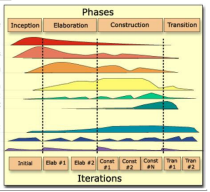
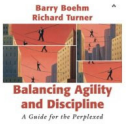
Are there software projects to which agile processes do not apply?

- Conclusion
  - No agile projects
    - If the developers do not want!
    - If management does not support it at all!
    - If requirements do not change often
      - Theoretical issue
    - Almost only theoretical reasons
  - But profit from plan driven processes
    - Focus on the architecture
      - Not satisfying your customer might not be your highest risk
    - Reduce risks in first iterations
    - Focus on useful templates
      - Estimations
      - Software Architecture Documentation
      - etc.
    - Take advantage of RUP Phases
      - Inception
      - Elaboration
- Introduction
  - To my person
    - Daniel Tobler
    - Software Architect
    - Realtime and Embedded Systems
    - Zühlke Engineering AG
  - How did I get in contact with Agility?
    - Started an agile project in 2000 without knowing about agility
    - No process (like RUP) --> we are bad
    - Educated by Ken Schwaber at OOPSLA 2002
    - CSM since 2006
  - Climbing strategies
    - Mount Everest

    - Uetliberg

  - Every project requires another process
- Where processes differ
  - Base on different values
    - Agile
      - 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 Manifesto
    - Plan-Driven
      - Develop Software iteratively
      - Manage requirements
      - Use component-based architectures
      - Visually model software
      - Verify Software quality
      - Control changes to Software
      - RUP Best Practises
  - Define different level of Ceremony
    - Low
    - High
    - Number of control elements (e.g. documents)
    - Quality of the control elements
  - Every Process requires Discipline, but
    - Self Discipline
      - Effective way to gain experience
      - Sport/Training
      - We do this because it helps us
      - Bases on own experience
    - Follow a Plan
      - Effective way to spread knowledge
      - Military
      - Follow that guideline and you will succeed
      - Standards and templates
- Plan-Driven Processes (RUP)
  - RUP as "lived" at Zühlke
  - Nature
    - Highly Iterative
    - Result Oriented
      - Every iteration delivers potential shippable software
    - Software Architecture centric
    - Risk driven
    - Focus on technology and guidelines
    - Phases



    - Many Templates and Guidelines
      - RUP on 1 DVD
      - Most of the content not useful, no practical background (rubbish)
      - Needs to be tailored
  - Advantages
    - Reducing the risks is highlighted first
      - Embedded: Getting the adequate Hardware is always a risk
      - Performance is an issue
      - Architecture must be proven first
    - Focus on Software Architecture
    - Inception and Elaboration
    - RUP Light: Relatively simple to introduce and offers good guidelines
  - Disadvantages
    - No Team
      - Project Manager is responsible
      - Software Architect is responsible
    - Theory and Practice often different
      - Long iterations (undiscoverable)
      - Software Architect as god
      - Project runs as waterfall since process is not understood
      - Process defined, but nobody "lives" it
- Agile Processes (Scrum)
  - Scrum/XP Hybrid
  - Nature
    - Highly iterative
    - Result Oriented
    - Documented in a small book
    - Focus on people
      - Cannot be learned from a book
    - Psychological elements
    - Often combined with XP
  - Advantages
    - Team-Involvement
    - Process improvements
      - Everybody is responsible
      - RUP defines this as well, but this fact gets lost among all other process details (I just say: 1 DVD)
      - Explicitly at each iteration
      - Elements of other processes explicitly allowed
    - Overtime is distributed more evenly
  - Disadvantages
    - Software Architecture may get lost
    - Weakness in definition of initial phases
    - Focus only THE ONLY risk: Customer is not satisfied
- Comparison with Decision Factors
  - Differ using Decision Factors
    - Size
      - Size
        - Number of people involved in the project.
      - Consider developers, testers, integrators, architects, and project managers
      - Offshore requires more developers.
    - Criticality
      - Criticality
        - Damage to the user due to impact of defect in the software.
      - Do not measure the damage the project causes to the company developing the software!
      - The higher the criticality, the more initial analysis is necessary
    - Team skills
      - Team Skills
        - Percentage of developers able to work with a process (Level 1) versus developers able to design a process (Levels 2&3).
      - Level 1: Developers working according a process
      - Level 2: Developers able to expand and tailor a process
      - Level 3: Developers knowing the reasons for certain methodologies
      - The level has nothing do to with experience
      - It is not necessary to have only Level 3 developers
    - Change
      - Change
        - Number of requirements that can or have to be collected at the beginning of the project and that remain unchanged until the final release.
      - The team already develops the 10th tax regulation application
      - Measure for the maturity of the team in the domain
      - The requirements for a nuclear power plant must have different quality than those of a lunch ordering web application
      - Measure for the complexity of the architecture
        - The architecture of a lunch ordering web page is not that complex because it uses a standard architecture.
    - Culture
      - Culture
        - How well is agility accepted in your organization? Management and developers may have different stakes.
      - A.) How well the developers accept an agile process
      - B.) How well the management/customer accepts an agile process
      - Distinguish between both!
  - Size
    - Being used for large projects
    - But successful?
    - What is successful at large projects?
      - Force direct communication
      - Plan enough time for communication
    - Both processes "work" on large processes
  - Criticality
    - FDA/TÜV trust V-Modell, etc.
    - Prototype with an agile process, final product with a formally accepted process
    - Critical products are often embedded
      - No standard architecture
      - Coordination Points
      - Embedded Projects require coordination with Hardware and Mechanics
      - Mechanics and especially Hardware cannot be developed iteratively
      - Hardware must support defined Software Architecture
      - First iterations focus on concepts, not on customer benefit
    - The more coordination points the longer the initial phase
    - Critical products can be developed with an agile approach
  - Team skills
    - Level 1 developers need more guidance
    - More guidelines for Level 1 developers
    - But
      - Play piano-beginners better with more conductors?
      - Project Manager
        - = Team Leader
        - = Technical Project Manager (Architect)
        - under permanent stress
      - Tailoring

        - Overwhelming documentation requires experience
        - Tailoring done by committee
        - but tailoring for every project required! Climbing strategies
        - tailoring for biggest possible project
        - Committee is responsible for improvement exclusively
        - Post-Mortem retrospectives
    - Dou you really want to make a project with only level 1 developers?
    - Do you really want to make a project with only level 3 developers?
    - If you have mostly level 1 developers
      - Use a few useful Guidelines from RUP
      - Then move responsibility to the developers to form a team
  - Change
    - Theoretical issue
    - If requirements do not change, waterfall is sufficient!
    - There are projects where more requirements must be collected up front
      - to define the architecture
      - to satisfy regulatory issues
      - Sad, but fact
      - fixed price contracts
    - RUP offers initial phases
      - Inception
      - Elaboration
    - Collect high level requirements if you have the chance
      - But do not detail them
    - Always reduce risks first
      - Not satisfying the customer is a risk too
      - For some applications this is the highest risk
      - For some applications it is not
    - It is good idea to plan initial iterations
  - Culture
    - Yes, there are developers that do not want RUP
    - Yes, there are developers that do not want Agility
    - You cannot convince the majority