Managing Socio-Technical Dependencies in Distributed Software Development

Anita Sarma
University of Nebraska, Lincoln
July 2015
SATToSE 2015

General Chair: Tom Mens
Program Chair: Anya Helene Bagge

Current Research Group:
Coordination

RAF Leuchars Airshow 2013, [Martyn Edward](https://www.flickr.com/photos/martynedward/), Flickr
Dependencies


Parallel Work

Cleidson R. B. de Souza, David F. Redmiles, Li-Te Cheng, David R. Millen, John F. Patterson: How a good software practice thwarts collaboration: the multiple roles of APIs in software development. SIGSOFT FSE 2004: 221-230


Yuriy Brun, Reid Holmes, Michael D. Ernst, David Notkin: Crystal: precise and unobtrusive conflict warnings. SIGSOFT FSE 2011: 444-447
Conflicts in Distributed Software Development

- **Direct Conflicts**: Two developers edit the same file concurrently (Merge conflicts)

- **Indirect Conflicts**: Conflicts arising because of changes in one file affecting changes in another (Build & Test conflicts, and other breaking changes)
## Conflicts in Distributed Software Development

<table>
<thead>
<tr>
<th>Project</th>
<th>#Merges</th>
<th>#conflicts</th>
<th># conflicts</th>
<th># Res. Days Avg (Med)</th>
<th># conflicts</th>
<th># Res. Days Avg (Med)</th>
<th># conflicts</th>
<th># Res. Days Avg (Med)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perl</td>
<td>185</td>
<td>74 (40%)</td>
<td>14 (8%)</td>
<td>23 (10)</td>
<td>4 (2%)</td>
<td>0.7 (1)</td>
<td>56 (30%)</td>
<td>31 (14)</td>
</tr>
<tr>
<td>Storm</td>
<td>88</td>
<td>39 (44%)</td>
<td>17 (19%)</td>
<td>6 (2)</td>
<td>9 (10%)</td>
<td>5 (8)</td>
<td>13 (15%)</td>
<td>8 (3)</td>
</tr>
<tr>
<td>Jenkins</td>
<td>505</td>
<td>204 (54%)</td>
<td>68 (14%)</td>
<td>23 (4)</td>
<td>74 (15%)</td>
<td>5 (2)</td>
<td>28 (6%)</td>
<td>7 (2)</td>
</tr>
<tr>
<td>Voldemort</td>
<td>380</td>
<td>170 (34%)</td>
<td>55 (15%)</td>
<td>20 (4)</td>
<td>16 (4%)</td>
<td>2 (0.75)</td>
<td>133 (35%)</td>
<td>6 (4)</td>
</tr>
</tbody>
</table>

- Merge conflicts: 8% to 19%
- Build conflicts: 2% to 15%
- Test conflicts: 6% to 35%
Communication


Sabrina Marczak, Ban Al-Ani, David F. Redmiles, Rafael Prikladnicki: The Interplay among Trust, Risk, and Reliance in Global Systems Engineering Teams. ICGSE 2014: 46-55

Patrick Wagstrom, Subhajit Datta: Does latitude hurt while longitude kills? geographical and temporal separation in a large scale software development project. ICSE 2014: 199-210
Interdependency

Anita Sarma, Ban Al-Ani, Erik Trainer, Roberto Silveira Silva Filho, Isabella A. da Silva, David F. Redmiles, André van der Hoek:

Adrian Bachmann, Christian Bird, Foyzur Rahman, Premkumar T. Devanbu, Abraham Bernstein:
The missing links: bugs and bug-fix commits. SIGSOFT FSE 2010: 97-106
Coordination Tools Can Help

Photo by www.gotcredit.com on Flickr
Coordination Paradigms

Coordination Paradigms

- Basic Functionality
  - Asynchronous communication
  - Instant Messaging, monitoring changes to artifacts
  - Communication archival along with artifacts
  - Passive awareness of development activities and developers, manage information overload

- Structured Process
  - Parallel development, roles and access rights
  - Fine grained versioning, conflict resolution
  - Prescribed and defined coordination support

- Information Discovery
  - Parallel development, roles and access rights
  - Fine grained versioning, conflict resolution
  - Organizational memory, knowledge acquisition and dissemination, relations navigation

- Information Provision
  - Continuous coordination, collaborative architecture, seamless development environments,
  - Advanced conflict detection
  - Collocation benefits to distributed development

- Integrated Operations
  - Task allocation and assignment
  - Task management
  - Artifact management
  - Communication management

- Artifact Management
  - Access to common set of artifacts, isolated workspaces and version control

- Communication
  - Asynchronous communication
  - Instant Messaging, monitoring changes to artifacts
  - Communication archival along with artifacts

- Task Management
  - Task allocation and assignment
  - Prescribed and defined coordination support
  - Organizational memory, knowledge acquisition and dissemination, relations navigation

- Coordination Paradigms
  - Continuous coordination, collaborative architecture, seamless development environments,
  - Advanced conflict detection
  - Collocation benefits to distributed development
  - Passive awareness of development activities and developers, manage information overload
  - Parallel development, roles and access rights
  - Fine grained versioning, conflict resolution
  - Organizational memory, knowledge acquisition and dissemination, relations navigation
  - Communication archival along with artifacts
  - Asynchronous communication
  - Prescribed and defined coordination support
  - Task allocation and assignment
  - Artifact management
  - Communication management
  - Task management
Coordination Paradigms

- Continuous coordination, collaborative architecture, seamless development environments,
- Passive awareness of development activities and developers, manage information overload
- Advanced conflict detection
- Collocation benefits to distributed development
- Instant Messaging, monitoring changes to artifacts
- Fine grained versioning, conflict resolution
- Organizational memory knowledge acquisition and dissemination, relations navigation
- Communication archival along with artifacts
- Parallel development, roles and access rights
- Prescribed and defined coordination support
- Asynchronous communication
- Access to common set of artifacts, isolated workspaces and version control
- Task allocation and assignment

Communication  Artifact Management  Task Management
Examples of Coordination Tools

Integrated Operations

Continuous coordination, collaborative architecture, seamless development environments,

Passive awareness of development activities and developers, manage information overload

Advanced conflict detection

Collocation benefits to distributed development

Instant Messaging, monitoring changes to artifacts

Fine grained versioning, conflict resolution

Organizational memory, knowledge acquisition and dissemination, relations navigation

Communication archival along with artifacts

Parallel development, roles and access rights

Prescribed and defined coordination support

Asynchronous communication

Access to common set of artifacts, isolated workspaces and version control

Task allocation and assignment

Communication
Artifact Management
Task Management
Interactive and explorative environment to correlate and understand the complex relationships among:

- code (artifacts)
- communication records
- tasks (bug reports)
- time

Tesseract Screenshot
Filtered Artifact Network

Project

File name

Developers

Issues

Severity:  Enhancements  Trivial  Minor  Normal  Major  Critical  Blocker

Empirically-based Software Quality Research

Computer Science and Engineering, UNL
Cross-Linked Displays

- Project
  - gnomes/rhythmbox
  - DoubleClick node to drill down upon all selected.

- Files
  - File name
  - rhythmdb/rhythmdb-query-mid
  - rhythmdb/rhythmdb-query-mid
  - rhythmdb/rhythmdb-bridge
  - rhythmdb/rhythmdb-bridge
  - rhythmdb/rhythmdb-high
  - library/crosstools
  - shell/Makefile.am
  - shell/main.c
  - shell/rb-shells.c
  - shell/rb-remote.c
  - iradio/Makefile.am
  - iradio/rb-station-properties.d
  - shell/rb-playlist-manager.c
  - shell/rb-shell-player.c
  - shell/rb-shell-player.h

- Developers
  - Name
  - Jill Berner
  - Ruth Schmitz
  - Lorraine Henkel
  - Tine Reab
  - Alicia Dimaggio
  - Stephen Walther
  - Charles Kautz
  - Margie Caruth
  - Norman Buffington
  - Thelma Gillman
  - Jane Pagan
  - Sonya Pinckney
  - Douglas Parry
  - Joseph Truesdale
  - Rodney Harwood
  - Chris Dickinson

- Issues
  - Severity: ✓ Enhancement ✓ Trivial ✓ Minor ✓ Normal ✓ Major ✓ Critical ✓ Blocker
  - Graph showing changes over time.
Results

- Seasoned developers create a model of ST dependencies
- Useful for onboarding and expertise finding
  - externalizing the network
- Useful in finding gaps in communication (managers)

Open questions
- what other types of relationships might be useful?
- should users be allowed to drive these investigations?
- how will such investigations scale – computation and cognitive?
DOMINOES – Exploratory Data Analysis

- Explore relationships in software repositories
- Use matrices to represent relationships
- Combine matrices through linear operations

Concept

- Developer modified files
- Files changed together
- Transpose of developer modified files
- Who needs to coordinate with whom

\[
\begin{align*}
(A) &= \begin{bmatrix}
a_{11} & \ldots & a_{1k} \\
a_{n1} & \ldots & a_{nk}
\end{bmatrix} \\
(D) &= \begin{bmatrix}
d_{11} & \ldots & d_{1k} \\
d_{k1} & \ldots & d_{kk}
\end{bmatrix} \\
(A^T) &= \begin{bmatrix}
a_{11} & \ldots & a_{1n} \\
a_{k1} & \ldots & a_{kn}
\end{bmatrix} \\
(C_R) &= \begin{bmatrix}
cr_{11} & \ldots & cr_{1n} \\
cr_{n1} & \ldots & cr_{nn}
\end{bmatrix}
\end{align*}
\]

\[
X = (A) \times (D) \times (A^T) = (C_R)
\]

[Cataldo et al., 2006]
DOMINOES – Matrices

- Basic tiles
  - [class|method], [commit|method], [developer|commit], [bug|commit]

- Derived tiles
  - [developer|method], [bug|method]
  - [method|method], [class|class], [developer|developer]
  - ...

- Allows investigations of:
  - expertise breadth of a developer
  - latent team structure
  - predicting conflicts
  - predicting bugginess of a project
  - ...

DOMINOES UI

Empirically-based Software Quality Research

Computer Science and Engineering, UNL

e2.unl.edu 22
Examples of Coordination Tools

- **Integrated Operations**
  - Continuous coordination, collaborative architecture, seamless development environments,
  - Collocation benefits to distributed development

- **Information Provision**
  - Passive awareness of development activities and developers, manage information overload
  - Advanced conflict detection

- **Information Discovery**
  - Instant Messaging, monitoring changes to artifacts
  - Fine grained versioning, conflict resolution
  - Organizational memory, knowledge acquisition and dissemination, relations navigation

- **Structured Process**
  - Communication archival along with artifacts
  - Parallel development, roles and access rights
  - Prescribed and defined coordination support

- **Basic Functionality**
  - Asynchronous communication
  - Access to common set of artifacts, isolated workspaces and version control
  - Task allocation and assignment

**Communication**

**Artifact Management**

**Task Management**
Monitor ongoing changes in remote workspace
- Identify potential conflicts
  - merge conflicts (direct conflicts)
  - conflicts arising from dependency violation (indirect conflicts)
- Notify developers of emerging conflicts

Other Workspace Awareness Tools

- **Current tools (Conflict mitigation):**
  - CollabVS [Dewan et al., ECSCW’07]
  - FastDash [Biehl et al., CHI’07]
  - Crystal [Brun et al. FSE’11]
  - WeCode [Guimarães & Rito Silva., ICSE’12]
Results

- Conflicts are detected as they emerge
- Developers undertake action upon noticing a potential conflict
- Communication time increases during the task

- Open questions
  - how to promote awareness at higher levels of abstraction?
  - how can impact analyses scale to identify behavioral changes?
  - how to balance information overload and Interruption?
  - how to track and notify about changes across branches?
  - ...
Examples of Coordination Tools

- **Communication**
  - Asynchronous communication

- **Artifact Management**
  - Access to common set of artifacts, isolated workspaces and version control

- **Task Management**
  - Task allocation and assignment

**Integrated Operations**
- Continuous coordination, collaborative architecture, seamless development environments,
  - Collocation benefits to distributed development

**Information Provision**
- Passive awareness of development activities and developers, manage information overload
  - Organizational memory, knowledge acquisition and dissemination, relations navigation

**Information Discovery**
- Instant Messaging, monitoring changes to artifacts
  - Prescribed and defined coordination support

**Structured Process**
- Communication archival along with artifacts
  - Parallel development, roles and access rights

**Basic Functionality**
- Fine grained versioning, conflict resolution
  - Task allocation and assignment

- Advanced conflict detection

- Asynchronous communication
  - Access to common set of artifacts, isolated workspaces and version control

- Parallel development, roles and access rights
  - Prescribed and defined coordination support

- Communication archival along with artifacts
  - Organizational memory, knowledge acquisition and dissemination, relations navigation

- Instant Messaging, monitoring changes to artifacts
  - Parallel development, roles and access rights
Cassandra Approach

- Obtain task context (task – files)
- Order of tasks (Developer preferences)

- Identify edited files ($F_e$)
- Identify dependent files ($F_d$)

- Analyze tasks for conflicts

- Formalize constraints
  - hard constraints (>)
  - soft constraints ($\neq$)

Evaluate Constraints
Constraint Evaluation

Evaluate Constraints (Z3)

- Optimize Solution
- Match developer preferences
- Display conflict information
- Display recommended task order

Re-evaluate constraints

SAT

[ 4 2 3 1 ]

[ 1 2 3 4 ]

Evaluate Constraints (Z3)

Re-evaluate constraints

SAT

[ 4 2 3 1 ]

[ 1 2 3 4 ]

Evaluate Constraints (Z3)

Re-evaluate constraints

SAT

[ 4 2 3 1 ]

[ 1 2 3 4 ]
Interface – Mylyn

Interface – Mylyn Plugin

![Task List](image1)

![Task List](image2)

![Problems](image3)
Constraint Evaluation

Evaluate Constraints (Z3)

- SAT
  - Optimize Solution
  - Match developer preferences

- UnSAT
  - Relax constraints

- Display conflict information
- Display recommended task order

Re-evaluate constraints

[ 4 2 3 1 ]
[ 1 2 3 4 ]

Empirically-based Software Quality Research

Computer Science and Engineering, UNL
Results

- Cassandra successful in
  - scheduling conflict minimal tasks
  - 50%-97% conflicts avoided, less than 3 minutes
  - optimizing based on developer preferences
  - sensitivity of task context ($F_e$) precision

- Open Questions
  - what factors can be used to relax constraints?
  - how can we automate effort estimation?
  - can we schedule tasks across development stages?
  - ...

Bakhtiar Khan Kasi, Anita Sarma: Cassandra: proactive conflict minimization through optimized task scheduling. ICSE 2013: 732-741
Road Ahead
Future Directions for Coordination Tools

- Integrated Operations
  - Continuous coordination, collaborative architecture, seamless development environments,
  - Collocation benefits to distributed development

- Information Provision
  - Passive awareness of development activities and developers, manage information overload
  - Advanced conflict detection

- Information Discovery
  - Instant Messaging, monitoring changes to artifacts
  - Fine grained versioning, conflict resolution

- Structured Process
  - Communication archival along with artifacts
  - Parallel development, roles and access rights
  - Prescribed and defined coordination support

- Basic Functionality
  - Asynchronous communication
  - Access to common set of artifacts, isolated workspaces and version control
  - Task allocation and assignment

- Artifacts Management
- Task Management
Future Directions for Coordination Tools

With use of social media on the rise, how does it affect the communication strand?

We also need to coordinate across different development stages!

What about End Users? They coordinate asynchronously

Move from individual tools and (small) empirical studies to create theories of coordination

Communication
Artifact Management
Task Management

Continuous coordination, collaborative architecture, seamless development environments
Development activities and developers, manage information overload
Advanced conflict detection
Collocation benefits to distributed development
Monitoring changes to artifacts
Fine grained versioning, conflict resolution
Knowledge acquisition and dissemination, relations navigation
Communication archival along with artifacts
Parallel development, roles and access rights
Prescribed and defined coordination support

Asynchronous communication
Access to common set of artifacts
Isolated workspaces and version control
Parallel development, roles and access rights
Passive awareness of development activities and developers, manage information overload
Advanced conflict detection
Collocation benefits to distributed development

Empirically-based Software Quality Research
Computer Science and Engineering, UNL
Thank you!

- This work is supported by:
  - NSF CCF -1016134, IIS-1110916, IIS-1314365, CCF-CAREER
  - AFOSR - 9550-10-1-0406

- Interaction Design and Coordination Lab & Collaborators