Software Practitioner Perspectives on Merge Conflicts and Resolutions

Shane McKee, Nicholas Nelson, Anita Sarma, and Danny Dig
Oregon State University, Corvallis, Oregon
{mckee, nelson, anita, dig}@oregonstate.edu

@nnelson8675  nicozone
Why practitioner perspectives matter

“You cannot combine tens of conflicting commits... it’s not sane.”

“I have to jump around between tools and copy and paste version numbers... this is why integration matters.”

“I’m often dealing with code other people wrote. Nobody can review every pull request... Code is much easier to write than read.”
Merge Conflicts and Resolutions

- Collaborative development requires periodic synchronization of divergent changes.

- 19% of all merges result in merge conflicts. [Kasi & Sarma 2013, Brun et al. 2011]

- Resolutions can cause delays, integration errors, workflow disruptions. [Bird et al. 2012, Estler et al. 2014]

- Resolving merge conflicts is non-trivial.
- We have focused on techniques, predictions, and automated resolutions.
- However, practitioner perspectives have largely been ignored.
Research Goal & Questions

To empirically understand the perspective of practitioners when they approach and perform merge conflict resolutions.

RQ1 How do software practitioners approach merge conflicts?

RQ2 What unmet needs impact the difficulty of a merge conflict resolution?

RQ3 How well do tools meet practitioner needs for merge conflicts?
**Study Design - Interviews and Surveys**

**Interviews**

- Provided insight into how practitioners approach merge conflicts, and their unmet needs.
- 10 software practitioners from 7 organizations.
- Median of 5 years of software development experience.

<table>
<thead>
<tr>
<th>Part.</th>
<th>Exp.</th>
<th>Role</th>
<th>Project Source</th>
<th>Project Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>18 yrs.</td>
<td>Sr. Software Developer</td>
<td>Open</td>
<td>1700</td>
</tr>
<tr>
<td>P2</td>
<td>6 yrs.</td>
<td>Software Engineer</td>
<td>Open</td>
<td>1700</td>
</tr>
<tr>
<td>P3</td>
<td>3 yrs.</td>
<td>Software Engineer</td>
<td>Open</td>
<td>1700</td>
</tr>
<tr>
<td>P4</td>
<td>10 yrs.</td>
<td>Software Developer</td>
<td>Open</td>
<td>&lt;10</td>
</tr>
<tr>
<td>P5</td>
<td>3 yrs.</td>
<td>Infrastructure Engineer</td>
<td>Closed</td>
<td>&lt;10</td>
</tr>
<tr>
<td>P6</td>
<td>5 yrs.</td>
<td>Software Developer</td>
<td>Closed</td>
<td>&lt;10</td>
</tr>
<tr>
<td>P7</td>
<td>5 yrs.</td>
<td>Software Engineer</td>
<td>Open</td>
<td>200</td>
</tr>
<tr>
<td>P8</td>
<td>25 yrs.</td>
<td>Director</td>
<td>Open</td>
<td>600</td>
</tr>
<tr>
<td>P9</td>
<td>8 yrs.</td>
<td>Software Developer</td>
<td>Open</td>
<td>600</td>
</tr>
<tr>
<td>P10</td>
<td>2 yrs.</td>
<td>Software Developer</td>
<td>Open</td>
<td>&lt;5</td>
</tr>
</tbody>
</table>
Study Design - Interviews and Surveys

Survey

- Validate our results and provided a broader perspective.
- 162 participants from the software development industry.
- 74.2% had 6 or more years of software development experience.
Results
RQ1: How do practitioners approach merge conflicts?

F1  Complexity of conflicting lines of code
F2  Your knowledge/expertise in area of conflicting code
F3  Complexity of the files with conflicts
F4  Number of conflicting lines of code
F5  Time to resolve a conflict
F6  Atomicity of changesets in the conflict
F7  Dependencies of conflicting code on other components
F8  Number of files in the conflict
F9  Non-functional changes (whitespace, renaming, etc.)
RQ1: How do practitioners approach merge conflicts?

F1: Complexity of conflicting lines of code
F2: Your knowledge/expertise in area of conflicting code
F3: Complexity of the files with conflicts
F4: Number of conflicting lines of code
F5: Time to resolve a conflict
F6: Atomicity of changesets in the conflict
F7: Dependencies of conflicting code on other components
F8: Number of files in the conflict
F9: Non-functional changes (whitespace, renaming, etc.)
RQ1: How do practitioners approach merge conflicts?

Top-4 Factors for Difficulty

● Technical Aspects
  ○ Complexity of the code (F1, F3)
  ○ Size of the conflicting changes (F4)

● Social Aspects
  ○ Expertise in area of conflicting code (F2)

“Small is always easy. A 1-line merge conflict is always easier than a 400-line merge conflict, and can be done now.”
RQ1: How do practitioners approach merge conflicts?

Top-4 Factors for Difficulty

- **Technical Aspects**
  - Complexity of the code (F1, F3)
  - Size of the conflicting changes (F4)

- **Social Aspects**
  - Expertise in area of conflicting code (F2)

“A lot of what I work on is in my own little area... I know what to do... But in unfamiliar parts of code, then I’ll get someone else to resolve the merge conflict for me.”
RQ2: What unmet needs impact the difficulty of a merge conflict resolution?

N1  How easy is it to understand the code involved in the merge conflict

N2  Your expertise in the area of code with the merge conflict

N3  The amount of information you have about the conflicting code

N4  How well tools present information in an understandable way

N5  Changing assumptions within the code

N6  Complexity of the project structure

N7  Trustworthiness of tools

N8  Informativeness of commit messages

N9  Project culture

N10 Tool support for examining development history
RQ2: What unmet needs impact the difficulty of a merge conflict resolution?

N1: How easy is it to understand the code involved in the merge conflict

N2: Your expertise in the area of code with the merge conflict

N3: The amount of information you have about the conflicting code

N4: How well tools present information in an understandable way

N5: Changing assumptions within the code

N6: Complexity of the project structure

N7: Trustworthiness of tools

N8: Informativeness of commit messages

N9: Project culture

N10: Tool support for examining development history

Mean score (0 indicates no effect, 5 indicates great effect)
RQ2: What unmet needs impact the difficulty of a merge conflict resolution?

Top-4 Unmet Needs

- Social Needs
  - Expertise in area of conflicting code (N2)

- Technical Needs
  - Understandability of code (N1)
  - Contextual information about the conflict (N3)
  - Tool presentation of relevant info (N4)

“I’m often dealing with code other people wrote... So now I have to go back and do some archaeology to find out what’s going on. Code is much easier to write than read.”
RQ2: What unmet needs impact the difficulty of a merge conflict resolution?

Top-4 Unmet Needs

- **Social Needs**
  - Expertise in area of conflicting code (N2)

- **Technical Needs**
  - Understandability of code (N1)
  - Contextual information about the conflict (N3)
  - Tool presentation of relevant info (N4)

“You focus on understanding the small change, not the big one. It’s easier to understand... get the small change to go with the flow of the bigger change.”
RQ2: What unmet needs impact the difficulty of a merge conflict resolution?

Closed-Source Practitioners

N1 How easy is it to understand the code involved in the merge conflict
N2 Your expertise in the area of code with the merge conflict
N3 The amount of information you have about the conflicting code
N4 How well tools present information in an understandable way
N5 Changing assumptions within the code
N6 Complexity of the project structure
N7 Trustworthiness of tools
N8 Informativeness of commit messages
N9 Project culture
N10 Tool support for examining development history

Open-Source Practitioners

N1 How easy is it to understand the code involved in the merge conflict
N2 Your expertise in the area of code with the merge conflict
N3 Tool support for examining development history
N4 The amount of information you have about the conflicting code
N5 How well tools present information in an understandable way
N6 Changing assumptions within the code
N7 Complexity of the project structure
N8 Trustworthiness of tools
N9 Informativeness of commit messages
N10 Project culture
RQ2: What unmet needs impact the difficulty of a merge conflict resolution?

<table>
<thead>
<tr>
<th>Closed-Source Practitioners</th>
<th>Open-Source Practitioners</th>
</tr>
</thead>
<tbody>
<tr>
<td>N1 How easy is it to understand the code involved in the merge conflict</td>
<td>N1 How easy is it to understand the code involved in the merge conflict</td>
</tr>
<tr>
<td>N2 Your expertise in the area of code with the merge conflict</td>
<td>N2 Your expertise in the area of code with the merge conflict</td>
</tr>
<tr>
<td>N3 The amount of information you have about the conflicting code</td>
<td>N3 <strong>Tool support for examining development history</strong></td>
</tr>
<tr>
<td>N4 How well tools present information in an understandable way</td>
<td>N4 The amount of information you have about the conflicting code</td>
</tr>
<tr>
<td>N5 Changing assumptions within the code</td>
<td>N5 How well tools present information in an understandable way</td>
</tr>
<tr>
<td>N6 Complexity of the project structure</td>
<td>N6 Changing assumptions within the code</td>
</tr>
<tr>
<td>N7 Trustworthiness of tools</td>
<td>N7 Complexity of the project structure</td>
</tr>
<tr>
<td>N8 Informativeness of commit messages</td>
<td>N8 Trustworthiness of tools</td>
</tr>
<tr>
<td>N9 Project culture</td>
<td>N9 Informativeness of commit messages</td>
</tr>
<tr>
<td><strong>N10 Tool support for examining development history</strong></td>
<td>N10 Project culture</td>
</tr>
</tbody>
</table>
RQ2: What unmet needs impact the difficulty of a merge conflict resolution?

**Closed-Source Practitioners**

- N1 How easy is it to understand the code involved in the merge conflict
- N2 Your expertise in the area of code with the merge conflict
- N3 The amount of information you have about the conflicting code
- N4 How well tools present information in an understandable way
- N5 Changing assumptions within the code
- N6 Complexity of the project structure
- N7 Trustworthiness of tools
- N8 Informativeness of commit messages
- N9 Project culture
- N10 Tool support for examining development history

**Open-Source Practitioners**

- N1 How easy is it to understand the code involved in the merge conflict
- N2 Your expertise in the area of code with the merge conflict
- N3 **Tool support for examining development history**
- N4 The amount of information you have about the conflicting code
- N5 How well tools present information in an understandable way
- N6 Changing assumptions within the code
- N7 Complexity of the project structure
- N8 Trustworthiness of tools
- N9 Informativeness of commit messages
- N10 Project culture

OSS projects have frequent changes, in goals and personnel, which requires additional support for history exploration. This “pain point” has not been addressed by current merge toolsets.
RQ3: How well do tools meet practitioner needs for merge conflicts?

11 Better usability
12 Better ways of filtering out less relevant information
13 Better ways of exploring project history
14 Better graphical presentation of information
15 Better transparency in tool functionality/operations
16 Better terminology that is more consistent with my other tools
RQ3: How well do tools meet practitioner needs for merge conflicts?

1. Better usability
2. Better ways of filtering out less relevant information
3. Better ways of exploring project history
4. Better graphical presentation of information
5. Better transparency in tool functionality/operations
6. Better terminology that is more consistent with my other tools

Mean score (0 indicates no effect, 5 indicates great effect)
RQ3: How well do tools meet practitioner needs for merge conflicts?

Top-3 Tool Improvements

- Better Usability (I1)
  ○ Average of 2.5 tools for merge conflicts

- Better Filtering of Less-Relevant Information (I2)
  ○ Larger projects, larger scalability concerns

- Better Project History Exploration (I3)
  ○ Practitioners use workaround, but seamless support is needed

“I have to jump around between tools and copy and paste version numbers... this is why integration matters.”
RQ3: How well do tools meet practitioner needs for merge conflicts?

Top-3 Tool Improvements

- Better Usability (I1)
  - Average of 2.5 tools for merge conflicts

- Better Filtering of Less-Relevant Information (I2)
  - Larger projects, larger scalability concerns

- Better Project History Exploration (I3)
  - Practitioners use workaround, but seamless support is needed

“You want to extract the relevant commits. The ones that actually clash... you want to zoom in on them and understand just enough and don’t waste time.”
RQ3: How well do tools meet practitioner needs for merge conflicts?

Top-3 Tool Improvements

- Better Usability (I1)
  - Average of 2.5 tools for merge conflicts

- Better Filtering of Less-Relevant Information (I2)
  - Larger projects, larger scalability concerns

- Better Project History Exploration (I3)
  - Practitioners use workaround, but seamless support is needed

“Give me a way to explore the history. To drill down, to go back up, you know? To resurface and understand what happened.”
Effectiveness of practitioners’ toolsets in supporting perceived size and complexity of merge conflicts, split along development experience.
Effectiveness of practitioners’ toolsets in supporting perceived size and complexity of merge conflicts, split along development experience.
Effectiveness of practitioners’ toolsets in supporting perceived size and complexity of merge conflicts, split along development experience.
Effectiveness of practitioners’ toolsets in supporting perceived size and complexity of merge conflicts, split along development experience.

Mean: 4.278
\(\Delta(A1-A2): 0.496\)

Mean: 3.782
\(\Delta(A1-A3): 0.930\)

Mean: 3.347
Mean: 2.783
Conclusions

RQ1 How do software practitioners approach merge conflicts?

Practitioners rely on their expertise in the conflicting code to understand and assess the merge conflict.

Practitioners rely on simple estimations, rather than precise metrics calculated by tools.
Conclusions

RQ2 What unmet needs impact the difficulty of a merge conflict resolution?

Practitioners have unmet needs along dimensions of:

1. comprehending code snippets in isolation,
2. understanding the code context underlying multiple code snippets that are split across multiple files, and commits,
3. The ability to quickly comprehend the complexity of these code snippets.
Conclusions

RQ3  How well do tools meet practitioner needs for merge conflicts?

● Tools are lacking in *usability, information filtering*, and *history exploration* support.
● Practitioners are doing *workarounds* and using *multiple tools* to resolve merge conflicts.
● Tools do not scale to large, complex merge conflicts (especially along the complexity dimension).
Research supported by NSF grants CCF-1439957, CCF-1553741, CCF-1560526, and IIS-1559657.

@nnelson8675  nicozone
RQ1: Difficulties in Assessing Merge Conflicts

F1 Complexity of conflicting lines of code
F2 Your knowledge/expertise in area of conflicting code
F3 Complexity of the files with conflicts
F4 Number of conflicting lines of code

RQ2: Unmet Needs in Resolving Merge Conflicts

N1 How easy is it to understand the code involved in the merge conflict
N2 Your expertise in the area of code with the merge conflict
N3 The amount of information you have about the conflicting code
N4 How well tools present information in an understandable way

RQ3: Merge Conflict Tool Improvements

I1 Better usability
I2 Better ways of filtering out less relevant information
I3 Better ways of exploring project history
## Survey Participants Merge Toolsets (Top 10)

<table>
<thead>
<tr>
<th>Tool</th>
<th># of Participants</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Git</td>
<td>37</td>
<td>Version Control System</td>
</tr>
<tr>
<td>Vim/vi</td>
<td>17</td>
<td>Text Editor</td>
</tr>
<tr>
<td>Text Editor (unspecified)</td>
<td>14</td>
<td>Text Editor</td>
</tr>
<tr>
<td>Git Diff</td>
<td>11</td>
<td>Diffing Tool</td>
</tr>
<tr>
<td>GitHub</td>
<td>11</td>
<td>Website</td>
</tr>
<tr>
<td>Eclipse</td>
<td>10</td>
<td>IDE</td>
</tr>
<tr>
<td>KDiff3</td>
<td>9</td>
<td>Diff &amp; Merge</td>
</tr>
<tr>
<td>Meld</td>
<td>8</td>
<td>Diff &amp; Merge</td>
</tr>
<tr>
<td>SourceTree</td>
<td>8</td>
<td>Git/Hg Desktop Client</td>
</tr>
<tr>
<td>Sublime Text</td>
<td>7</td>
<td>Text Editor</td>
</tr>
</tbody>
</table>
Survey Participants

Professional Development Experience

- 1-5 years: 16
- 6-10 years: 5
- 11-15 years: 24
- 16-20 years: 59
- 21-25 years: 16
- 26+ years: 42

Collaborative Software Project Roles

- Software Engineer/Developer
- Systems Engineer
- Project Manager
- Systems Administrator
- Project Maintainer
- Other(s)
Merge Conflict

- Merge conflicts are a scenario in which two copies of the same codebase diverge and cannot be automatically merged, thus requiring human intervention to resolve.

- This definition excludes other types of conflicts that exist within software projects (i.e. social conflicts or semantic conflicts, such as build or test failures).