eXtreme Programming

Can Common Sense be so eXtreme ?

Lecture Project Management

presented by

Werner Wild, CEO Evolution
Innsbruck, San Francisco, Zurich

E-Mail: werner.wild@unibz.it
Slides at: Reserve Collection

Sad Facts – CHAOS Report

- Average Cost Overrun: 214%
- Average Time Overrun: 239%
- Average Specified Features implemented: 74%

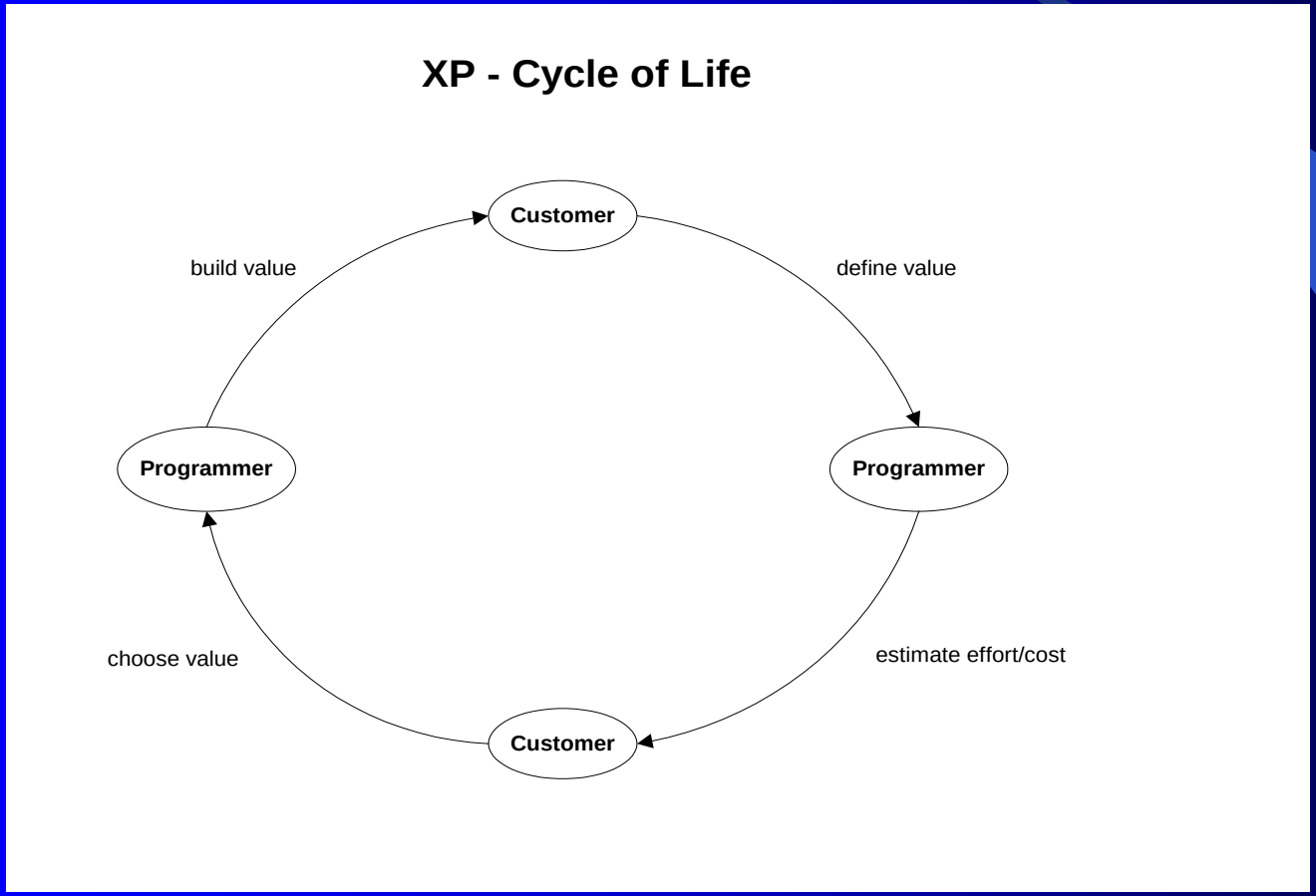
- Only 12% of all SW-Projects are on-time and on-budget !!!

Project Success Factors

Success Factor	% of Responses
1. User Involvement	15.9%
2. Executive Management Support	13.9%
3. Clear Statement of Requirements	13.0%
4. Proper Planning	9.6%
5. Realistic Expectations	8.2%
6. Smaller Project Milestones	7.7%
7. Competent Staff	7.2%
8. Ownership	5.2%
9. Clear Vision & Objectives	2.9%
10. Hard-Working, Focused Staff	2.4%
Other	13.9%

Sample size: 431 successful projects (out of 3682 !)

XP - Cycle of Life



Customer Bill of Rights

You have the right to:

- an Overall Plan
- get the most possible value per programmer's week
- see progress in a running system
- change your mind
- be informed of schedule changes

Programmer Bill of Rights

You have the right to:

- Know what's needed
- Produce quality at all times
- Ask for and receive help
- Make/update own estimates
- Accept responsibilities

Decisions – Business

Business decisions in planning are:

- Dates
- Scope
- Priority

Business decisions are made by customers only!

Customer' Mantra

Customer, ask yourself at ALL times:

"What is the most valuable functionality to have next?"

… and, by the way, trust the developer's estimates; after all, they are the technical experts ☺ … !

Decisions - Technical

Technical decisions in planning are:

- Estimates

Technical decisions are made by developers only!

Best Practices

- Manage Requirements
- Manage Change
- Use Component Architecture
- Develop Iteratively
- Model Visually
- Continuously Verify Quality

Agile Process' Risk Mitigation

Agile processes mitigate

- requirements risk
  by picking new requirements every few weeks
- implementation risk
  by breaking planning up into small enough pieces that when one piece blows up it will affect the overall plan as quickly and visibly as possible

Modeling – Why ?

- We are modeling for improved
  - Communication
  - Understanding

And not for code generation !

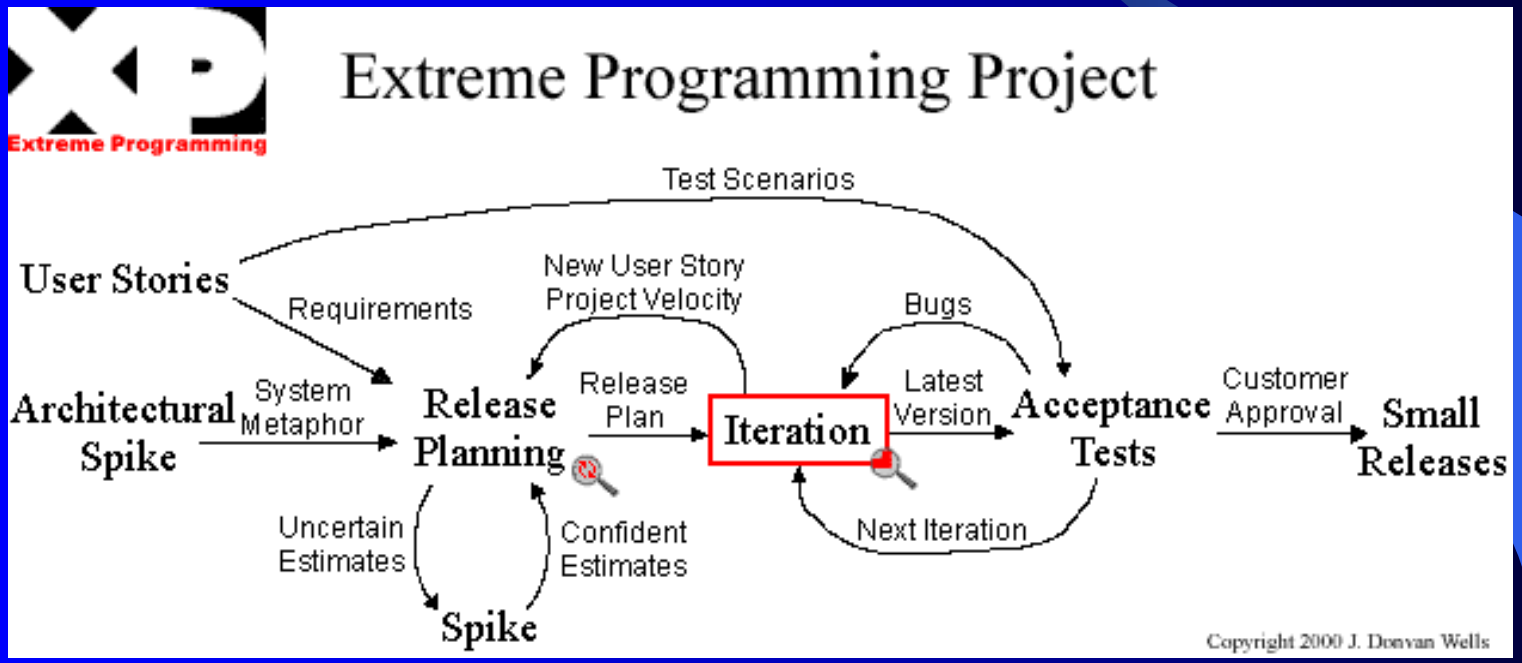
XP – 4 Values

- Communication
- Simplicity
- Feedback
- Courage

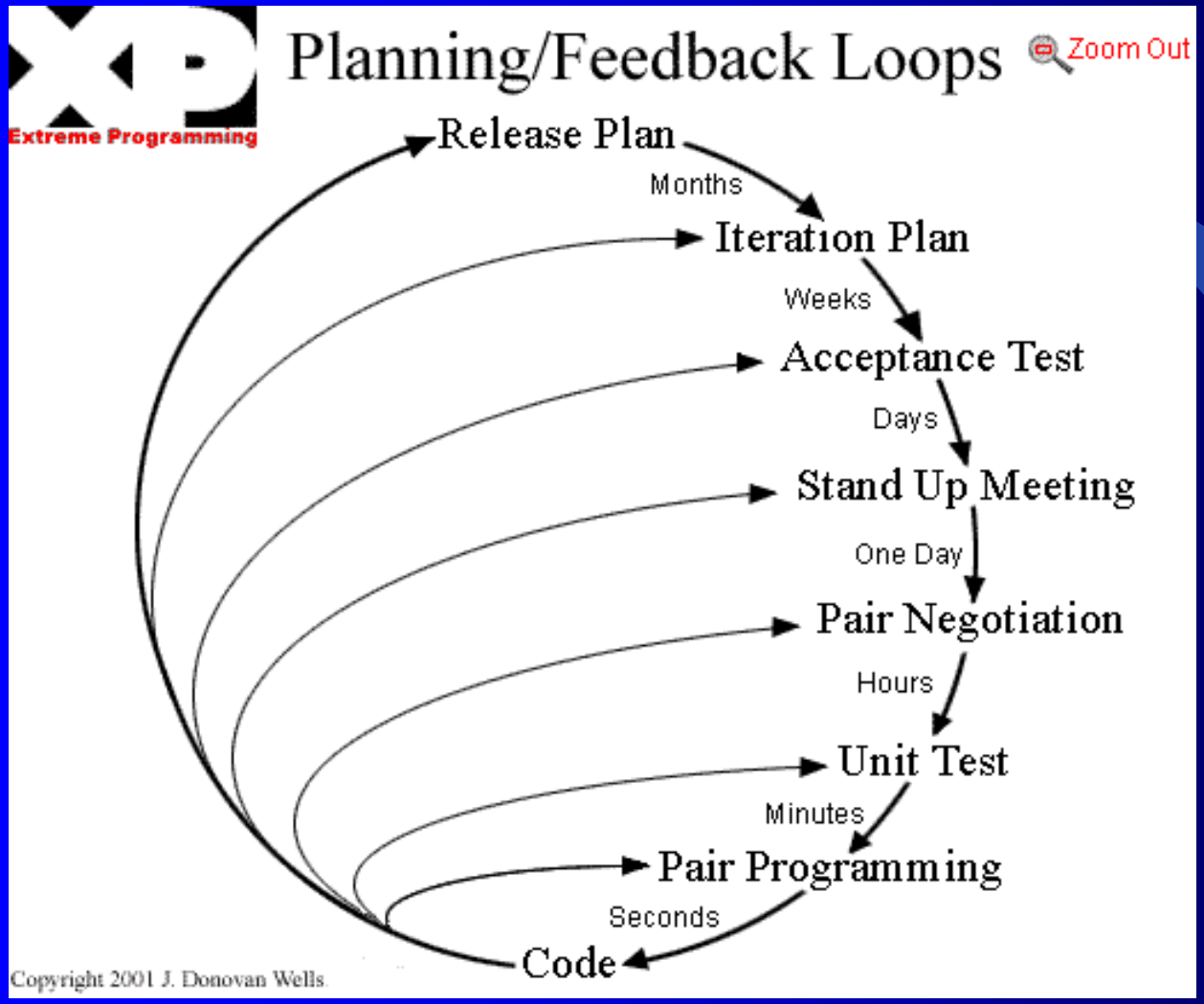
XP – 12 Practices

- The Planning Game
  - Quickly determine the scope of the next release. Priorities
- Small releases
  - Release new versions in very short cycles
- Metaphor
  - Find a simple metaphor describing how the system works
- Simple design
  - Make the design as simple as possible
- Testing
  - Test the code continuously. Write the tests before the production code.
- Refactoring
  - Mercilessly restructure the code to remove duplications, improve communication, simplify, or add flexibility
- Pair programming
  - Two programmers at one machine
- Collective ownership
  - Everyone owns and can change any code anywhere in the system
- Continuous integration
  - Integrate and build the system many times a day
- 40-hour week
  - Don't work more than 40 hours a week
- On-site customer
  - Include real users in the team – they must speak with one voice though
- Coding standards
  - Use a coding standard to improve communication

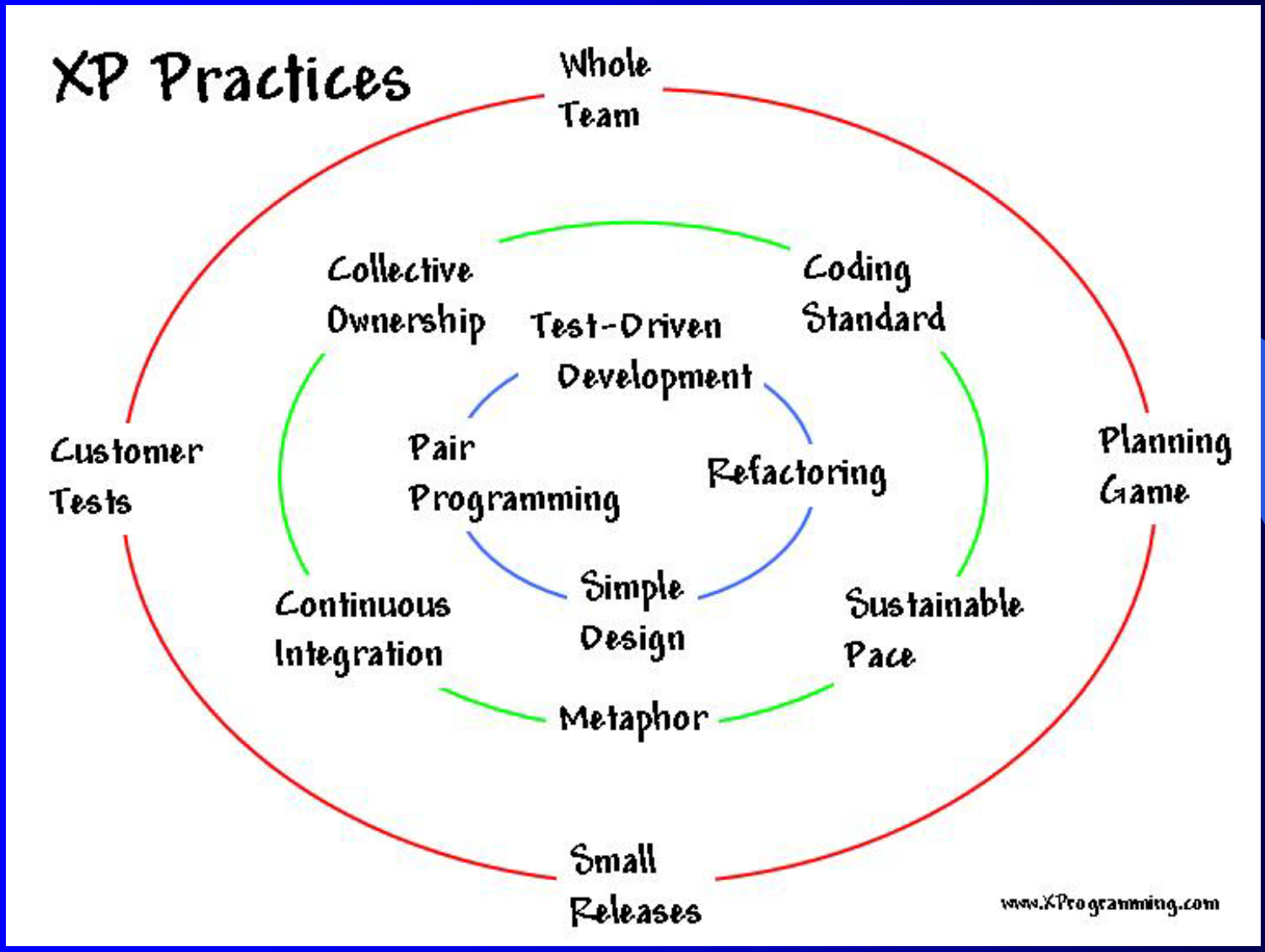
XP – A Simple Process



XP – Feedback Loops



Practices Working Together



XP – Misunderstood

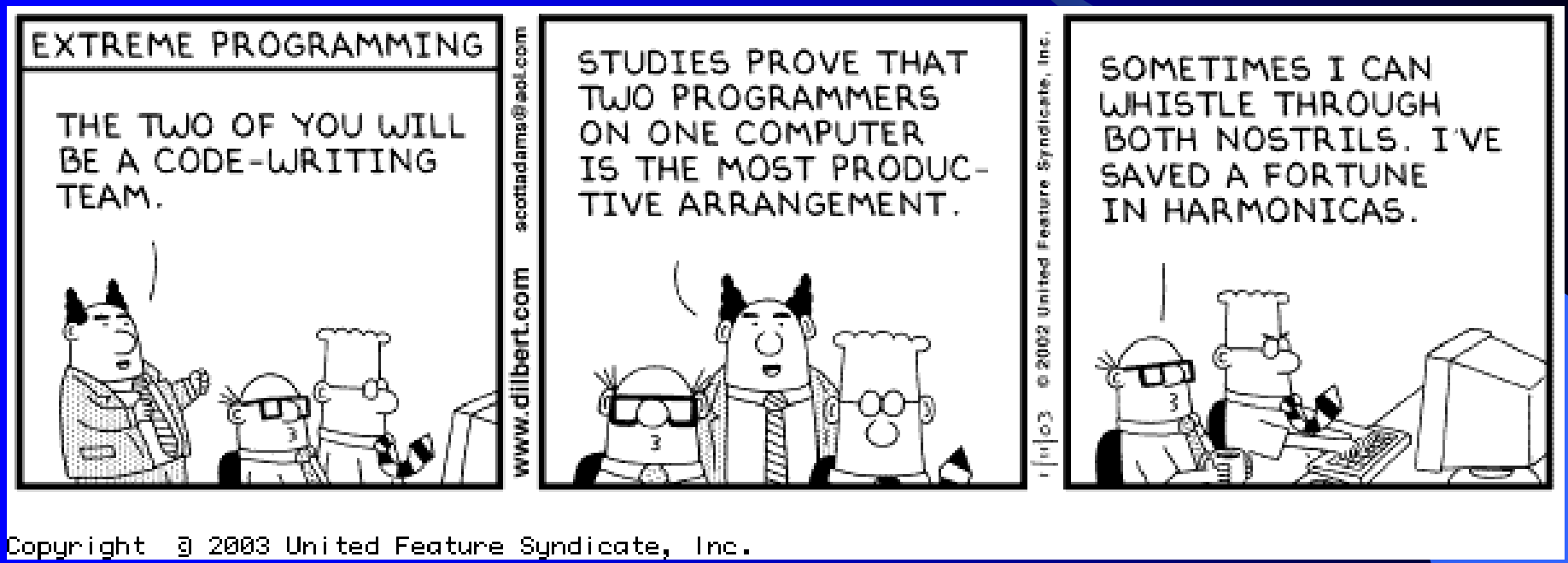
Four values revisited (1)

- Communication
  - Developers are encouraged to communicate with each other
  - A lot of the practices encourage communication
    - Pair programming, testing, planning game, continuous integration,…
- Simplicity
  - What is the simplest thing that could possibly work?

Four values revisited (2)

- Feedback
  - Tests, stories, the system is well structured, simple, and self-explanatory
- Courage
  - Correct errors or shortcomings with great courage, don't hesitate
    ("the rules" in concert give us this result!)

XP – Misunderstood



Experiences in Switzerland

- Highly positive feedback from real users ☺
  - After some initial hesitations to make real contributions to the planning game
- Same for user's managers ! ☺
  - Very excited: got features on time & when requested
- Not so from IT-Managers ! ☹
  - Feared loss of control & empowerment of users !!!
  - Politics: other development groups were not as successful (using non-agile approaches)

Experiences in Switzerland

- After leaving client things went down:
  - No more pair programming (after a few days)
  - No more planning games (after a few weeks)
  - Little testing
  - No more On-Premise customer contacts

  ⇒Back to chaotic software development (CMM level 0 ☺)

  ⇒People left the project !

Success Factors

- TRUST !!!
- Management Support (to set business priorities & customer availability)
- Eagerness to change culture
- Open minded, knowledgeable developers
- Developers not “socially challenged”
- Developers eager to learn and to experiment
- At least one highly experienced OO-developer in team !!!
- Motivated Customers
- Freedom to choose tools by the team, for the team
- …TRUST !!!

XP Introduction - The Wrong Way