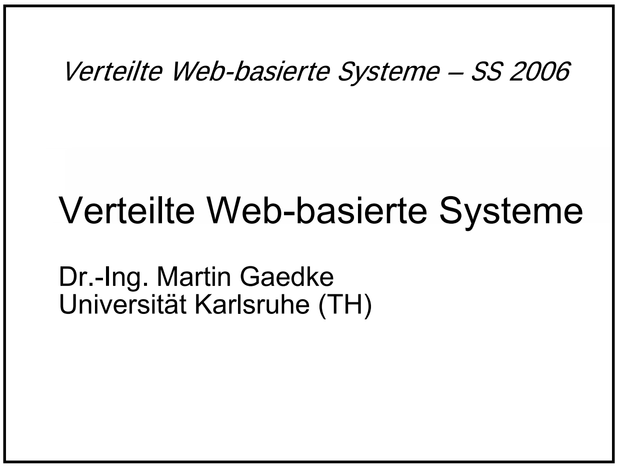
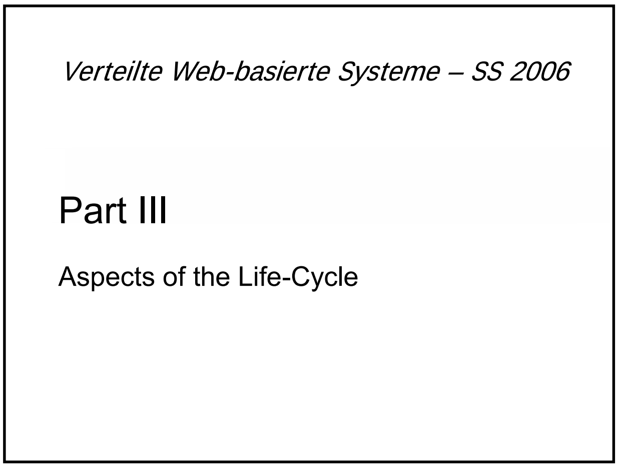


Verteilte Web-basierte Systeme – SS 2006

Verteilte Web-basierte Systeme

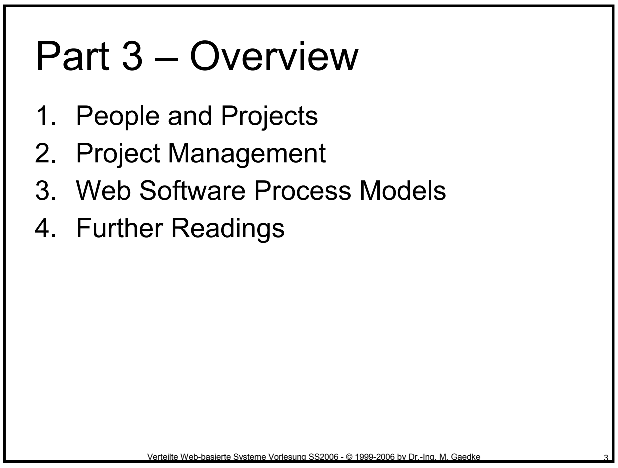
Dr.-Ing. Martin Gaedke
Universität Karlsruhe (TH)



Verteilte Web-basierte Systeme – SS 2006

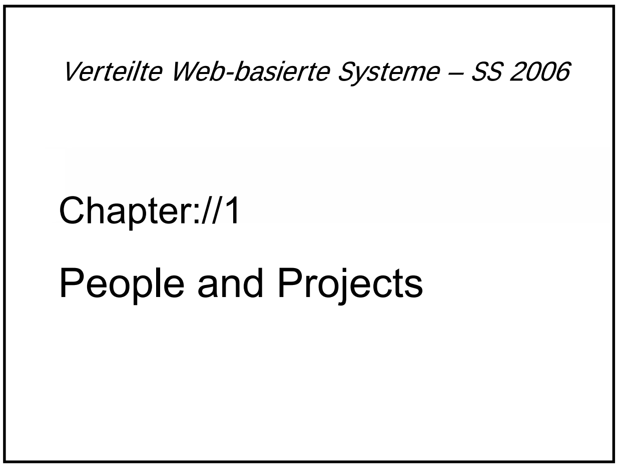
Part III

Aspects of the Life-Cycle



Part 3 – Overview

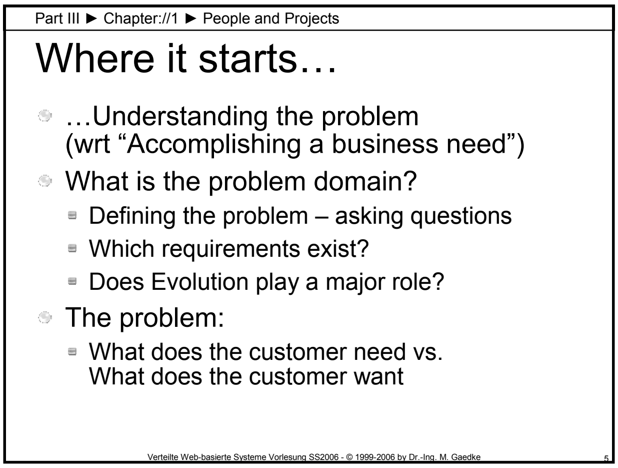
1. People and Projects
2. Project Management
3. Web Software Process Models
4. Further Readings



Verteilte Web-basierte Systeme – SS 2006

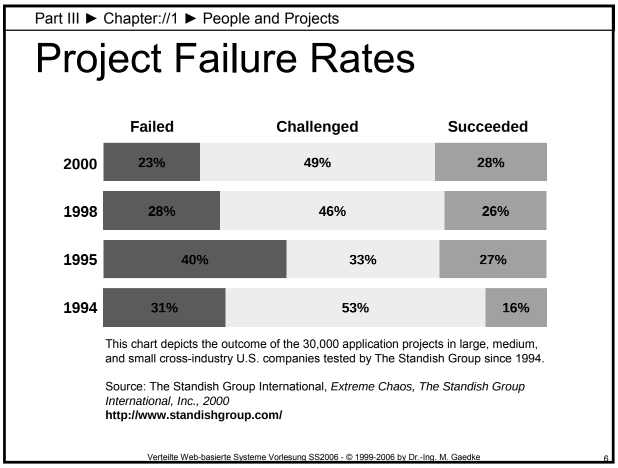
Chapter://1

People and Projects



Where it starts…

- …Understanding the problem
  (wrt “Accomplishing a business need”)
- What is the problem domain?
  - Defining the problem – asking questions
  - Which requirements exist?
  - Does Evolution play a major role?
- The problem:
  - What does the customer need vs.
    What does the customer want

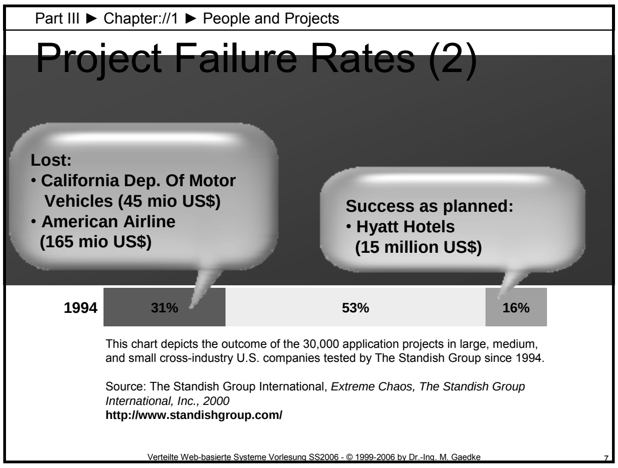


Project Failure Rates

	Failed	Challenged	Succeeded
2000	23%	49%	28%
1998	28%	46%	26%
1995	40%	33%	27%
1994	31%	53%	16%

This chart depicts the outcome of the 30,000 application projects in large, medium, and small cross-industry U.S. companies tested by The Standish Group since 1994.

Source: The Standish Group International, *Extreme Chaos, The Standish Group International, Inc., 2000*
http://www.standishgroup.com/



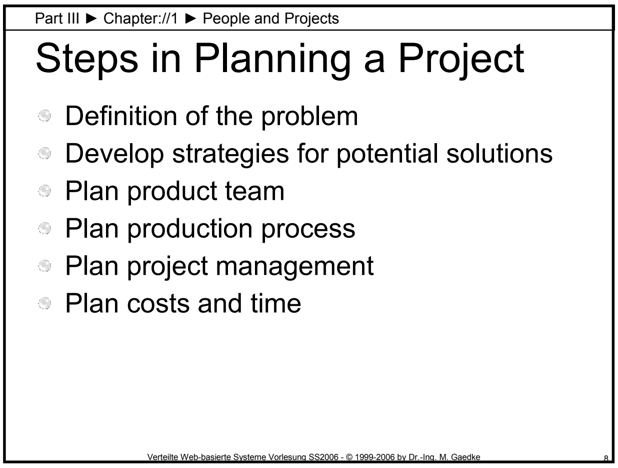
Project Failure Rates (2)

Lost:
- **California Dep. Of Motor Vehicles (45 mio US$)**
- **American Airline (165 mio US$)**

Success as planned:
- **Hyatt Hotels (15 million US$)**

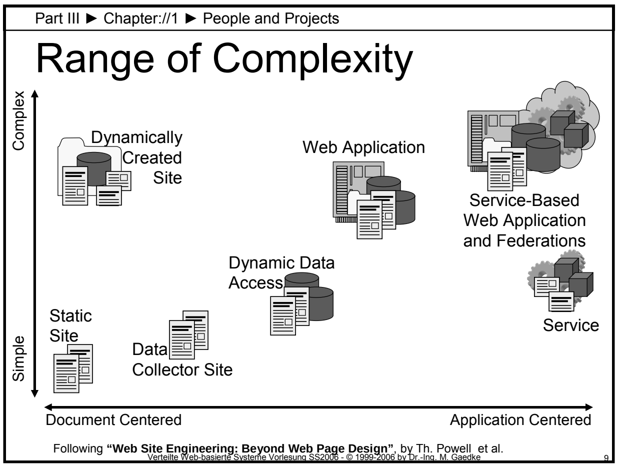
This chart depicts the outcome of the 30,000 application projects in large, medium, and small cross-industry U.S. companies tested by The Standish Group since 1994.

Source: The Standish Group International, *Extreme Chaos, The Standish Group International, Inc., 2000*
http://www.standishgroup.com/



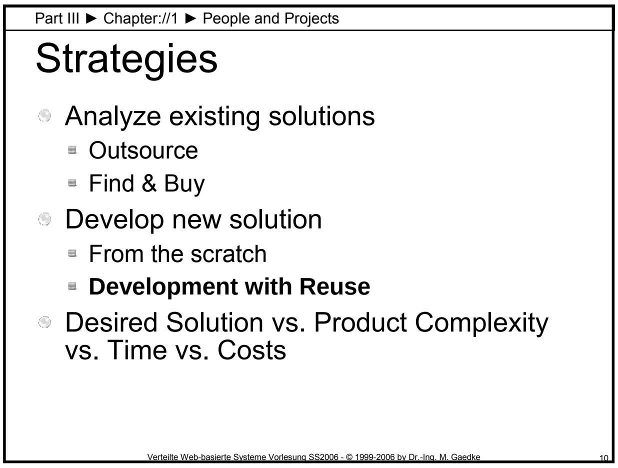
Steps in Planning a Project

- Definition of the problem
- Develop strategies for potential solutions
- Plan product team
- Plan production process
- Plan project management
- Plan costs and time



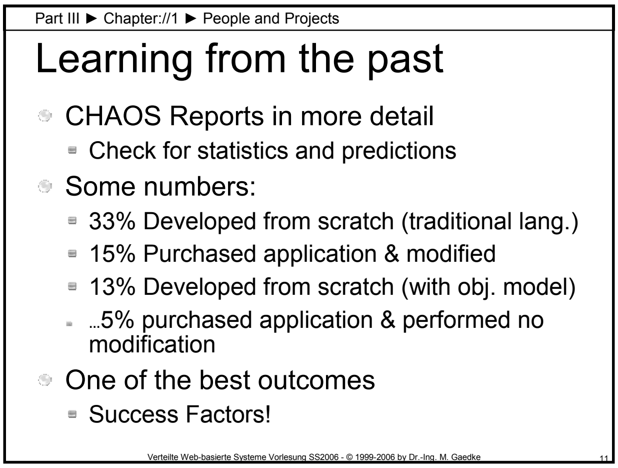
Range of Complexity

Following **"Web Site Engineering: Beyond Web Page Design"**, by Th. Powell et al.



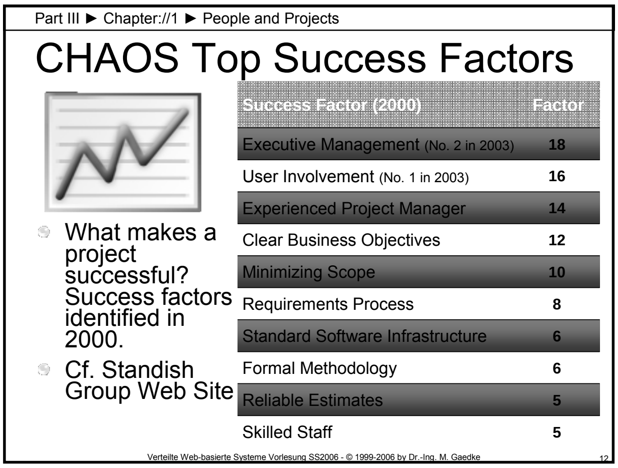
Strategies

- Analyze existing solutions
  - Outsource
  - Find & Buy
- Develop new solution
  - From the scratch
  - **Development with Reuse**
- Desired Solution vs. Product Complexity vs. Time vs. Costs



Learning from the past

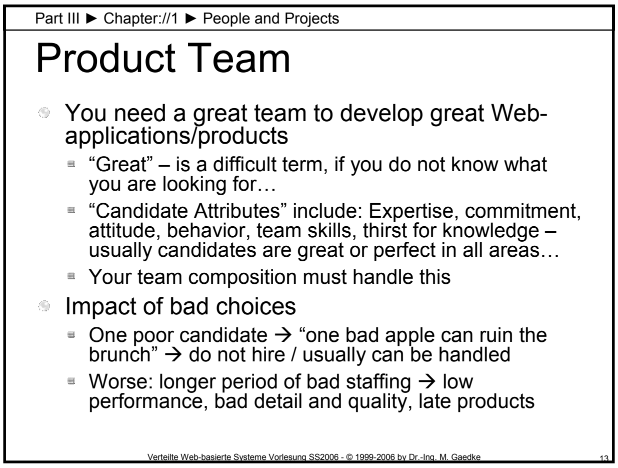
- CHAOS Reports in more detail
  - Check for statistics and predictions
- Some numbers:
  - 33% Developed from scratch (traditional lang.)
  - 15% Purchased application & modified
  - 13% Developed from scratch (with obj. model)
  - …5% purchased application & performed no modification
- One of the best outcomes
  - Success Factors!



CHAOS Top Success Factors

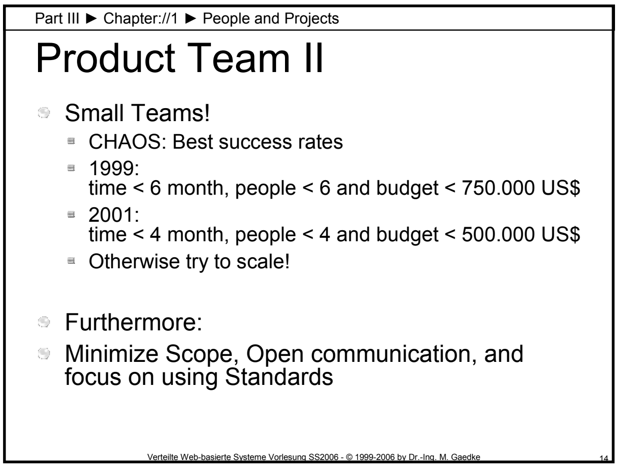
- What makes a project successful? Success factors identified in 2000.
- Cf. Standish Group Web Site

Success Factor (2000)	Factor
Executive Management (No. 2 in 2003)	18
User Involvement (No. 1 in 2003)	16
Experienced Project Manager	14
Clear Business Objectives	12
Minimizing Scope	10
Requirements Process	8
Standard Software Infrastructure	6
Formal Methodology	6
Reliable Estimates	5
Skilled Staff	5



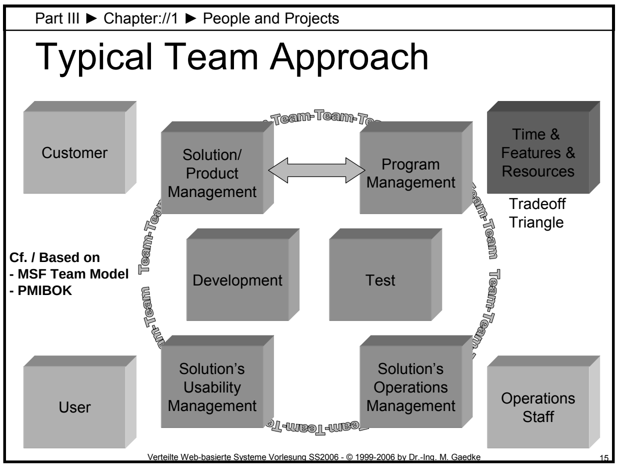
Product Team

- You need a great team to develop great Web-applications/products
  - “Great” – is a difficult term, if you do not know what you are looking for…
  - “Candidate Attributes” include: Expertise, commitment, attitude, behavior, team skills, thirst for knowledge – usually candidates are great or perfect in all areas…
  - Your team composition must handle this
- Impact of bad choices
  - One poor candidate → “one bad apple can ruin the brunch” → do not hire / usually can be handled
  - Worse: longer period of bad staffing → low performance, bad detail and quality, late products



Product Team II

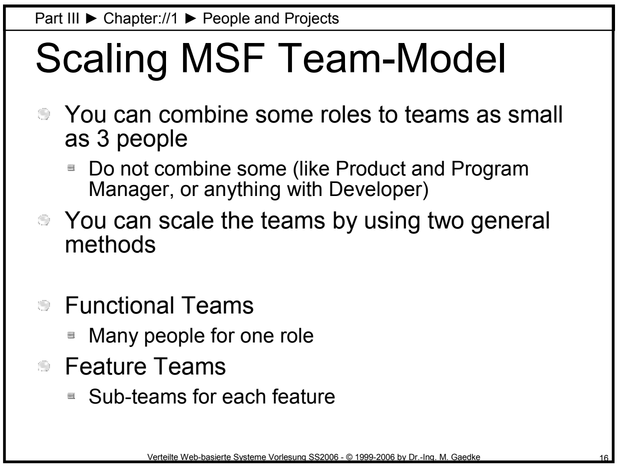
- Small Teams!
  - CHAOS: Best success rates
  - 1999:
    time < 6 month, people < 6 and budget < 750.000 US$
  - 2001:
    time < 4 month, people < 4 and budget < 500.000 US$
  - Otherwise try to scale!
- Furthermore:
- Minimize Scope, Open communication, and focus on using Standards



Typical Team Approach

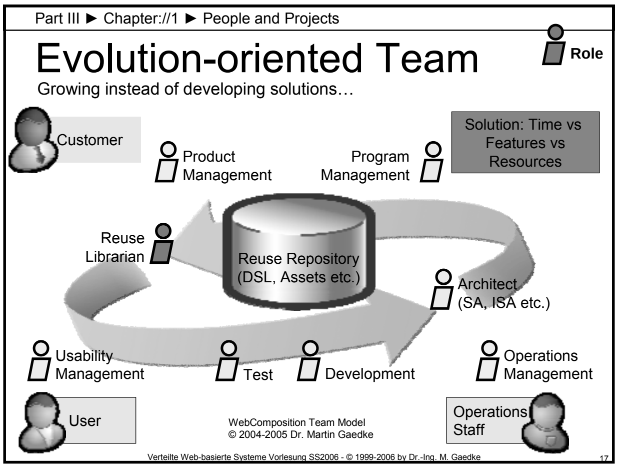
Customer
Solution/ Product Management
Program Management
Time & Features & Resources
Tradeoff Triangle
Team-Team-Team
Development
Test
User
Solution’s Usability Management
Solution’s Operations Management
Operations Staff

Cf. / Based on
- MSF Team Model
- PMIBOK



Scaling MSF Team-Model

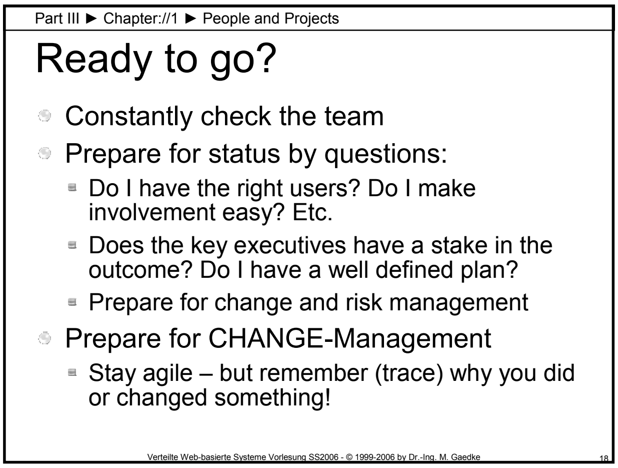
- You can combine some roles to teams as small as 3 people
  - Do not combine some (like Product and Program Manager, or anything with Developer)
- You can scale the teams by using two general methods
- Functional Teams
  - Many people for one role
- Feature Teams
  - Sub-teams for each feature



Evolution-oriented Team

Growing instead of developing solutions…

WebComposition Team Model
© 2004-2005 Dr. Martin Gaedke



Ready to go?

- Constantly check the team
- Prepare for status by questions:
  - Do I have the right users? Do I make involvement easy? Etc.
  - Does the key executives have a stake in the outcome? Do I have a well defined plan?
  - Prepare for change and risk management
- Prepare for CHANGE-Management
  - Stay agile – but remember (trace) why you did or changed something!

Verteilte Web-basierte Systeme – SS 2006

Chapter://2

Project Management

Basics: Project & Tasks

- **Project** –
  - **Temporary endeavor undertaken to create a unique product or service**
  - I.e. set of activities which ends with specific accomplishment and which has
    - (1) Non-routine tasks,
    - (2) Distinct start/finish dates, and
    - (3) Resource constraints (time/money/people/equipment).
- "Tasks" are activities which must be completed to achieve project goal
  - Can have subtasks
  - Have start and end points (and has a duration)
  - Short relative to the project
  - And are significant (Use verb-noun form, e.g. "create design")
  - Use action verbs such as "create", "define" and "gather"

Basics: WBS & Program

- Work Breakdown Statement (WBS)
  - Categorized list of tasks with an estimate of resources required
- Program
  - A group of projects managed in a coordinated way to obtain beneftis not available from managing them individually
  - E.g. Program: PRJ Design, PRJ Construction
  - E.g. Program: PRJ Version 1, PRJ Version 2

Project Problems

- Only ~28% of application projects are delivered **on time**
  - BUT 98% of these with project manager
- Problems to solve:
  - Unrealistic Schedules
    - May yield to artificial documents
  - Avoid cost of iteration/going back a stage
    - Rewriting documents by bad solutions
  - Seriously reviewing and approving takes time
    - Next stage may start before document approved
  - Deliverables not suitable for reuse-oriented models
    - Documents are likely to constrain reuse

PM Disciplines & Skills

- Skills – Usually general management
  - I.e. finance, planning (strategic, tactical, operational), work management, leading vs. managing, delegation, negotiation, etc.
- Steps to start with
  - Define Team, Methods and Standards
  - Define scope and vision of the project → Initiate Phase
  - Risk Management – is a MUST!
  - Report, monitor, review, control etc.
- Many more exist
  - **Project Management Institute Body of Knowledge**, provides over 35 years of experience
  - PM Knowledge Areas and required skills: Integration, Scope, Time, Cost, Quality, HR, Communications, Risk and Procurement Management
  - Cf. PMI's PMBOK http://www.pmibok.com and Standish Group

PM Initiate Phase Activities

Activities to start with in the Initial Phase

- Prepare for Product Life Cycle Management
- Projected Organization and Personnel Management
- Establish open communication in the team
- Advocate for customer vs. Advocate for team vs. …
- Interim Milestone (IM) of Initial Phase:
  - Core Team Organized

M

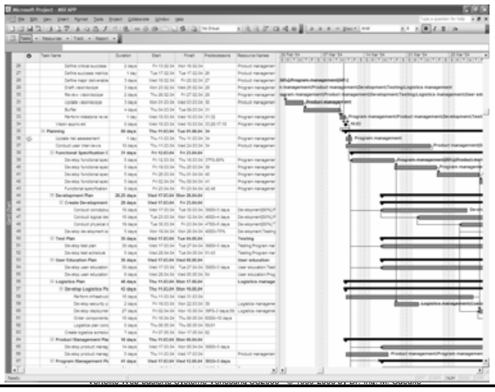
PM Tools

- General Project Management Tools
  - e.g. support for GANTT charts, PERT diagrams)
- Process Models
  - Deliverable-oriented process models
- Dedicated tools
  - Risk management
  - Prediction knowledge bases
  - Requirement tools
- Few (if any) project management research dedicated to Web Application production
  - Requires experience

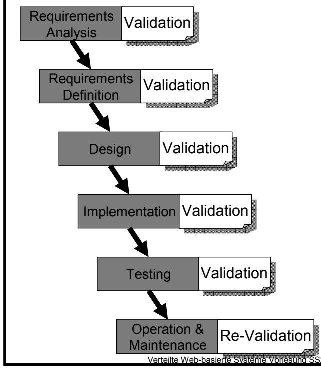
PM-Tool: Gantt Chart

- Gantt Chart - Preferred visual reporting device used for conveying a project's schedule.
  - Graphically displays the work breakdown, total duration needed to complete tasks, as well as %completion
  - Does not display level of effort, and is not an effective planning tool on its own
  - May be integrated with other spreadsheet-type reporting devices that convey additional information related to project planning

PM-Tool: Gantt Chart 2

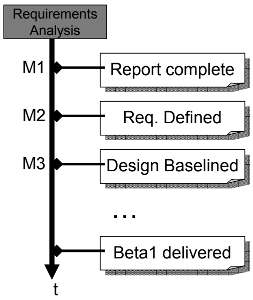


PM-Tool: Deliverables



Requirements Analysis:
- Feasibility study
- Requirements Definition:
  - Req. specification
- Design:
  - Design specification
- Implementation:
  - Web Application code
- Testing:
  - Test result report
- Delivery:
  - Acceptance test / final system
- Operation:
  - Usage report, feedback

PM-Tool: Milestones



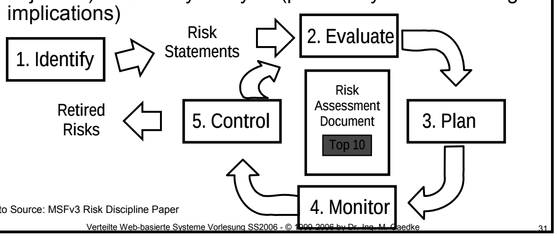
- **Milestones** – Significant event in the project, usually completion of a major deliverable
  - Important checkpoints or interim goals for a project
  - Used to catch scheduling problems early
  - Name by noun-verb form, e.g. "report due", "prototype complete"
- Milestone “Rules of thumb”
  - Too many milestones are useless
  - Focus on hard-results (Not: 80% of Site finished)

Risks & Potential Sources

- Potential sources:
  - **People** – e.g. Customer, Team
  - **Process** – e.g. Schedule, Requirements, Design
  - **Technology** – e.g. Availability, Security
  - **Environmental** – e.g. Legal, Business
- Risk Management is a recurring process throughout the whole project
  - There is no excuse for not doing it…

Risk Management Process

- **Risk** – Any event that could potentially have a negative impact on the project
  - Remember: Risk is not a problem as refers to the future!
  - Evaluate: Quality analysis (prioritize effect on project objective) & Quantity analysis (probability and estimating implications)



Related to Source: MSFv3 Risk Discipline Paper

Risk Assessment (simplified)

Risk	**Probability (1-7)**	**Impact (1-7)**	**Total Risk (P x I)**
Lead Developer leaves team	2	6	**12**
Regulation §1-3	2	7	**14**
WS not available	4	5	**20**

Organization Structure

Will not be covered by this lecture

Some notes:

- Few (if any) Web Engineering Research
- Cf. standard Software Engineering literature
- Regarding evolution of content
  - Workflows respectively process models for publishing content and people involved
  - Cf. Content Management Systems literature

Costs Estimation

Will not be covered by this lecture

Some notes:

- Few Web Engineering Research
- Cf. standard Software Engineering literature
- Issues to look at
  - Costs for Marketing (search engines, advertising)
  - Return on investment (ROI) have to be considered, e.g. advertisements, integration of other Web Applications (Marketplaces, Web Application Production Lines)
  - Computer & Network costs
  - Political influences

Verteilte Web-basierte Systeme – SS 2006

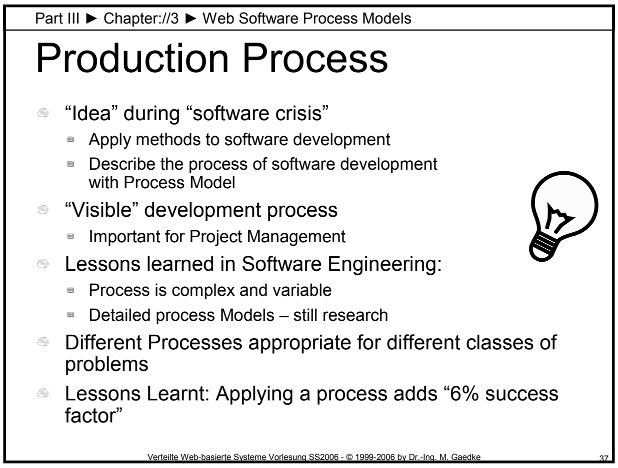
Chapter://3

Web Software Process Models

Goals of Web Engineering

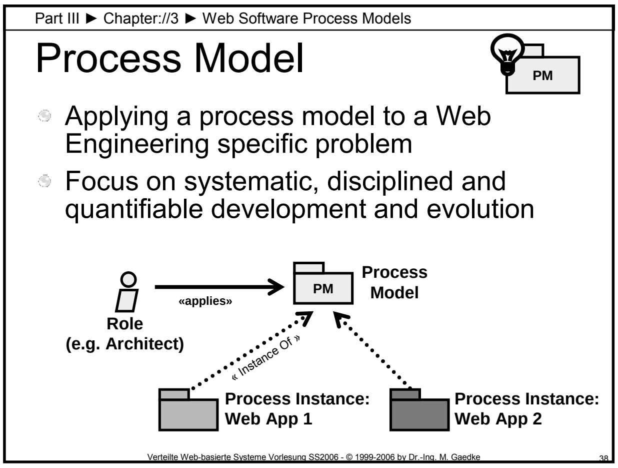
- Develop (high quality) Web Applications
  - Effective
  - Efficient
  - Achieve desired application
  - in a Predictable Way
- **Maintain and Evolve**
  - **Plan for change** (Solution may change the problem!!!)

…using systematic, disciplined and quantifiable Approaches: Process Models



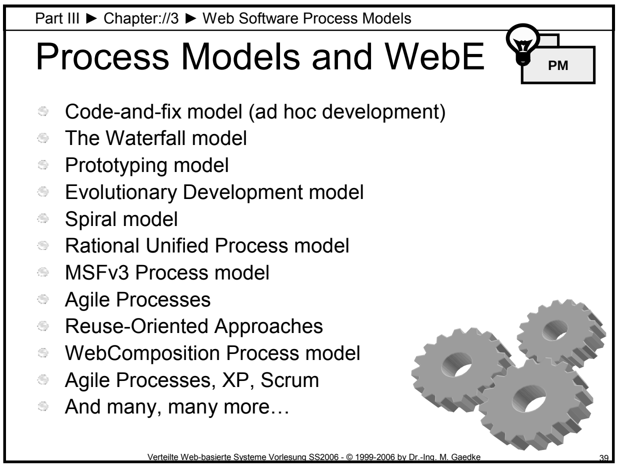
Production Process

- "Idea" during "software crisis"
  - Apply methods to software development
  - Describe the process of software development with Process Model
- "Visible" development process
  - Important for Project Management
- Lessons learned in Software Engineering:
  - Process is complex and variable
  - Detailed process Models – still research
- Different Processes appropriate for different classes of problems
- Lessons Learnt: Applying a process adds "6% success factor"



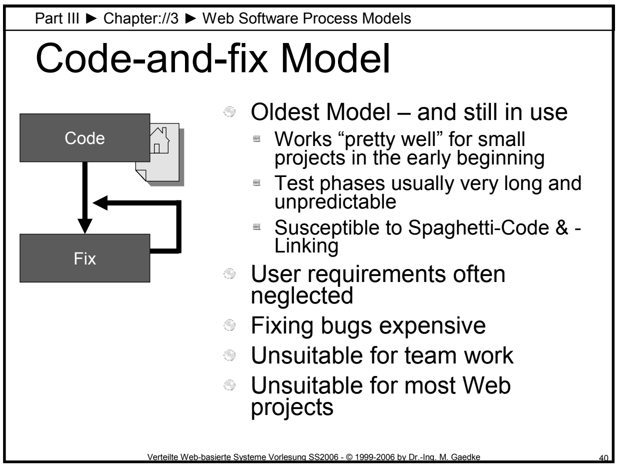
Process Model

- Applying a process model to a Web Engineering specific problem
- Focus on systematic, disciplined and quantifiable development and evolution



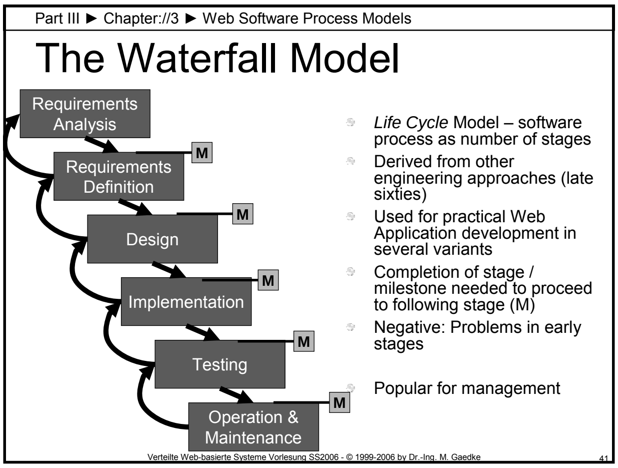
Process Models and WebE

- Code-and-fix model (ad hoc development)
- The Waterfall model
- Prototyping model
- Evolutionary Development model
- Spiral model
- Rational Unified Process model
- MSFv3 Process model
- Agile Processes
- Reuse-Oriented Approaches
- WebComposition Process model
- Agile Processes, XP, Scrum
- And many, many more…



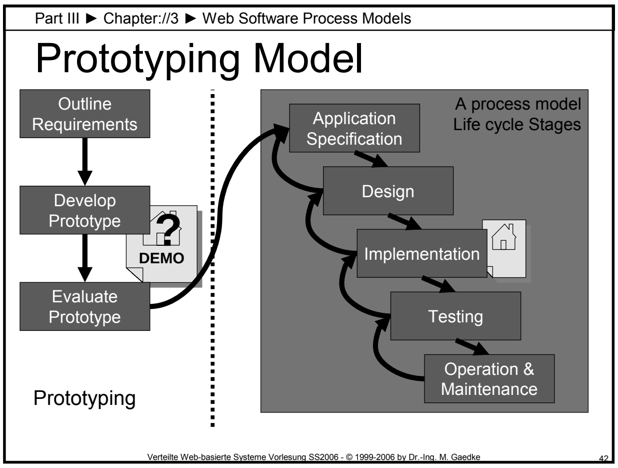
Code-and-fix Model

- Oldest Model – and still in use
  - Works "pretty well" for small projects in the early beginning
  - Test phases usually very long and unpredictable
  - Susceptible to Spaghetti-Code & -Linking
- User requirements often neglected
- Fixing bugs expensive
- Unsuitable for team work
- Unsuitable for most Web projects

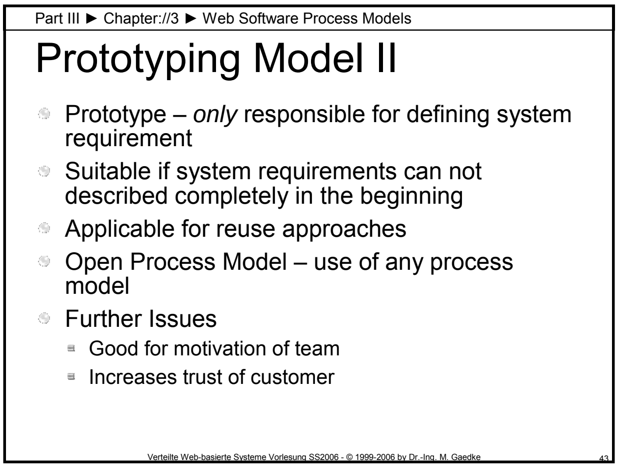


The Waterfall Model

- *Life Cycle* Model – software process as number of stages
- Derived from other engineering approaches (late sixties)
- Used for practical Web Application development in several variants
- Completion of stage / milestone needed to proceed to following stage (M)
- Negative: Problems in early stages
- Popular for management

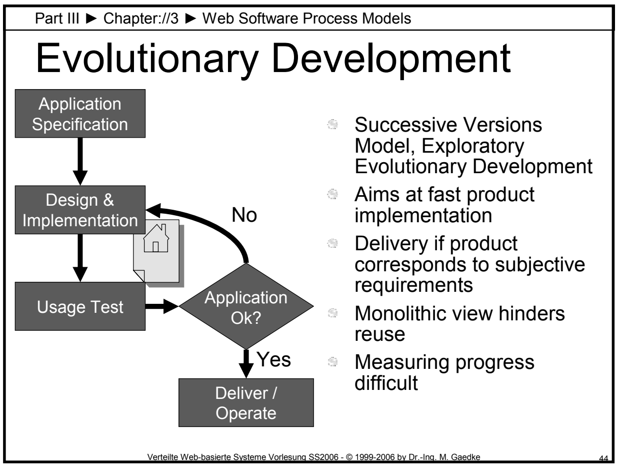


Prototyping Model



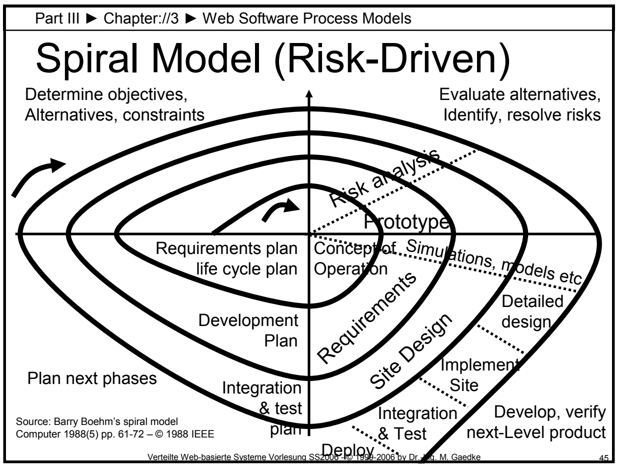
Prototyping Model II

- Prototype – *only* responsible for defining system requirement
- Suitable if system requirements can not described completely in the beginning
- Applicable for reuse approaches
- Open Process Model – use of any process model
- Further Issues
  - Good for motivation of team
  - Increases trust of customer



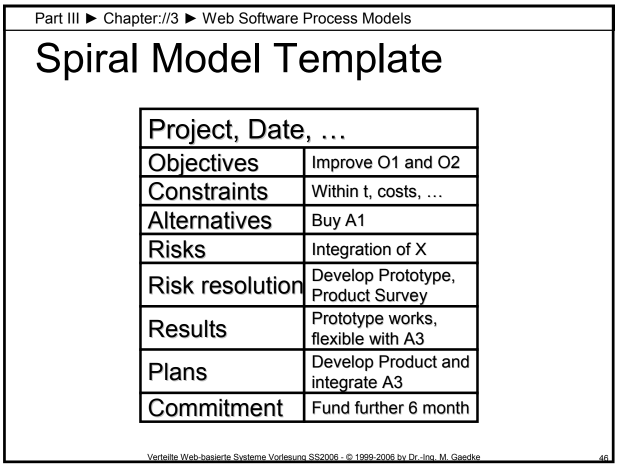
Evolutionary Development

- Successive Versions Model, Exploratory Evolutionary Development
- Aims at fast product implementation
- Delivery if product corresponds to subjective requirements
- Monolithic view hinders reuse
- Measuring progress difficult



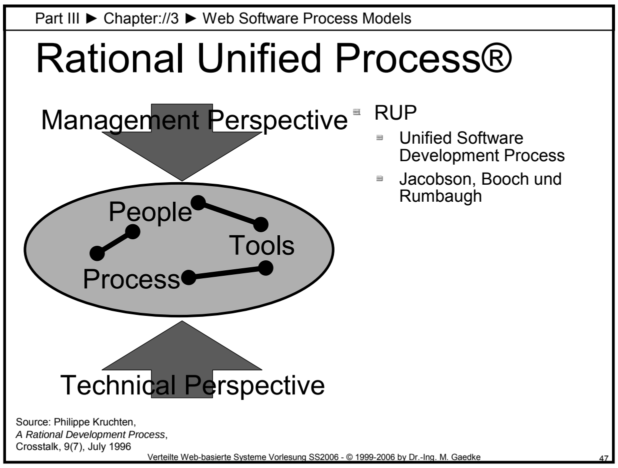
Spiral Model (Risk-Driven)

Source: Barry Boehm's spiral model
Computer 1988(5) pp. 61-72 – © 1988 IEEE



Spiral Model Template

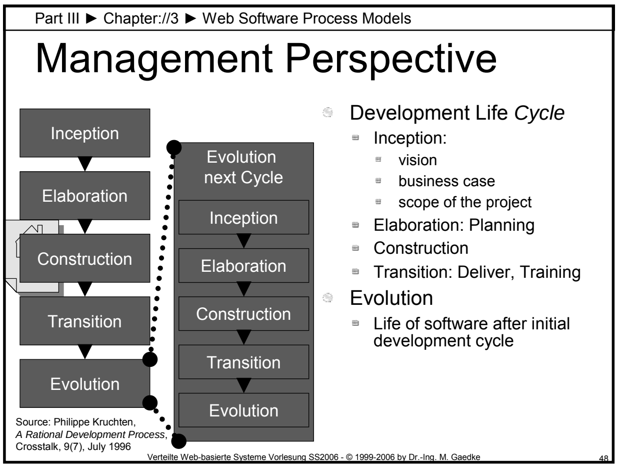
Project, Date, …	
Objectives	Improve O1 and O2
Constraints	Within t, costs, …
Alternatives	Buy A1
Risks	Integration of X
Risk resolution	Develop Prototype, Product Survey
Results	Prototype works, flexible with A3
Plans	Develop Product and integrate A3
Commitment	Fund further 6 month



Rational Unified Process®

- RUP
  - Unified Software Development Process
  - Jacobson, Booch und Rumbaugh

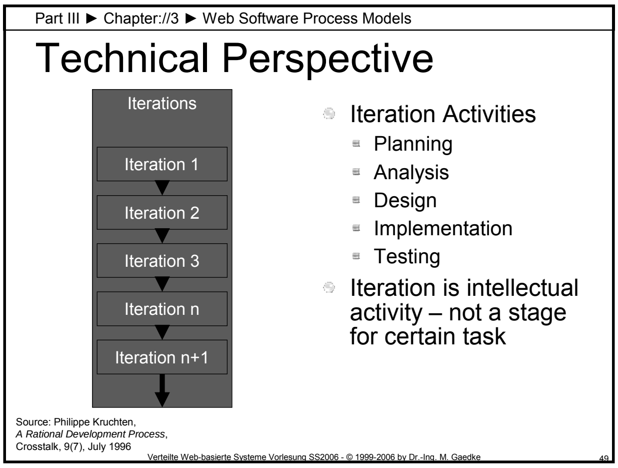
Source: Philippe Kruchten,
A Rational Development Process,
Crosstalk, 9(7), July 1996



Management Perspective

- Development Life *Cycle*
  - Inception:
    - vision
    - business case
    - scope of the project
  - Elaboration: Planning
  - Construction
  - Transition: Deliver, Training
- Evolution
  - Life of software after initial development cycle

Source: Philippe Kruchten,
A Rational Development Process,
Crosstalk, 9(7), July 1996



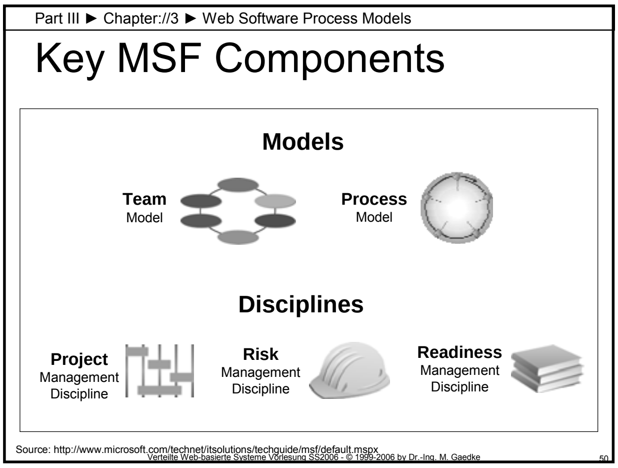
Part III ► Chapter://3 ► Web Software Process Models

Technical Perspective

Iterations
Iteration 1
Iteration 2
Iteration 3
Iteration n
Iteration n+1

- Iteration Activities
  - Planning
  - Analysis
  - Design
  - Implementation
  - Testing
- Iteration is intellectual activity – not a stage for certain task

Source: Philippe Kruchten,
A Rational Development Process,
Crosstalk, 9(7), July 1996

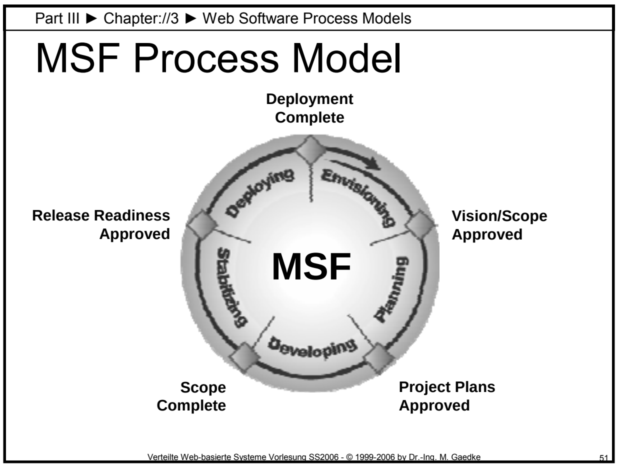


Part III ► Chapter://3 ► Web Software Process Models

Key MSF Components

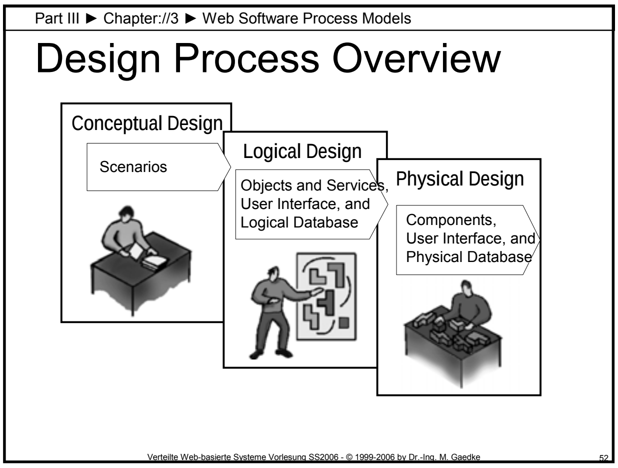
Models
Team Model
Process Model
Disciplines
Project Management Discipline
Risk Management Discipline
Readiness Management Discipline

Source: http://www.microsoft.com/technet/itsolutions/techguide/msf/default.mspx



Part III ► Chapter://3 ► Web Software Process Models

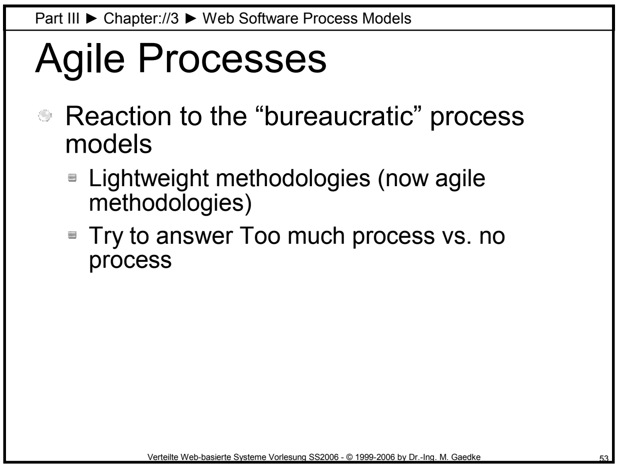
MSF Process Model



Part III ► Chapter://3 ► Web Software Process Models

Design Process Overview

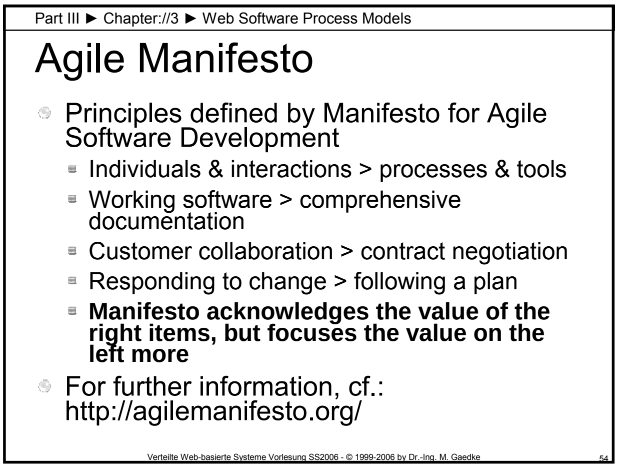
Conceptual Design
Scenarios
Logical Design
Objects and Services, User Interface, and Logical Database
Physical Design
Components, User Interface, and Physical Database



Part III ► Chapter://3 ► Web Software Process Models

Agile Processes

- Reaction to the “bureaucratic” process models
  - Lightweight methodologies (now agile methodologies)
  - Try to answer Too much process vs. no process



Part III ► Chapter://3 ► Web Software Process Models

Agile Manifesto

- Principles defined by Manifesto for Agile Software Development
  - Individuals & interactions > processes & tools
  - Working software > comprehensive documentation
  - Customer collaboration > contract negotiation
  - Responding to change > following a plan
  - **Manifesto acknowledges the value of the right items, but focuses the value on the left more**
- For further information, cf.: http://agilemanifesto.org/

Agile (Process) Findings

- Separation of design and construction
  - Construction is automated by the compiler
  - … all the effort is design (this includes coding)
  - Design is a creative process …
  - as such: not easy to plan, predictability is impossible
- Iterative development is essential
  - Allows to deal with changes in required features
- Style of planning
  - Long term plans: fluid
  - Short term plans: stable for a single iteration

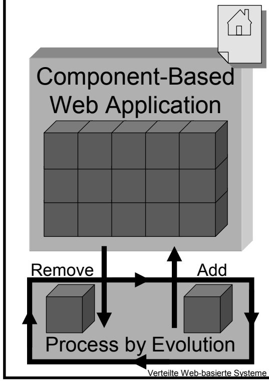
Agile (Process) Findings - 2

- Duration for an iteration
  - In general as short as possible (depends: customer & developer)
  - E.g. XP - between one and three weeks
  - E.g. SCRUM - a month
- Accepting the process rather than imposition of a process
  - Accepting a process requires **commitment**
  - I.e. empowers development team
  - I.e. everyone in team equal place in leadership

Agile Methodologies

- XP (Extreme Programming)
  - Testing as foundation of development
  - Write tests first
  - Evolutionary design process with focus on refactoring
- Cockburn's Crystal Family
  - Different projects require different methodologies
  - Focus on least disciplined methodology (that could still succeed)
  - Iteration reviews encouraging the process to be self-improving
- Scrum
  - Iteration = Sprints (of 30 days)
  - Scrum = Every day fifteen minute meeting
- Feature Driven Development (FDD)
  - Start: Develop an Overall Model, Build a Features List, Plan by Feature
  - Iteration: Design by Feature, Build by Feature
- Many others
  - RUP and MSF can be used in an agile manner – but don't have to!

Reuse-Oriented Approaches



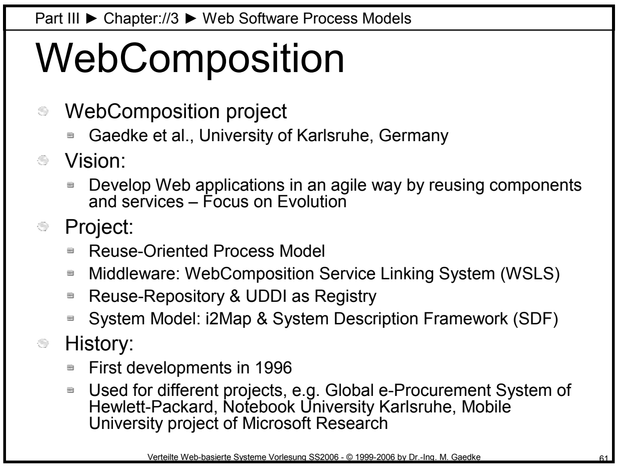
- Web Engineering in context of Reuse
- Product is assembly from *reusable components*
  - Idea: All needed Components exist
- These Approaches focus on being **agile** in the context of:
  - **Producer Reuse**
  - **Consumer Reuse**

Model for Producer Reuse

- *Develop (Components)* ***for Reuse***
- Domain Engineering
  - Process Model for Production of Domain Components
- Process Model
  1. Domain Analysis
  2. Develop Components
  3. Deploy Components in Reuse-Repository
  4. Make them available via Registries

Model for Consumer Reuse

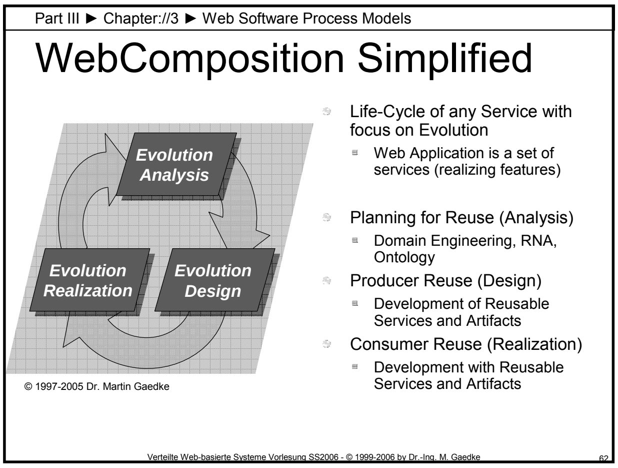
- *Develop* ***with Reuse*** *(of Components)*
- Orthogonal Process Model
- Process Model
  1. Accessing
  2. Understanding
  3. Adapting



Part III ► Chapter://3 ► Web Software Process Models

WebComposition

- WebComposition project
  - Gaedke et al., University of Karlsruhe, Germany
- Vision:
  - Develop Web applications in an agile way by reusing components and services – Focus on Evolution
- Project:
  - Reuse-Oriented Process Model
  - Middleware: WebComposition Service Linking System (WSLS)
  - Reuse-Repository & UDDI as Registry
  - System Model: i2Map & System Description Framework (SDF)
- History:
  - First developments in 1996
  - Used for different projects, e.g. Global e-Procurement System of Hewlett-Packard, Notebook University Karlsruhe, Mobile University project of Microsoft Research

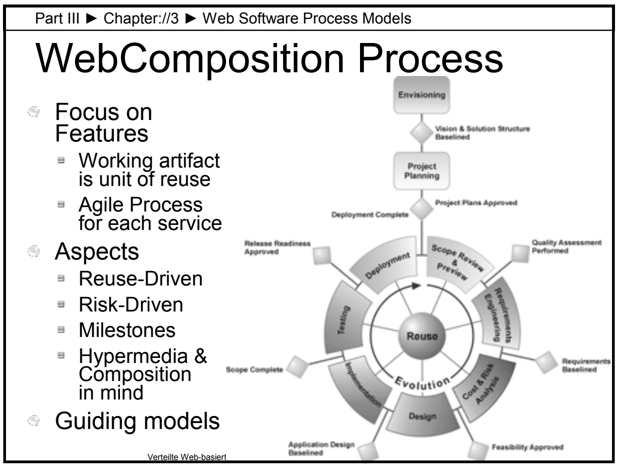


Part III ► Chapter://3 ► Web Software Process Models

WebComposition Simplified

© 1997-2005 Dr. Martin Gaedke

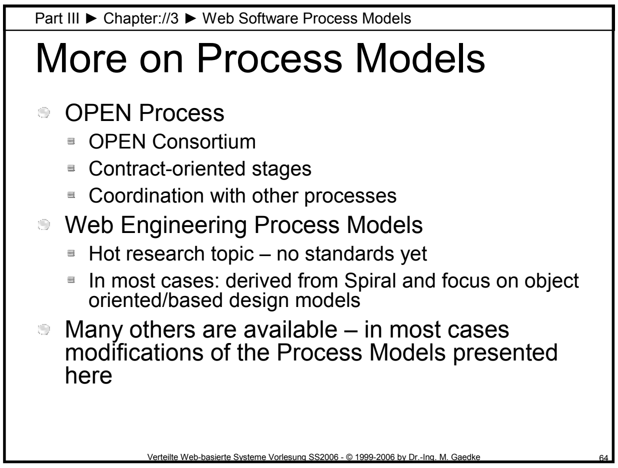
- Life-Cycle of any Service with focus on Evolution
  - Web Application is a set of services (realizing features)
- Planning for Reuse (Analysis)
  - Domain Engineering, RNA, Ontology
- Producer Reuse (Design)
  - Development of Reusable Services and Artifacts
- Consumer Reuse (Realization)
  - Development with Reusable Services and Artifacts



Part III ► Chapter://3 ► Web Software Process Models

WebComposition Process

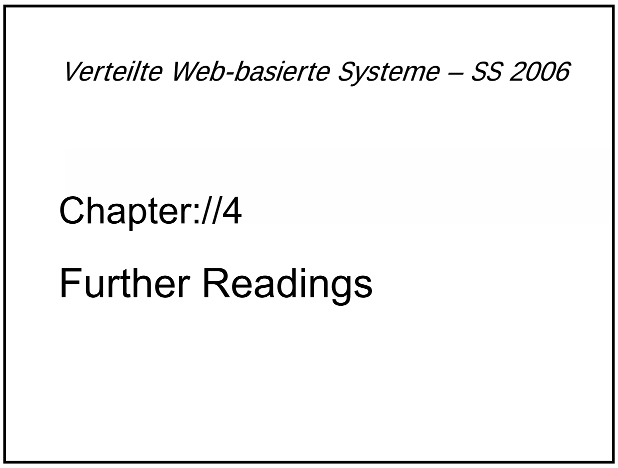
- Focus on Features
  - Working artifact is unit of reuse
  - Agile Process for each service
- Aspects
  - Reuse-Driven
  - Risk-Driven
  - Milestones
  - Hypermedia & Composition in mind
- Guiding models



Part III ► Chapter://3 ► Web Software Process Models

More on Process Models

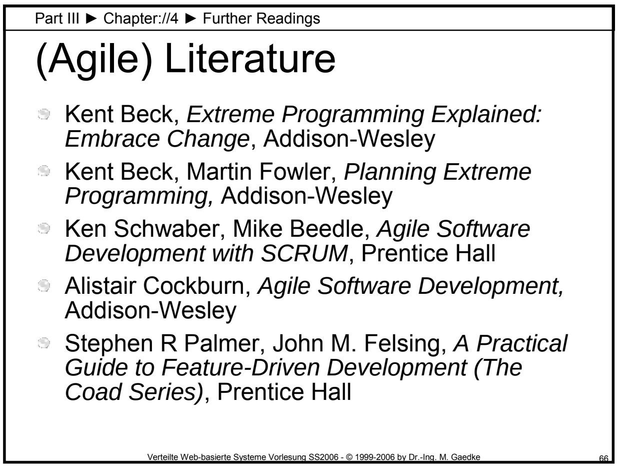
- OPEN Process
  - OPEN Consortium
  - Contract-oriented stages
  - Coordination with other processes
- Web Engineering Process Models
  - Hot research topic – no standards yet
  - In most cases: derived from Spiral and focus on object oriented/based design models
- Many others are available – in most cases modifications of the Process Models presented here



Verteilte Web-basierte Systeme – SS 2006

Chapter://4

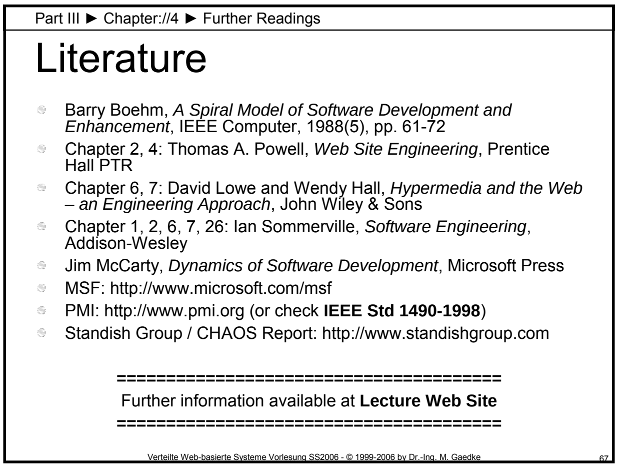
Further Readings



Part III ► Chapter://4 ► Further Readings

(Agile) Literature

- Kent Beck, *Extreme Programming Explained: Embrace Change*, Addison-Wesley
- Kent Beck, Martin Fowler, *Planning Extreme Programming,* Addison-Wesley
- Ken Schwaber, Mike Beedle, *Agile Software Development with SCRUM*, Prentice Hall
- Alistair Cockburn, *Agile Software Development,* Addison-Wesley
- Stephen R Palmer, John M. Felsing, *A Practical Guide to Feature-Driven Development (The Coad Series)*, Prentice Hall



Part III ► Chapter://4 ► Further Readings

Literature

- Barry Boehm, *A Spiral Model of Software Development and Enhancement*, IEEE Computer, 1988(5), pp. 61-72
- Chapter 2, 4: Thomas A. Powell, *Web Site Engineering*, Prentice Hall PTR
- Chapter 6, 7: David Lowe and Wendy Hall, *Hypermedia and the Web – an Engineering Approach*, John Wiley & Sons
- Chapter 1, 2, 6, 7, 26: Ian Sommerville, *Software Engineering*, Addison-Wesley
- Jim McCarty, *Dynamics of Software Development*, Microsoft Press
- MSF: http://www.microsoft.com/msf
- PMI: http://www.pmi.org (or check **IEEE Std 1490-1998**)
- Standish Group / CHAOS Report: http://www.standishgroup.com

======================================

Further information available at **Lecture Web Site**

======================================

Verteilte Web-basierte Systeme Vorlesung SS2006 - © 1999-2006 by Dr.-Ing. M. Gaedke 67