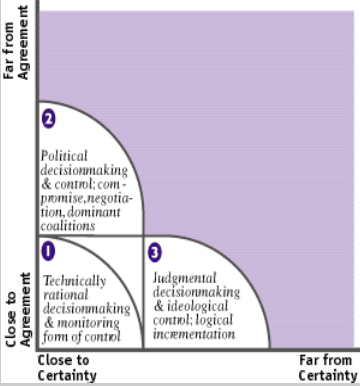
\* Copyright © 2003 MetaProg GmbH




Quality in Software

## Close to Agreement, Far from Certainty

- Some issues have a high level of agreement but not much certainty as to the cause and effect linkages to create the desired outcomes.
- In these cases, monitoring against a preset plan will not work. A strong sense of shared mission or vision may substitute for a plan in these cases.
- Comparisons are made not against plans but against the mission and vision for the organization.
- In this region, the goal is to head towards an agreed upon future state even though the specific paths cannot be predetermined.



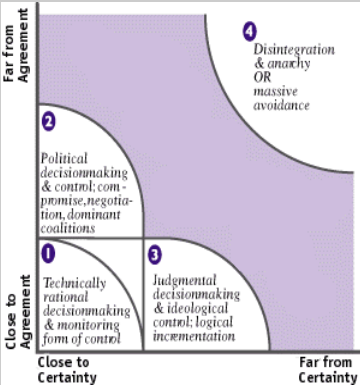
\* Copyright © 2003 MetaProg GmbH



Quality in Software

## Anarchy: Far from Agreement, Far from Certainty

- Situations where there are very high levels of uncertainty and disagreement, often result in a breakdown or anarchy.
- The traditional methods of planning, visioning, and negotiation are insufficient in these contexts.
- One personal strategy to deal with such contexts is avoiding the issues that are highly uncertain and where there is little agreement.
- While this may be a protective strategy in the short run, it is disastrous in the long run. This is a region that organizations should avoid as much as possible.



\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

## The Edge of Chaos

- ▶ There is a large area on this diagram which lies between the anarchy region and regions of the traditional management approaches.
- ▶ Stacey calls this large center region the zone of complexity - others call it the edge of chaos.
- ▶ In the zone of complexity the traditional management approaches are not very effective but it is the zone of high creativity, innovation, and breaking with the past to create new modes of operating.
- ▶ This is where agile processes are at home.

The diagram is a 2x2 matrix with 'Agreement' on the vertical axis (Close to Agreement at bottom, Far from Agreement at top) and 'Certainty' on the horizontal axis (Close to Certainty on left, Far from Certainty on right). It is divided into four numbered regions:

- 1 (Bottom-Left):** Technically rational decisionmaking & monitoring form of control
- 2 (Top-Left):** Political decisionmaking & control; compromise, negotiation, dominant coalitions
- 3 (Bottom-Right):** Judgmental decisionmaking & ideological control; logical incrementation
- 4 (Top-Right):** Disintegration & anarchy OR massive avoidance

There are also descriptive terms for the boundaries and center:

- Between 1 and 2: Intuition
- Between 2 and 4: Muddling through
- Between 1 and 3: Search for error
- Between 3 and 4: Unprogrammable decision-making - "outcomes" rather than solutions
- Center: Identification, development & selection
- Bottom-Right: Agenda building

\* Copyright © 2003 MetaProg GmbH

**MetaProg\***  
Quality in Software

This diagram shows the same axes as the previous one. A large purple shaded area covers the top-right and center-right portions of the matrix, representing the 'zone of complexity' or 'edge of chaos'.

This diagram is identical to the one in the first slide, showing the four numbered regions and their associated decision-making styles.

\* Copyright © 2003 MetaProg GmbH

## Managing Complexity

- ▶ We spend much of our time teaching and learning how to manage in areas (1), (2) and (3).
- ▶ In these regions, we can present models which extrapolate from past experience and thereby can be used to forecast the future.
- ▶ This is the hallmark of good science in the traditional mode.
- ▶ When we teach approaches, techniques and even merely a perspective in area (4) the models seem 'soft' and the lack of prediction seems problematic.

\* Copyright © 2003 MetaProg GmbH

## Adapting to Complexity

- ▶ We need to reinforce that managers and leaders of organizations need to have a diversity of approaches to deal with the diversity of contexts.
- ▶ Stacey's matrix honors what we already have learned but also urges us to move with more confidence into some of the areas which we understand intuitively but are hesitant to apply because they do not appear as 'solid.'
- ▶ This is the area where empirical processes have their strength.

\* Copyright © 2003 MetaProg GmbH

## Facilitation Tips

- ▶ Present the matrix in a layered format as shown above, with or without Stacey's words in the various regions.
- ▶ Ask participants to identify concrete examples from their workplace for each region.
- ▶ Discuss which approaches make sense for which region, and why.
- ▶ Discuss of what factors would suggest an issue is more likely to be in the zone of complexity or one of the other regions? Look for issues of how many people or institutions are connected, the time frame between the 'cause' and potential effects, areas of higher turbulence or unpredictability etc.
- ▶ Start with historical examples and then move into current or future issues.
- ▶ After the participants have worked with the matrix in a real context, have them reconvene and reflect on how they used it. Did they adapt it, change it for their own context?

\* Copyright © 2003 MetaProg GmbH

## Thanks

- ▶ Kent Beck
- ▶ Ron Jeffries
- ▶ Ken Schwaber
- ▶ Rachel Davies
- ▶ The Agile Alliance
- ▶ The Scrum Alliance

\* Copyright © 2003 MetaProg GmbH