Collaborative Software Design & Development

Modifying Software Engineering Practices for Distributed Development (Collaboration 3)

James Hristu
Jeff Ruckel

Tuesday, March 25th 2008
Papers

“Tool Support for Geographically Dispersed Inspection Teams”
Filippo Lanubile, Teresa Mallardo, and Fabio Calefato

“Essential communication practices for Extreme Programming in a global software development team”
Lucas Layman, Laurie Williams, Daniela Damian, Hynek Bures
Introduction

- Distributed Software Development (DSD)
  - Becoming common practice in today's industry
- Global Software Development (GSD)
  - Special case of DSD where the team is dispersed across national boundaries
  - Why GSD?
    - Enable international corporate growth
    - Benefit from access to a larger resource pool
    - Reduce development cost
Challenges of GSD

→ Cultural and language differences
→ Trust and commitment
→ Extended feedback loops
→ Asynchronous communication
→ Knowledge management

→ These challenges seem to preclude software engineering processes that rely heavily on communication
Tailoring Processes for GSD

→ Communication-dependent processes must be tailored for GSD
  - Develop and adopt tools and technologies
  - Establish communication methodology
  - Modify the processes themselves

→ Affects traditional and modern processes

→ Our two papers are examples of how teams have modified processes for GSD
  - Inspections
  - Extreme Programming (XP)
“Tool Support for Geographically Dispersed Inspection Teams”

Filippo Lanubile, Teresa Mallardo, and Fabio Calefato

2004
IBIS Paper Overview

- Overview of Traditional Inspections
- Reengineering Inspections
- IBIS Overview
- Three Experiences with IBIS
Inspections Overview

- A specific form of peer review
- Primary goal:
  - To identify defects as early as possible
- Distinguishing characteristics:
  - A phased process to follow
  - Roles performed by peers during review
  - A reading toolset to guide individual defect detection activity
  - Forms and report templates to collect product and process data
Traditional Inspection Process

Phase 1 - Planning

- Choose team members
- Assign roles
  - Moderator
  - Author
  - Inspectors
- Make schedule
Traditional Inspection Process

Phase 2 - Overview

- Optional phase
- Present process and product info
- Familiarize newcomers

Diagram:
- Planning
- Preparation
- Inspection Meeting
- Rework
- Follow-up
Traditional Inspection Process

Phase 3 – Preparation

- Completed individually
- Analyze document
  - Code, requirements docs, etc.
- Find potential defects

Diagram:

1. Planning
2. Overview
3. Inspection Meeting
4. Rework
5. Follow-up
**Traditional Inspection Process**

**Phase 4 - Inspection Meeting**
- Team meets
- Collect and discuss defects
- Further review
Traditional Inspection Process

Phase 5 – Rework

- Author revises document
- Fix defects
Traditional Inspection Process

Phase 6 – Follow-up

Moderator verifies fixes
Gives final recommendation
Collects data
- Process
- Product
- For quality improvement
Evolution of Inspections

- Inspections have changed over time
- Impetus for change
  - Desire to reduce cost/interval
- Goals for Phases have changed
  - Preparation
    - Pure understanding → Defect detection
  - Inspection Meetings
    - Defect discovery → Collection and discrimination
- Need for meetings has been debated
Researching the Need for Meetings

- Parnas & Weiss (1987)
  - Dropped team meeting in Active Design Reviews

- Votta (1993)
  - Collection meetings increased inspection interval by 1/3
  - Meeting gains matched by meeting losses

- Several other studies
  - Net meetings improvement not positive
  - Nominal teams at least as effective as real teams
    - BUT reduced cost and interval

- However, there ARE benefits cited to meetings
  - Training newcomers, filtering false-positives, etc.
Reengineering the Inspection Process

→ Sauer et al. proposed reorganization of inspection process
→ Based on empirical studies
  ◦ Arguing against need for traditional meetings
  ◦ Behavioral theory of group performance
→ Attempt to reduce cost and interval
→ Replacement of phases
  ◦ Preparation and Inspection Meeting replaced
  ◦ New phases: Discovery, Collection and Discrimination
Traditional Vs. Reengineered Inspection Processes

Traditional

Planning

Overview

Rework

Follow-up

Reengineered

Planning

Overview

Rework

Follow-up
Reengineered Inspection Process

Phase 3 – Discovery

- Completed individually
- Analyze document
  - Code, requirements docs, etc.
- Find potential defects
- Reflects goal change
  - Pure understanding → Defect detection
Reengineered Inspection Process

Phase 4 – Collection

- Completed individually
  - By Author OR Moderator
- Removed goal of finding further defects
- Merge duplicate defects
Reengineered Inspection Process

Phase 5 – Discrimination

- Conducted optionally
- Only phase in reengineered process in which team members meet
- Reduced number of discussants
- Determine which defects are false positives
IBIS

- Internet-Based Inspection System (IBIS)
- Designed for geographically dispersed teams
- Implements Reengineered Inspection Process
- Uses dynamic web pages
- Event notification through email
Phase 1 - Planning

- **Initiated when:**
  - Author sends product for review

- **Moderator**
  - Determines that product meets entry criteria
  - Selects a template
  - Invites inspectors
  - Assigns reading techniques
Phase 2 - Overview

- Optional virtual meeting
- Uses 3rd party peer to peer conferencing tool
- Text-based speech
- Queues questions
Phase 3 - Discovery

- Conducted individually
- Supports
  - Checklist
  - Reading assignments
- Log defects
- When complete
  - Defects sent to moderator
Phase 4 - Collection

- Conducted individually
  - By Moderator
- Find and merge duplicates
- Each defect
  - Sent directly to rework OR
  - Plan a discrimination phase
Phase 5 - Discrimination

- Asynchronous discussion
- Selected participants
- Threaded discussion forum
  - Supports voting
  - Mark as false positive
  - Add comments
Phase 6 - Rework

- Author makes corrections
- Defect resolution entries
- Explanation if not fixed

![Defect Resolution Example](Image)
Phase 7 – Follow-up

- Moderator
  - Determines if defects have been corrected
  - May invite additional reviewers
- Notification email sent to inspection team
  - Detail report
  - Summary
Experiences with IBIS

- Three experiences
- Conducted with graduate students
  - Interacting from home or university labs
  - Simulating dispersed inspection teams
First Experience

- Conducted two inspections
  - Requirements document
  - Style violations in a web document
    - Style guide given to participants
- Unusually large inspection teams
  - 10 and 8 member teams, respectively
- Collected quantitative and qualitative data
## First Experience Results

<table>
<thead>
<tr>
<th></th>
<th>Inspection 1 (requirements document)</th>
<th>Inspection 2 (programming style)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collated defects</td>
<td>72</td>
<td>68</td>
</tr>
<tr>
<td>Defects selected for</td>
<td>34</td>
<td>9</td>
</tr>
<tr>
<td>discrimination</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Discussion filtering (selected</td>
<td>0.47</td>
<td>0.13</td>
</tr>
<tr>
<td>defs/collated)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Messages</td>
<td>78</td>
<td>16</td>
</tr>
<tr>
<td>Discussion intensity</td>
<td>2.3</td>
<td>1.8</td>
</tr>
<tr>
<td>(messages/selected defs)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Removed false positives</td>
<td>21</td>
<td>6</td>
</tr>
<tr>
<td>Slipped false positives</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>Discrimination efficacy</td>
<td>0.75</td>
<td>0.43</td>
</tr>
<tr>
<td>(Removed FP/all FP)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>True defects</td>
<td>44</td>
<td>54</td>
</tr>
</tbody>
</table>

- Discussants were more active when raising comments about requirements defects.
- Discriminating true defects from false positives was much more effective for requirements defects (discrimination efficacy = 0.75) than for style violations (discrimination efficacy = 0.43).
First Experience Results

- Participants stated
  - The tool was straightforward to use and convenient
  - Could perform the reviews at a time of their choosing

- Programming style defects were not worth discussing
  - Explicitly defined

- Divided opinions on asynchronous discussions
  - Some reviewers preferred a face-to-face discussion
    - They would be more drawn into the discussion
  - Other reviewers felt asynchronous discussion
    - Gave them more time to think about their comments
Second Experience

- IBIS used as an instrument for an empirical study on software inspections
- **Goal:**
  - To identify which defects were worth discussion and which discussants actively contribute to removing false positives
- All defects collated into a list (duplicates and unique)
  - Discussed during a Discrimination phase
  - Unique defects had a higher chance of being removed as false positives than duplicates
- Most discussion during the discrimination phase
  - Either by the author or the moderator
  - Other inspectors were less active
Third Experience

- Assess whether or not asynchronous discussions could replace face-to-face meetings
  - without losing effectiveness during the Discrimination phase
- Twelve teams formed for Discrimination phase
  - 6 teams for remote meetings
  - 6 teams for face-to-face meetings
- Each team to conduct Discrimination phase
  - 2 days notice for face-to-face
  - 2 days to conduct remote discussion
- Finding:
  - Asynchronous discussions as effective as face-to-face meetings
Conclusions

→ Asynchronous communications
  ➖ Allow for a larger team to be used
    ➖ Without “incurring an unbearable overhead”

→ The inspection process can be optimized by
  ➖ Focusing discussion on defects that have been reported by only one reviewer
  ➖ Restricting discussion to a subset of the inspection team

→ Asynchronous discussion
  ➖ Can be as effective as face-to-face meetings
    ➖ For discriminating between true defects and false positives.
Assessment of IBIS Paper

⇒ Pros
  ⇨ Well written
  ⇨ Good summary of inspections and their history
  ⇨ Excellent example of tailoring a traditional software engineering practice for GSD
  ⇨ Includes a broad list of alternate tools to IBIS

⇒ Cons
  ⇨ Not very technical
  ⇨ Not especially innovative
  ⇨ Three experiences are not very deep
  ⇨ Validity concerns, including:
    ➢ Grad students simulating GSD
    ➢ Not working on industry products
    ➢ Etc.
“Essential communication practices for Extreme Programming in a global software development team”

Lucas Layman, Laurie Williams, Daniela Damian, Hynek Bures

2006
XP Paper Overview

→ Extreme Programming (XP) Background
→ Case Study Introduction
→ Research Method
→ Team and Project Factors
→ Conjectures and Recommendations
→ Conclusion
Extreme Programming (XP) Background

- Agile software development practice that is highly responsive to customers, and builds quality software by respecting key values

- Key values
  - Communication
    - User/Programmer collaboration, verbal feedback
  - Simplicity
    - Simple code, simple design, build for today
  - Feedback
    - Unit testing, acceptance tests, team feedback
  - Courage
    - Persistence, removing code
  - Respect
    - Never breaking unit-tests, high quality designs
**XP-EF (Evaluation Framework)**

- **XP Context Factors (XP-cf)**
  - Record context factors such as team size, project size, and staff experience

- **XP Adherence Metrics (XP-am)**
  - Use objective and subjective metrics to express concretely and comparatively the extend to which a team utilizes XP practices

- **XP Outcome Measures (XP-om)**
  - Assess the business related results (productivity, quality, etc.) of using a full or partial set of XP practices
Case Study Introduction

- GSD team that used Extreme Programming (XP)
- Industrial case study from initial requirements definition through the end of product development
- Developed a simulator for a telecommunication signal transfer point system
- US telecommunication provider contracted a Czech software development corporation
  - First time working together
Research Method

→ Goals
  - Understanding everyday mechanics of team processes
  - Evaluation of the final product in terms of quality, productivity, and customer satisfaction

→ Data Collection
  - Qualitative
    - Informal email prompt sent to project management team in US and lead developer in the Czech Republic
    - Face-to-face interviews conducted with the customer and proxy customer after 3 months of production use
  - Quantitative
    - Thousands of lines of executable code (KLOEC)
    - Measurement of effort through XPlanner project tracking tool
    - XP-EF for team data, process factors, project outcomes
Team Factors

➔ Development team
  ➥ Czech Republic
  ➥ Development Lead and Software Developers
  ➥ Team size varied from 4 (beginning) to 7 (deadline) to 2 (maintenance).
  ➥ Experienced in telecommunication development
  ➥ Required to learn new interfaces for the project

➔ Project management team
  ➥ USA
  ➥ Project Manager, Customer, Proxy Customers (2), Project Tracker, Development Manager
Process Factors

- Team implemented many tenets of XP
  - Over 75% of the work was spent in pairs
  - Practiced test-driven development and unit tested all code
  - Collective code ownership
  - Integrated their code into a build machine once per day

- Sometimes they departed from standard XP practices
  - Planning meetings were only between project management in the USA and the development lead in the Czech Republic
    - Absence of the development team
    - Time-zone issues
  - Project management personnel explained system components on a whiteboard, which the development lead could not see
    - Diminished the usefulness of XP style informal meetings
Project Factors

➔ User stories
   ➲ Over 65 user stories in the final product
   ➲ Amount of work involved in each user story varied greatly
      ➲ Result of the team’s unfamiliarity with XP

➔ Iterations were 10 days in length
   ➲ XPlanner used for project velocity

➔ Collaboration tools
   ➲ Free VOIP technologies
   ➲ Mailing list
   ➲ XPlanner
   ➲ Instant Messaging (IM)
Project Outcome

➔ Prior to release, the system was tested by the customer, and bugs entered as user stories
  ➔ Defect numbers are lower than published industry averages
  ➔ Only one fatal crash after the product was shipped

➔ Simulator did not cover all the desired aspects of a real system
  ➔ Given time restraints, the developers and customer considered the project a success

➔ Productivity of the project was lower
  ➔ 4th generation language
  ➔ Extensive unit testing
Conjecture 1

→ Conjecture 1

“...In a globally-distributed XP team a well defined customer authority is essential for effective decision making and a clear requirements statement”

→ Customer role was ill-defined for the first month
→ After troublesome month, a customer was defined
Conjecture 1

Recommendation:

“Define a person to play the role of the customer up front. This individual must be able to make conclusive decisions on project functionality and scope, must be readily accessible, and must have a vested interest in the project.”

A definite customer:

- Important in XP for requirements management
- Answer development questions and make planning decisions
- Should be continuously available
- Provide customer acceptance test cases (CATs)
Conjecture 2

In a globally-distributed XP team, having a key member of one team physically located with the other team can provide an essential two-way communication conduit.

Development manager served this role

- Technical lead on the project who worked closely with both developers (Czech) and project management personnel (American)
- Served as technical and cultural liaison
  - For developers, he could answer technical questions and provide feature details quickly
  - For project management, he understood the status and progress of the project
- Only Czech-speaking project member in the US
Conjecture 2

Recommendation:

“When the project management and development teams are separated, create a role within the XP team whose purpose is to work closely with both development and project management teams on a daily basis, preferably someone who speaks all languages involved”

However, there is risk associated with a project that becomes overly dependent upon any small number of individuals
Conjecture 3

In a globally-distributed XP team, prompt responses to asynchronous queries positively impact development commitment and confidence and create a focused development environment.

Free VOIP technologies

Time-zone issues only allowed 2 hours of overlap.

Instant Messaging

Not used when discussing user requirements, as team desired all stakeholders to be aware of such conversations.

Informal mailing list

In place of the informal face-to-face communication in XP.
Conjecture 3

→ Recommendation

“When face-to-face, synchronous communication is infeasible, use an email listserv to increase the chance of a response and encourage prompt, useful, and conclusive responses to email”
Conjecture 4

- **Conjecture 4**
  - “In a globally-distributed XP team, providing the team with continuous access to process and product information can help us improve process control and plan effectiveness”

- **XPlanner tool**
  - Used to store electronic versions of user stories
  - Allowed the customer to have high visibility in the process and gauge the project’s progress
Recommendation

"Use globally-available project management tools to record and monitor the project status on a daily basis"

The recommended use of such a tool is not novel, but it's a particularly important communication tool in XP GSD
Conclusion

→ Face-to-face communication approximated by:
  - An email listserv
  - Globally-available project management tool
  - Intermediary development manager

→ Four success factors for XP GSD:
  - Communication bridgehead
  - Asynchronous communication loops
  - Customer authority
  - High process visibility
Assessment of XP Paper

➔ Pros
   ➔ Detailed case study, not just experiences
   ➔ Conjectures similar to IBIS paper conclusions
   ➔ Well written

➔ Cons
   ➔ Single case study
   ➔ Hypothesis driven, not data driven as claimed?
   ➔ Conjectures are derived from a successful XP GSD project
      ➔ What about a failed project?
Summary

► Communication-dependent processes must be tailored for GSD

► Examples of how processes were tailored for GSD
  ➤ Develop and adopt tools and technologies
    ➤ IBIS itself
    ➤ XPlanner, IM, VoIP, etc.
  ➤ Establish communication methodology
    ➤ IBIS enforces communication channels
    ➤ Evening conference call between Czech Rep. and US
  ➤ Modify the processes themselves
    ➤ IBIS use of reengineered inspection process
    ➤ Absence of development team during planning meetings
Summary (cont'd)

→ Asynchronous Communications

 hann Advantages
 hann Generally more suited for globally distributed teams
 ✓ Time-zone disparity
 hann Preferred by nonnative English speakers
 ✓ Allows more time to choose words

 hann Disadvantage
 hann Blocking effects increase interval
Questions?