# Perspectives on eXtreme Programming (XP)

Steven Fraser
*sdfraser@acm.org*
*Santa Clara, California*

## Outline

- Introduction to XP
- Perspectives on XP
- Discussion

## XP Beginnings?

- A set of 12 integrated software engineering and business best-practices
- The C3 project @ Chrysler (1996)
- *XP* Pioneer Kent Beck – with his first book: *"eXtreme Programming eXplained" (2000)*
  - In Kent's words a "physics experiment"

# Kent Beck's Simplification

*In my mind as we began XP as a physics experiment, where you remove all the variables possible so what you're left with is repeatable. Some of the usual variables we eliminated:*

- *Geographic separation*
- *Multiple customers*
- *Expensive deployment*
- *Stupid programmers*
- *Growth-averse database technology*
- *Computer-oriented programming language*
- *GUI-intense system*
- *Impersonal (>15 person) team*
- *Wildly changing requirements (replacing a legacy system)*
- *Disinterested business sponsors*

*Beck @ OOPSLA'02*

# XP Constraints

- Business/Requirements Engagement
  - Focused customers (on-site)
  - Engaged business sponsors
- Effective Teams
  - < 15
  - Standards
  - Pair programmers
  - Co-located teams
- Increasing Refactorability
  - Agile databases
  - Understandable code
- Constraining Complexity
  - Non-GUI-intense systems
  - Simplified deployment strategies

## XP “Values”

- Simplicity
- Communication
- Feedback
- Courage

*Beck, xTreme Programming Explained (00)*

## XP “Control Variables”

- Cost
- Time
- Quality
- Scope

*Beck, xTreme Programming Explained (00)*

# The 12 XP Practices

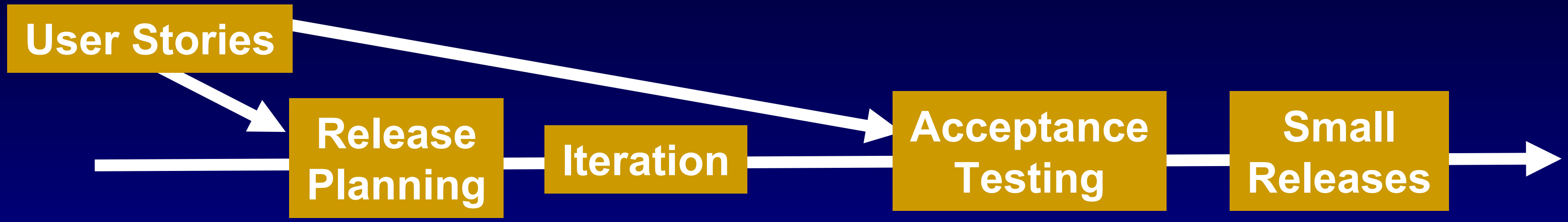
- Planning Game
- Small Releases
- Metaphor
- On-site Customer
- Simple Design
- Pair Programming
- Test-Driven Design
- Refactoring
- Continuous Integration
- Collective Ownership
- Coding Standards
- Sustainable Pace

*http://www.extremeprogramming.org*

## XP Practices Elaborated

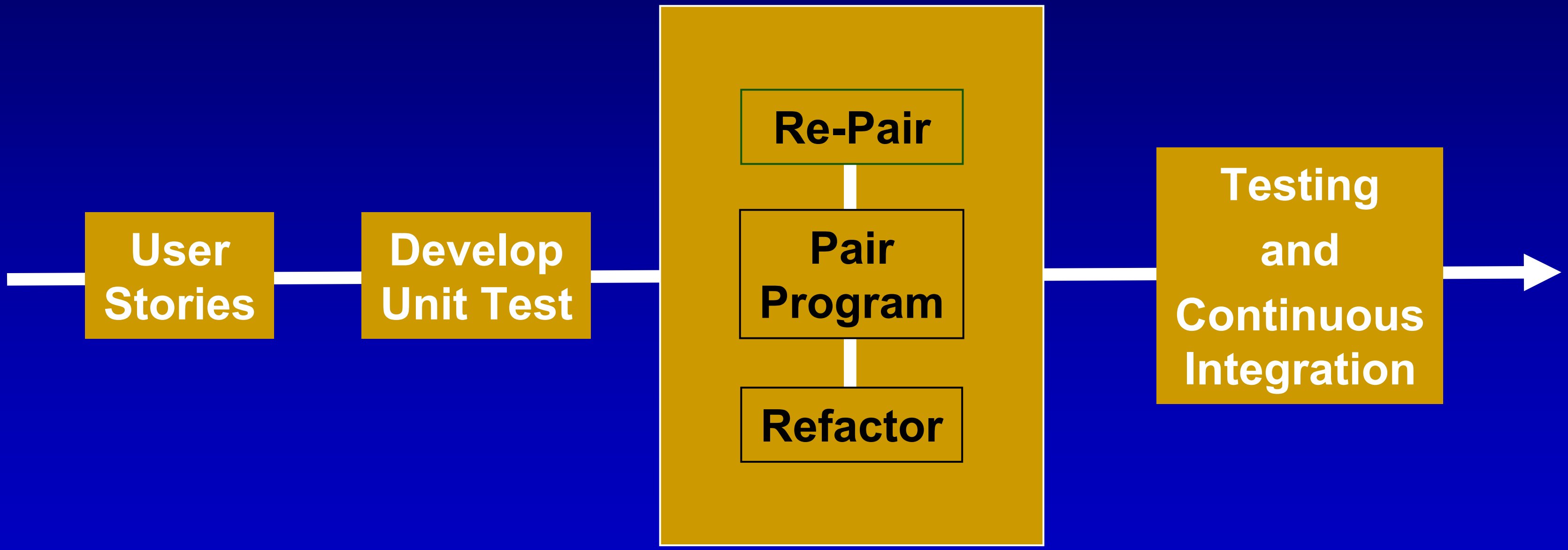
- Customers define application features with user "stories"
- A customer proxy is onsite throughout the project
- Automated unit tests developed and run continuously
- Teams put small code releases into production
- Teams use a common system of names and descriptions
- Simply written, understandable code meets requirements
- Code appears in a consistent style
- Teams frequently revise the overall code design
- Programmers work side-by-side in pairs
- Teams have collective ownership of the code
- Teams integrate code and release it to a repository every few hours
- Teams  work at a sustainable pace (no extended overtime)

# XP Process Increment



| Activity | Time-Scale |
|---|---|
| Increment | 2-3 Months |
| Iteration | 2-3 Weeks |
| Acceptance Testing | 1-2 Days |
| Stand-up Meeting | Once a Day |
| Pair Partnering | Hours - Days |
| Unit Testing | Minutes - Hours |
| Pair Programming | Minutes - Hours |

# XP Coding Iteration



# Xtreme Software Engineering?

| ***S/W Best Practice*** | ***Integrated into XP?*** |
|---|---|
| Planning | Iteration Planning/Incremental Release |
| Requirements Validation | On-site Customer/Stories |
| Design | Refactor |
| Iterations | Short-short Iterations |
| Regular Integration | Hourly/Daily Integration |
| Code Inspections | Pair Programming |
| Module Ownership & Standards | Collective Ownership & Standards |
| Testing | Automated Testing |
| Architecture | Metaphor (System of Names) |

# XP Perspectives

- Are all XP practices required?
- Will XP scale for systems/teams?
- How effective is pair programming?
- XP - “Hacker and Hero”?
- On-site customer engagement?
- XP project initiation?

## Are all XP practices required?

- Some would argue yes …
- Some would argue no …

…it depends!

# Perspectives on XP Practices

- It isn't XP if all 12 practices aren't used
- Additional practices are required, e.g. napping *[Beck @ OOPSLA'02]*
- Some practices are optional
- Do what is "right"

## Will XP scale for systems/teams?

- It doesn’t matter since if you need a big team - XP is not the discipline of choice
- Scale by introducing teams of teams

## How effective is pair programming?

- Evidence indicates – yes *[Williams & Kessler IEEE Software'00]*
- It depends…

## XP - “Hacker and Hero”?

- Skeptics argue that XP is an opportunity for a reversal of roles between zoo-keepers and their charges
- Converts suggest that XP success speaks for itself

## On-site customer engagement?

- Budget for the customer
- Motivate the customer

## How to initiate an XP project?

- Executive overview to engage sponsors
- Team XP overview tutorial
- Team XP practices/tools/standards tutorial
- Facilitate a “planning game”
- Coaching services (pair programming, test-first, …)
- Evaluate successes and challenges – iterate!

## XP Technology Transfer Questions

- Is the technology ready for “prime-time”?
- Is there a business case?
- Are the sponsors engaged?
- What is the critical path?

## Mechanisms for Tech Transfer

- Team tutorials, seminars, case studies
- Share demos and success stories
- Engage consultants, project managers, sponsors
- Establish pilot projects, news groups, forums…
- Manage logistics and schedule
- Assess impact and plan for follow-up

## “Tech Transfer” Success Measures

| | |
|---|---|
| **Revenue** | $, Customer Satisfaction, Success Stories |
| **Expense** | Time-to-Market, Tools, Consultants, Staff, Education, Infrastructure |
| **Quality** | Conformance Audits & Metrics |
| **Community** | Size, Stability & Growth |

*Success Momentum and Critical Mass*

# XP - Nirvana or Nemesis?

- Nirvana?
  - Cool Technology
  - Cost-Effective
  - Rapid Delivery
  - Customer Satisfaction
- Nemesis?
  - Real-time Systems
  - System/Team Size
  - Legacy Dependencies
  - Distributed Teams
  - Customer Interaction
  - Resistance to Change

## Summary

eXtreme Programming

- Bundles best-practices
- Is a team activity
- Requires discipline
- Continues to evolve

## For more Information:

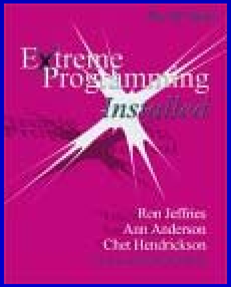
www.xprogramming.org

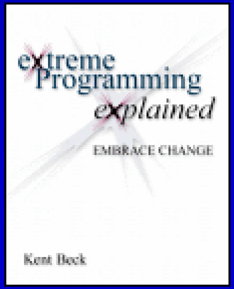
http://www.extremeprogramming.org/

http://www.xp2003.org

http://www.nalusda.gov/ttic/test1.htm

sdfraser@acm.org





Ken Auer and Roy Miller, Extreme Programming Applied: Playing to Win, Addison Wesley (2002).

Kent Beck, eXtreme Programming eXplained, Addison Wesley (2000).

Fred Brooks, The Mythical Man-Month, Addison Wesley (1995).

Luke Holmann, Journey of the Software Professional, Prentice Hall (1997).

Karl Wiegers, Creating a Software Engineering Culture, Dorset House (1996).

William C. Wake, Extreme Programming Explored, Addison Wesley (2001).

## About the Speaker

*Fraser is a California based consultant specializing in technology transfer, program management, and software process engineering. He spent more than 10 years with Nortel Networks in a variety of roles including: Process Architect, Senior Manager, and Disruptive Technologist.*

*From 1992 to 2001, Fraser was the Chair and Event Director of the Nortel Networks Design Forum, a proprietary global technology transfer event facilitated by interactive video conferencing, audio conferencing, web casting, and face-to-face interaction.*

*In 1994, he was a Visiting Scientist at the Software Engineering Institute (SEI) collaborating with the Application of Software Models project on the development of team-based domain analysis techniques. He completed his Doctoral studies in Electrical Engineering at McGill University in Montreal, Quebec, Canada.*