Software Process Engineering & Management Models

Paul Grünbacher

Institute for Systems Engineering & Automation
Johannes Kepler University Linz

Christian Doppler Laboratory for Automated Software Engineering
Johannes Kepler University Linz
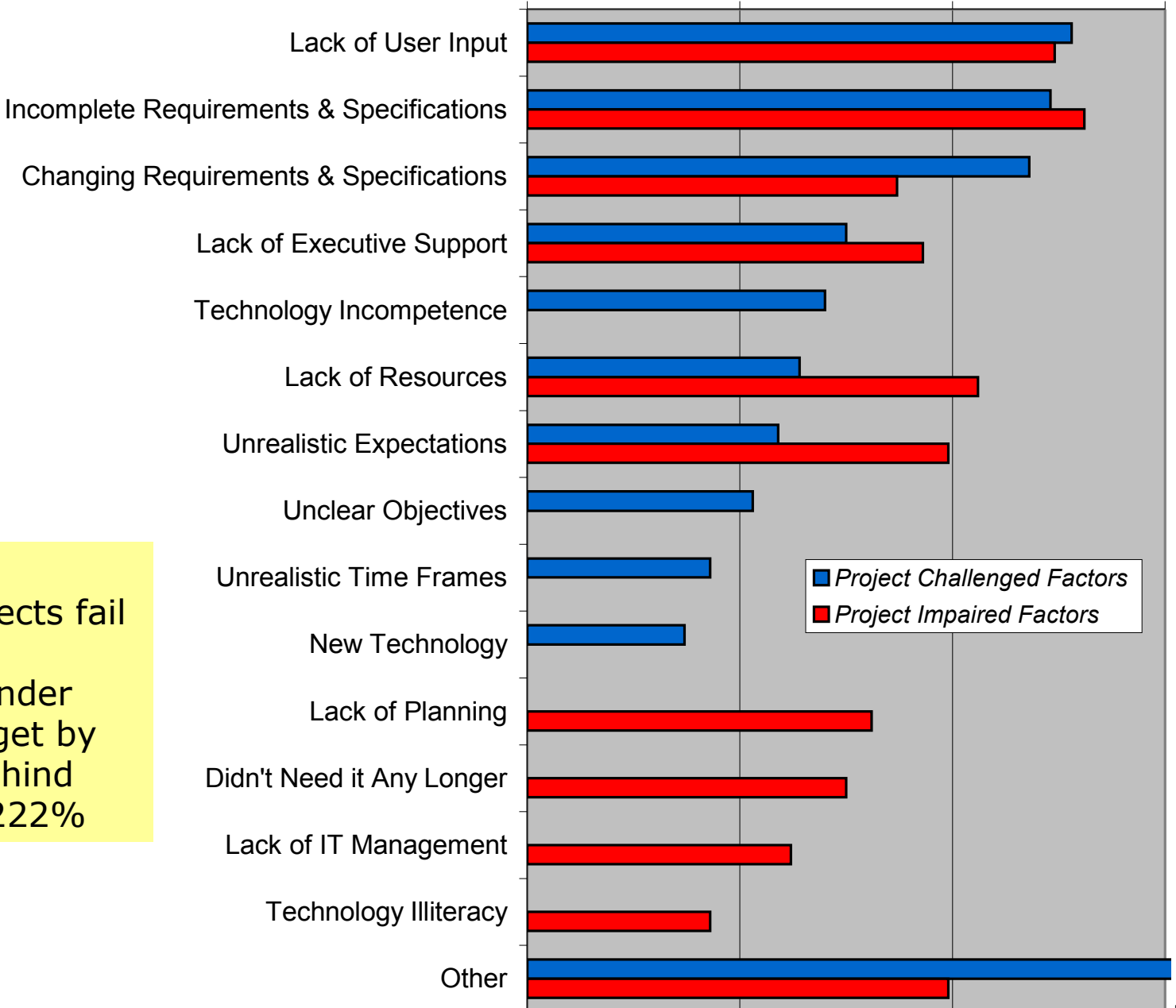
Email: paul.gruenbacher@jku.at

Overview

- *Part I:* Software engineering challenges
  - Some surveys
- *Part II:* WinWin-Spiral Model Principles
  - Iterative
  - Risk-driven
  - Stakeholder involvement
  - Life-cycle anchor points
  - Emphasis on system activities and artifacts
- *Part III:* Practices and examples
  - ISO 15504 („SPICE")
  - IBM Rational Unified Process
  - EasyWinWin
  - Agile Methods
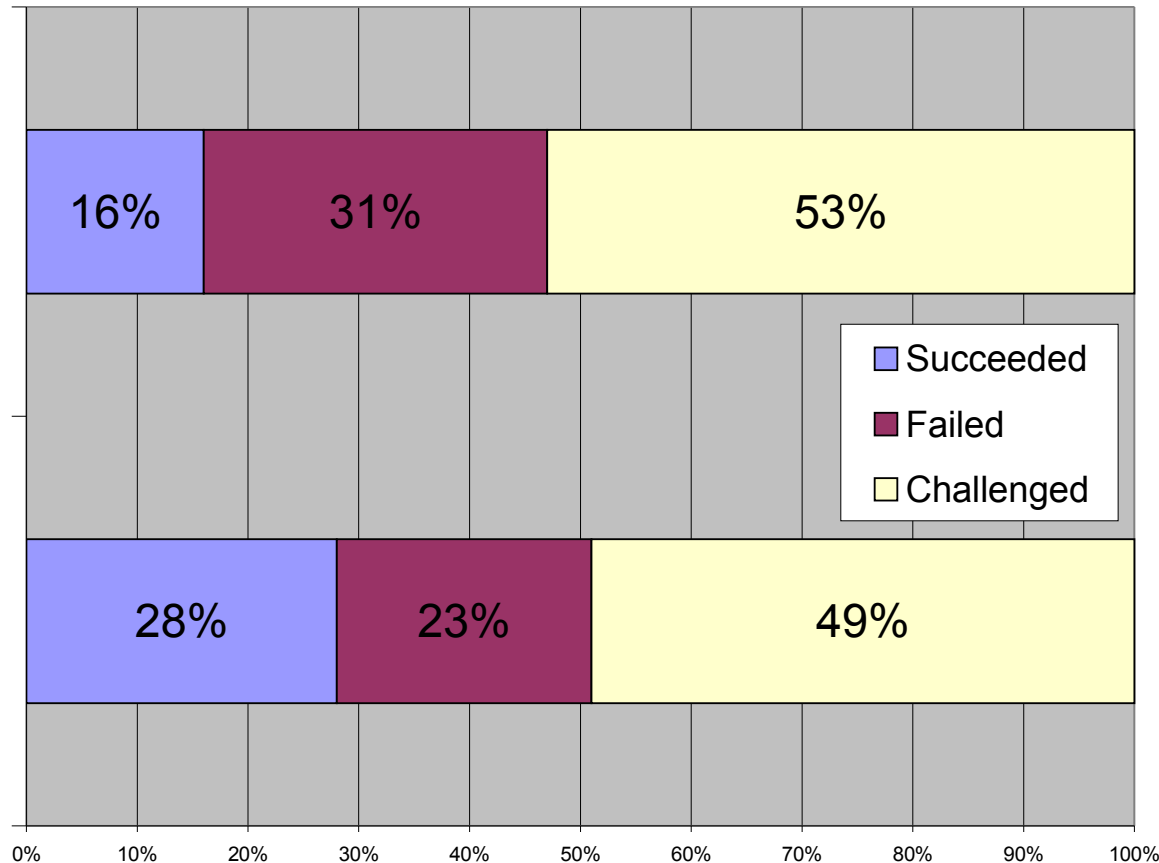- Summary

Project Challenges

0% 5% 10% 15%

Challenge	
Lack of User Input	
Incomplete Requirements & Specifications	
Changing Requirements & Specifications	
Lack of Executive Support	
Technology Incompetence	
Lack of Resources	
Unrealistic Expectations	
Unclear Objectives	
Unrealistic Time Frames	
New Technology	
Lack of Planning	
Didn't Need it Any Longer	
Lack of IT Management	
Technology Illiteracy	
Other	

Legend:
- ■ Project Challenged Factors
- ■ Project Impaired Factors

30% of software development projects fail

70% of the remainder are over budget by 189% and behind schedule by 222%

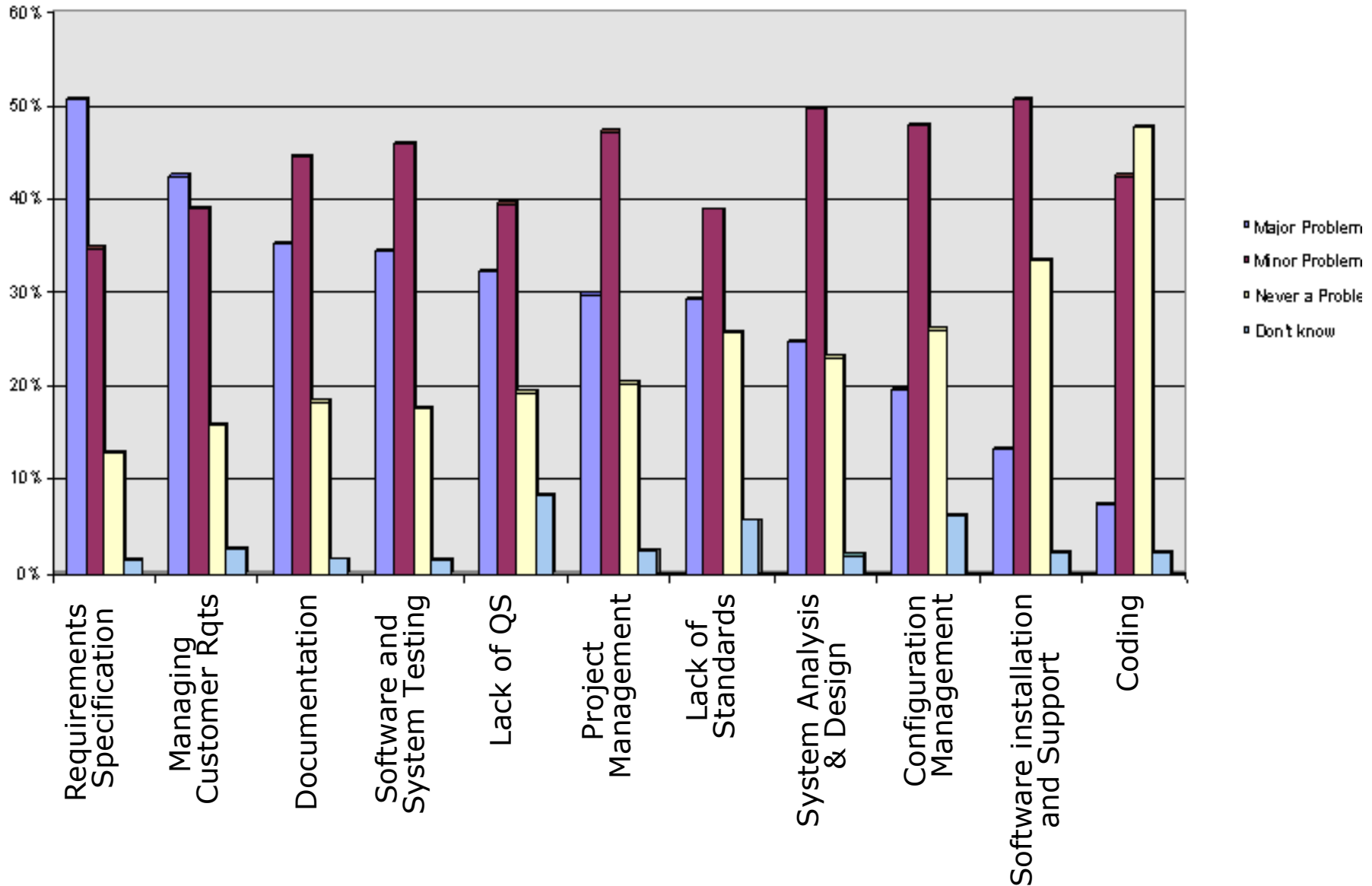*Source: 352 US Companies, 8000 Projects, The Standish Group, 1994*

3

CHAOS Top Success Factors 2000

- Executive support
- User involvement
- Experienced project manager
- Clear business objectives
- Minimized Scope
- Standard software infrastructure

16%	31%	53%
28%	23%	49%

Legend:
- Succeeded
- Failed
- Challenged

X-axis: 0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%

4

Perceived Relative Importance of SW Problems in Europe

Legend:
- Major Problem
- Minor Problem
- Never a Problem
- Don't know

Categories (left to right):
- Requirements Specification
- Managing Customer Rqts
- Documentation
- Software and System Testing
- Lack of QS
- Project Management
- Lack of Standards
- System Analysis & Design
- Configuration Management
- Software installation and Support
- Coding

"If You Don't Actively Attack the Risks,

The Risks Will Actively Attack You."

-Tom Gilb

4

Iterative vs waterfall process

- Attack risks through demonstrable progress

- Continuous integration

- Frequent, executable releases

- Continuous stakeholder involvement

**Progress
(% coded)**

Time (months)
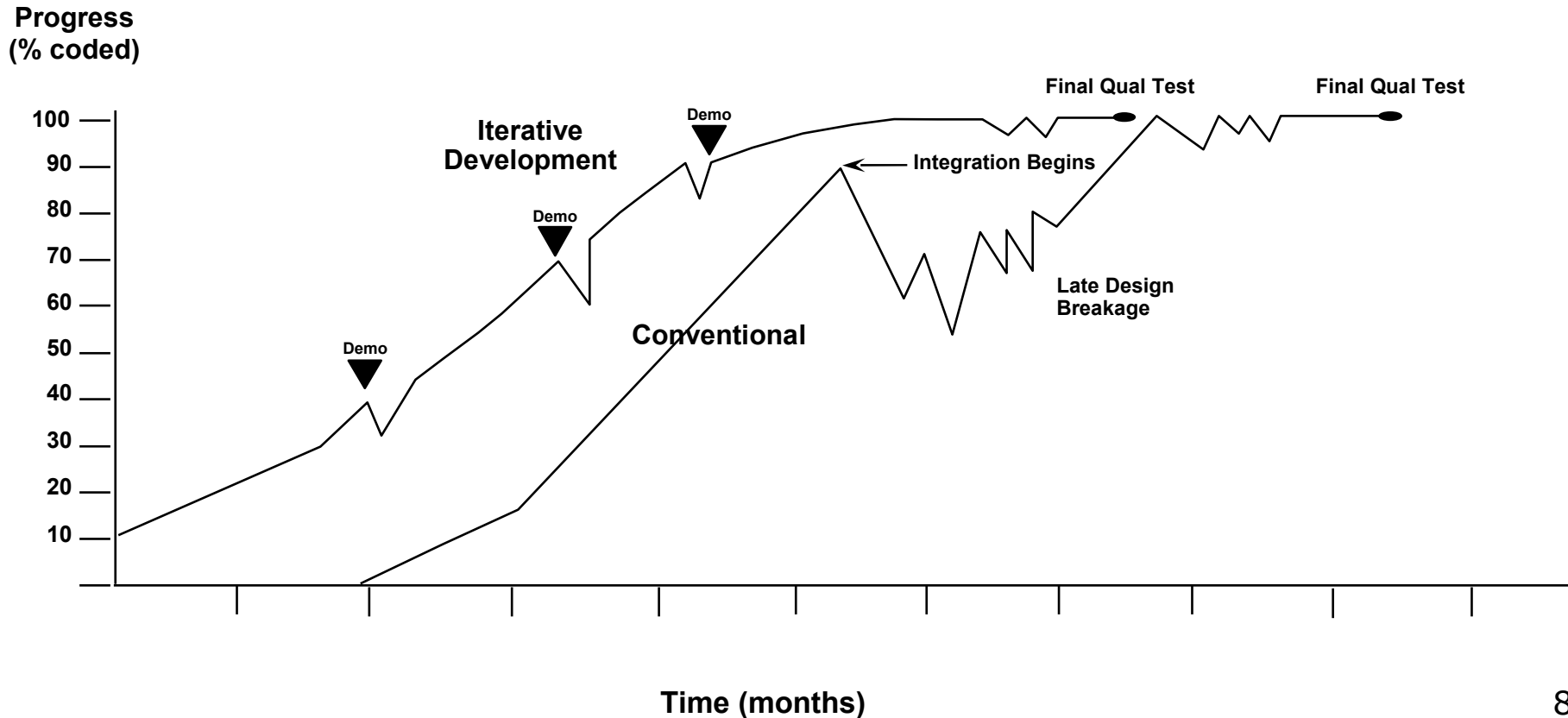
The WinWin Spiral Model:
A stakeholder & risk-driven process generator

WinWin Extension to the Spiral Model

1. Identify next-level Stakeholders

2. Identify Stakeholders' win conditions

3a. Reconcile win conditions

3b. Establish next level objectives, constraints, alternatives

4. Evaluate product and process alternatives. Resolve Risks

5. Define next level of product and process – including partitions

6. Validate product and process definitions

7. Review, commitment

Original Spiral Model

Barry W. Boehm, "A Spiral Model of Software Development and Enhancement," IEEE Computer, 5/1988.
Barry W. Boehm et al., "Using the Win Win Spiral Model: A Case Study," IEEE Computer, 7/1998.

Win-Win Spiral Model: Key Principles

- Iterative rather than sequential
  - Requirements, plans, designs, code, etc. evolve concurrently
- Risk-driven
  - Risks determine course of action
  - Risks determine level of effort on activities and level of detail of artifacts („how much is enough?")
- Involvement of success-critical stakeholders
  - Understand objectives, constraints, alternatives in each cycle
  - WinWin/EasyWinWin
- Life-Cycle anchor points
  - Intermediate commitment points (LCO, LCA, IOC)
- Emphasis on system activities/artifacts
  - Understand the context
  - Avoids premature suboptimization on hardware, software, or development considerations

Erfolgsfaktoren komplexer IKT-Projekte der öffentlichen Hand

Software Process Map

Level of Detail (y-axis)

Atomic Level
tools
techniques

Worldly Level
Practices that guide daily work

Universal Level

general guidelines
principles
frameworks

EasyWinWin

Unified Process (RUP)

XP

ISO 15504

Spiral Model

Context determines the applicability of approach

Tailoring is crucial

Specific Process Area   *Project*   *Organization*   **Scope**
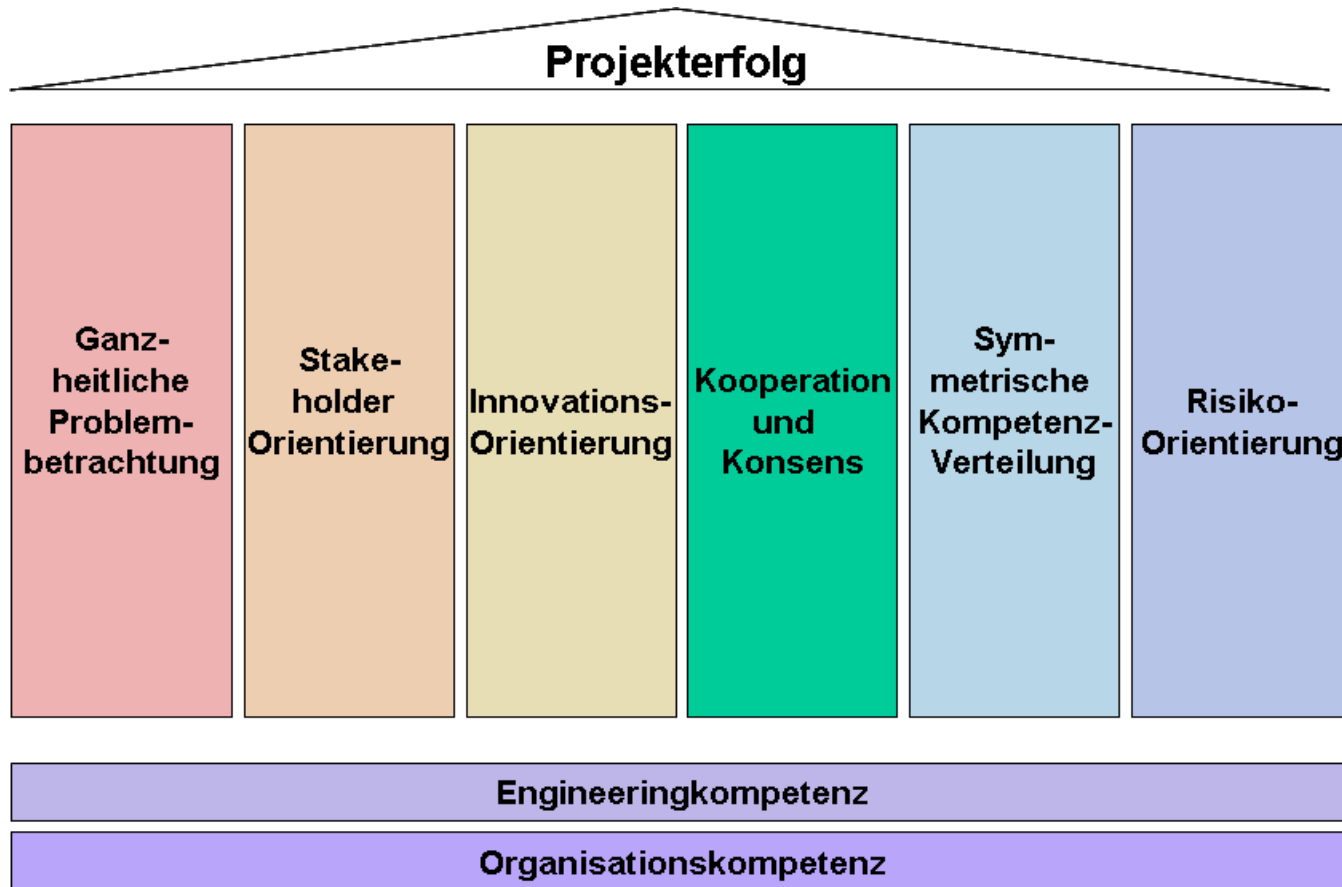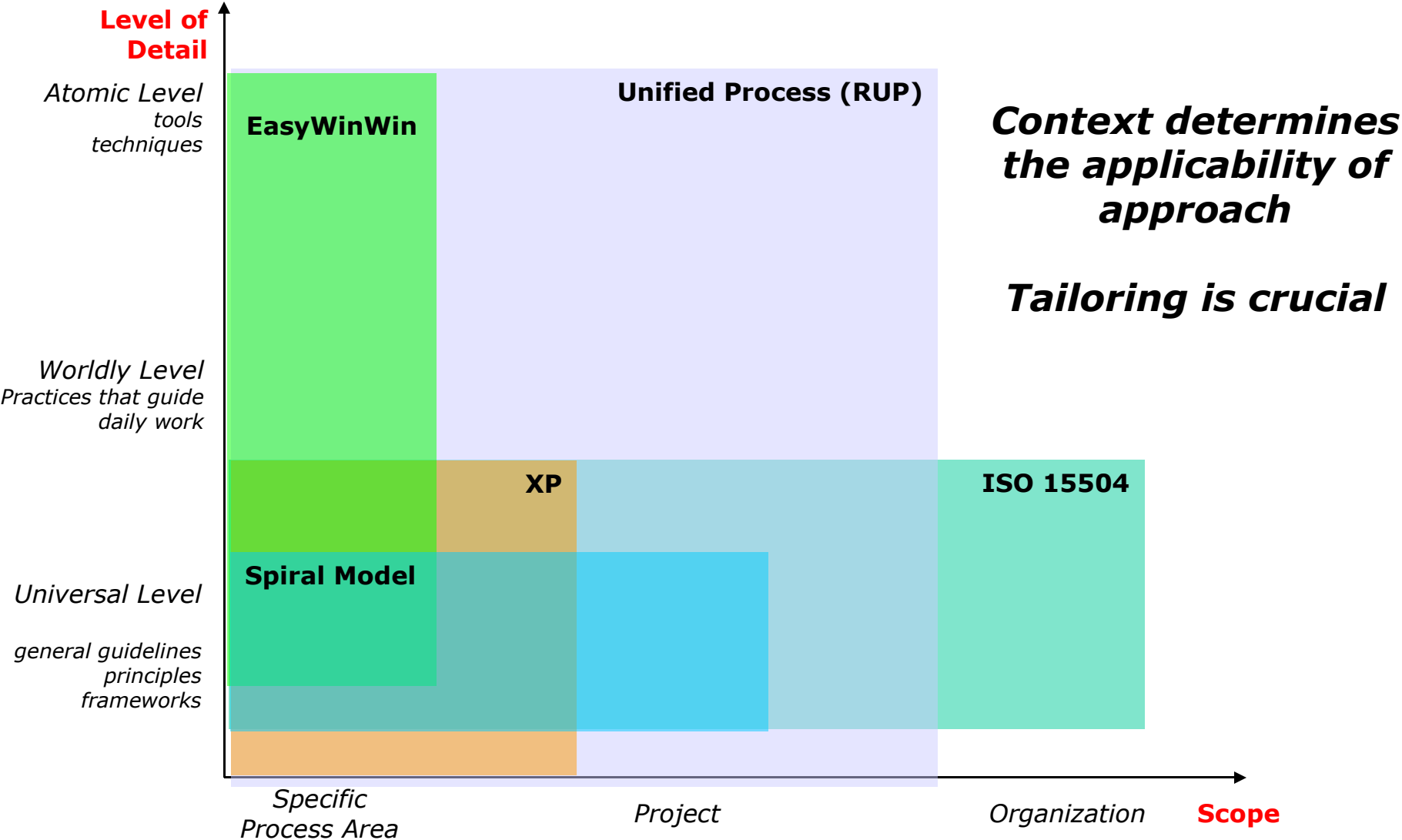
ISO 15504 Process Architecture (TR98)

Customer-Supplier

CUS.1   Acquisition
CUS.1.1   Acquisition Preparation
CUS.1.2   Supplier Selection
CUS.1.3   Supplier Monitoring
CUS.1.4   Customer Acceptance

CUS.2   Supply

CUS.3   Requirements Elicitation

CUS.4   Operation
CUS.4.1   Operational Use
CUS.4.2   Customer Support

Engineering

ENG.1   Development
ENG.1.1   System Requirements Analysis & Design
ENG.1.2   Software Requirements Analysis
ENG.1.3   Software Design
ENG.1.4   Software Construction
ENG.1.5   Software Integration
ENG.1.6   Software Testing
ENG.1.7   System Integration & Testing

ENG.2   System & Software Maintenance

Support

SUP.1   Documentation
SUP.2   Configuration Management
SUP.3   Quality Assurance
SUP.4   Verification
SUP.5   Validation
SUP.6   Joint Reviews
SUP.7   Audit
SUP.8   Problem Resolution

Management

MAN.1   Management
MAN.2   Project Management
MAN.3   Quality Management
MAN.4   Risk Management

Organisation

ORG.1   Organisational Alignment
ORG.2   Improvement
ORG.2.1   Process Establishment
ORG.2.2   Process Assessment
ORG.2.3   Process Improvement
ORG.3   Human Resource Management
ORG.4   Infrastructure
ORG.5   Measurement
ORG.6   Reuse

Capability Levels, Process Attributes

Quantitative measures used for continuous improvement.

Level 5   Optimising

PA.5.1   **Process Change**

PA.5.2   **Continuous Improvement**

Metrics make process performance and results controllable.

Level 4   Predictable

PA.4.1   **Measurement**

PA.4.2   **Process Control**

Predefined processes are tailored for specific use, resources are managed.

Level 3   Established

PA.3.1   **Process Definition**

PA.3.2   **Process Resource**

Level 2   Managed

PA.2.1   **Performance Management**

PA.2.2   **Work Product Management**

Process and work products are managed, responsibilities identified.

Level 1   Performed

PA.1.1   **Process Performance**

Processes are intuitively performed,
input and output work products are available.

Level 0   Incomplete

Performance and results are incomplete, ad-hoc processes

Sample Process Profile

	Capability Level 1	Capability Level 2	Capability Level 3
CUS.1 Acquisition Process			
CUS.1.1 Acquisition Preparation Process	▉		
CUS.1.2 Supplier Selection Process	▉		▉
CUS.1.3 Supplier Monitoring Process	▉	▉	
CUS.1.4 Customer Acceptance Process	▉	▉	
CUS.2 Supply Process	▉	▉	
CUS.3 Requirements Elicitation Process	▉	▉	▉
CUS.4 Operation Process	▉	▉	
CUS.4.1 Operational Use Process	▉	▉	
CUS.4.2 Customer Support Process	▉		
ENG.1 Development Process	▉	▉	
ENG.1.1 System Requirements Analysis and Design Process	▉		
ENG.1.2 Software Requirements Analysis Process			
ENG.1.3 Software Design Process	▉		
ENG.1.4 Software Construction Process	▉	▉	▉
ENG.1.5 Software Integration Process	▉	▉	
ENG.1.6 Software Testing Process			
ENG.1.7 System Integration and Testing Process			
ENG.2 System and Software Maintenance Process	▉	▉	▉
SUP.1 Documentation Process	▉		
SUP.2 Configuration Management Process	▉		
SUP.3 Quality Assurance Process	▉		
SUP.4 Verification Process	▉	▉	

15

Where am I

Getting Started | Search | Glossary | Index | Print | Feedback

Overview

- Overview
- Phases
- Disciplines
- Roles and Activities
  - Analyst Role Set
  - Developer Role Set
  - Tester Role Set
  - Manager Role Set
  - Additional Role Set
- Artifacts
  - Business Modeling A
  - Requirements Artifac
  - Analysis & Design An
    - Data Model
    - Design Model...
    - Software Architect
    - {More Analysis & D
  - Implementation Artifa
  - Test Artifact Set
  - Deployment Artifact S
  - Configuration & Chan
  - Project Management
  - Environment Artifact S
- Tool Mentors
- Templates
- White Papers
- Work Guidelines
- Examples Overview

Display Treebrowser

Rational Unified Process: Overview

Artifacts | Examples | Roles | Roadmaps | Site Map
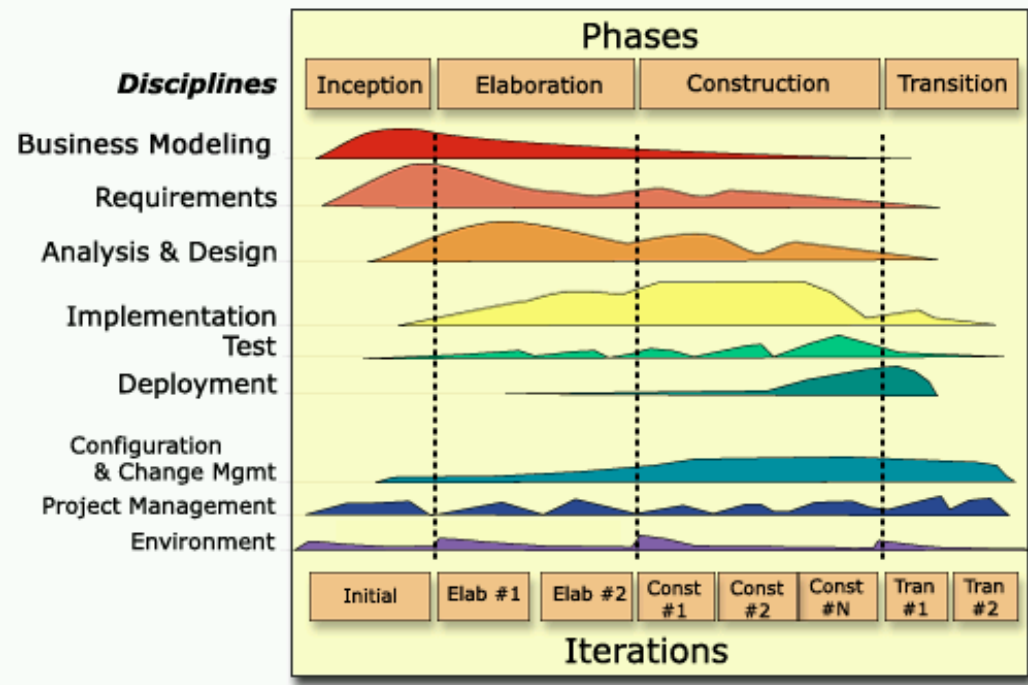
Click on an area of the screen for more information.

Involving Success-critical Stakeholders: The EasyWinWin approach

- Elicit and understand objectives, constraints, alternatives of success-critical stakeholders
  - Required in each cycle of the WinWin Spiral Model
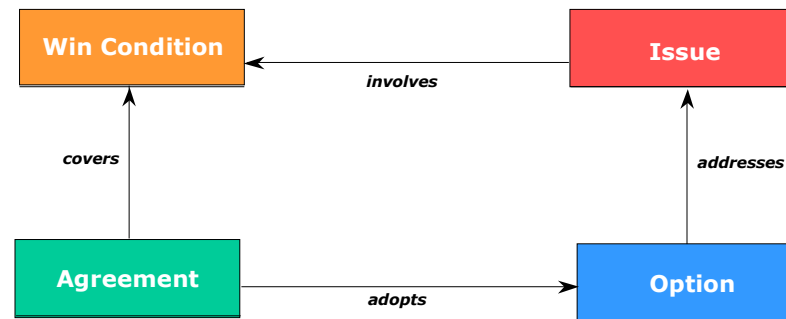
- Detailed Requirements Negotiation Process
  - Elicit, organize, prioritize, negotiate stakeholder objectives
  - Attain consensus among stakeholders
- Support
  - Process Guide for the Moderator
  - Group Support System (GSS) infrastructure

http://sunset.usc.edu/research/WINWIN/EasyWinWin/

Electronic Brainstorming, Glossary of Terms

Reveal conflicts and constraints
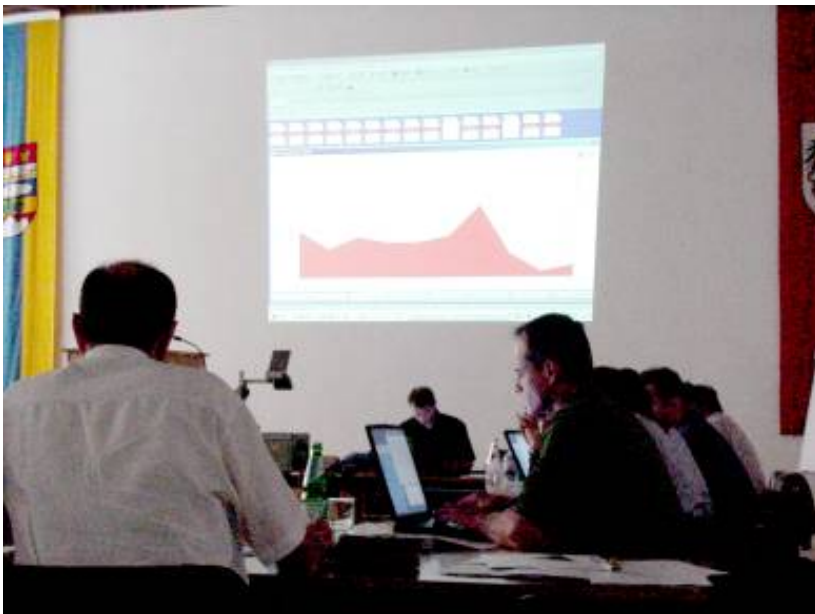
Improving Stakeholder Involvement

Benefits

Speed and efficiency for modest system

- Email and telephone: 1-3 months
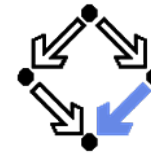
- EasyWinWin : 2-5 days

Low entry barrier for stakeholders

- Easy to learn and use

- Intuitive, time-efficient process

- Graphical Weather Forecast Editor
  - Complex graphical editor for the meteorologist

- Flood and Avalanche Control (Austria)
  - 25+ success-critical stakeholders
  - Shared vision and intitial requirements for mountain risk engineering system

From Iterative to Agile

Agile Alliance Manifesto

- Individuals and interactions over processes and tools

- Working software over comprehensive documentation

- Customer collaboration over contract negotiation

- Responding to change over following a plan

That is, while there is value in the items on

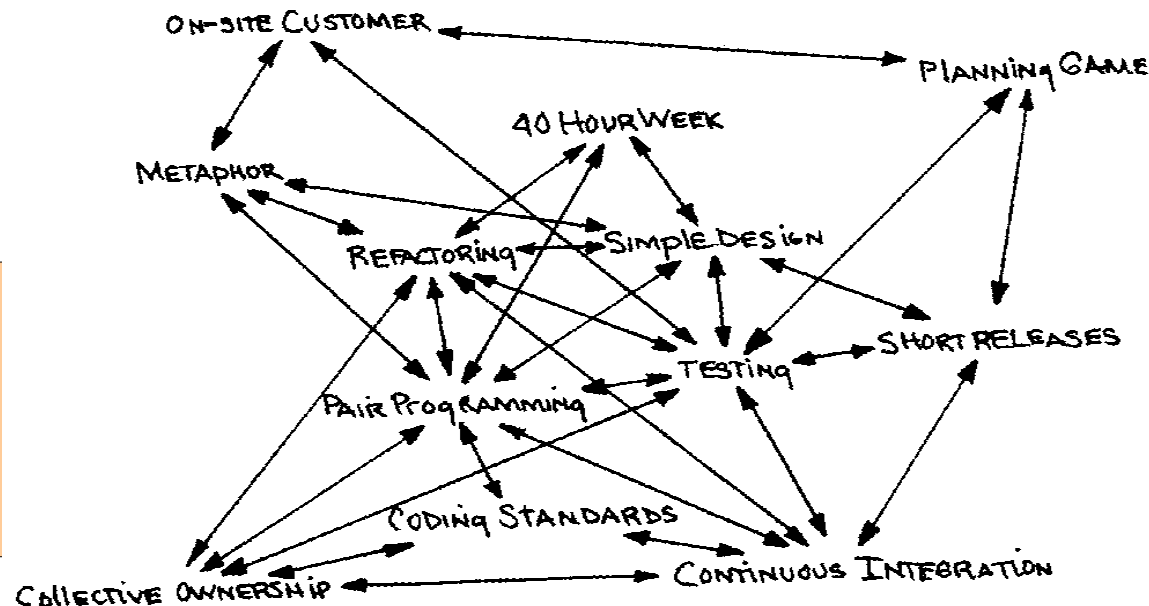the right, we value the items on the left more.

http://agilemanifesto.org/

http://www.computer.org/SEweb/Dynabook/WhatIs.htm
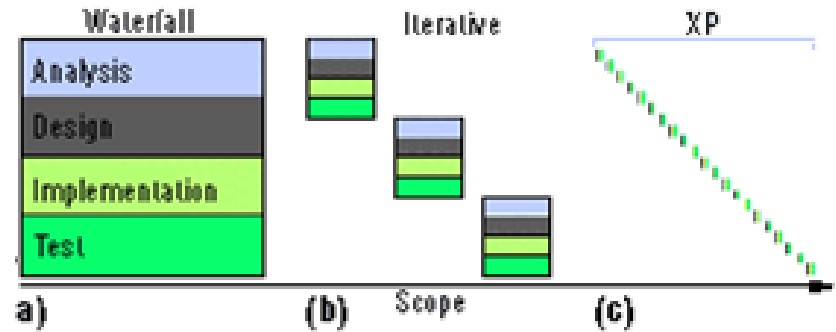
Caution: Practices that might non scale well

- Metaphor

- Small Releases

- Collective Ownership

- Refactoring

Summary

- Studies show critical areas in software development
- Win-Win Spiral model principles
  - Iterative
  - Risk-driven
  - Stakeholder involvement
  - Life-Cycle anchor points
  - Emphasis on system activities/artifacts
- Outlook
  - Value-based Software Engineering

    Biffl, S.; Aurum, A.; Boehm, B.; Erdogmus, H.; Grünbacher, P. (Eds.), Value-Based Software Engineering, 2006, 388 p., Hardcover, Springer-Verlag, ISBN: 3-540-25993-7

  - OCG Arbeitskreis "Software-Prozesse"

    http://www.ocg.at/

Stefan Biffl · Aybüke Aurum
Barry Boehm · Hakan Erdogmus
Paul Grünbacher (Eds.)

Value-Based
Software
Engineering

Springer