Proactive Detection of Higher-Order Software Design Conflicts

Jae young Bang and Nenad Medvidovic

Computer Science Department, Viterbi School of Engineering, University of Southern California

April 15th, 2015
Collaborative Software Design

- Complex modern software systems are designed by multiple software architects
- Architects simultaneously make design decisions
- Those decisions are documented into software models
- Collaborative evolution of software models
- Tracking and organizing modeling changes is not trivial
Version Control Systems

• “Systems that track incremental versions of files and directories over time”*

• Widely adopted in
  • General software development: SVN, CVS, Git, etc.
  • Software design: IBM DM, EMFStore, etc.

• Implement loose, on-demand synchronization
  • Individual workspaces
  • Changes are synchronized only when requested

• Pros
  • Work in parallel, higher productivity

• Cons
  • Not fully aware of each others’ work
  • Conflicts may occur

A Conflict Scenario

- Collaborative design, hard requirement on memory usage
  - Finds they have made changes that:
    1. Were made to the same object and were incompatible
    2. Can be merged but together violate system’s intended rules

- Makes a change
- Finds no issue in memory usage

- Makes a change
- Finds no issue in memory usage

Letty

Dom
A Conflict Scenario

• Letty and Dom make conflicting changes
  • They found they have made changes that:
    Synchronization: 1. Are not compatible and cannot be merged together or
    Higher-order: 2. Can be merged but together violate system’s intended rules

• Other names of synchronization conflicts
  • Context-free, Textual, or Direct conflicts

• Other names of higher-order conflicts
  • Context-sensitive or Indirect conflicts

• This research focuses on the higher-order conflicts
Why The Conflicts Are Bad

- Unrealistic not to cause any conflict, using a VCS
- Having an *unknown* conflict is a big risk
- Work done after a conflict was introduced may need to be reversed in the process of resolving the conflict
- Wasted time and effort – increased cost

“We often face inconsistencies between components developed by different engineers at later stages. *Half of the cases lead to full-scale reverting to earlier stages, and local patches are made for the other half*”

– A quote from an interview with a practicing professional architect

Revisiting Version Control

- What is the risk of not knowing an outstanding conflict?

What if Dom was aware of the conflict earlier?

Conflict found!
Dealing w/ Unknown Conflicts

• A solution to the risk of having unknown conflicts
  • Detect conflicts highly frequently, e.g., for each change
  • The cost may overwhelm its benefit – a trade-off

• Expensive detection techniques render the frequent detection even more unaffordable
  • Well-known, computation-intensive techniques
    • Discrete-event simulation
    • Markov-chain-based reliability analysis
    • Queueing-network-based performance analysis
Proactive Conflict Detection (PCD)

- Tools for collaborative *implementation* arose that detect conflicts proactively, i.e., before a developer syncs
- In the background, trial merging and conflict detection
- Hard to directly use them for collaborative design
- No empirical data yet reported on whether or to what extent PCD would impact collaborative *design*
FLAME

- **Framework for Logging and Analyzing Modeling Events**
  - Provides facilities for proactive detection of higher-order software design conflicts on top of model version control
  - Logs modeling changes and conflict detection results

- **Extensibility**
  - Extension points for plugging-in off-the-shelf tools

- **Operation-based version control**
  - Tracks modeling operations rather than “difs” between stored states of model (e.g., files)
  - Conflict detection can become more fine-grained
High-Level Architecture

![Diagram of High-Level Architecture]

- Client Manager
- Detector Manager
- FLAME Client
- Detection Engine
- FLAME Adapter
- Architect
- Modeling tool
- Detection tool
Off-the-Shelf Tools Integrated

Prism-MW the middleware from USC

GME* the modeling tool from Vanderbilt Univ.

XTEAM** the model analysis tool-chain from USC
1. Message latency
2. Memory usage
3. Energy consumption

*: Generic Modeling Environment (GME)
**: eXtensible Tool-chain for Evaluating Architectural Models (XTEAM)
Detection Engine

- Similar to FLAME Client

- What it does
  - Applies operation to local model (merging)
  - Invokes its conflict detection tool
  - Reports conflict information

- There can be multiple Detection Engines integrating different conflict detection tools
Performing Proactive Conflict Detection

- What conflicts may arise if everyone performs a commit now?

Detection Engine

Server-side

Letty

Dom

Black: committed op
White: uncommitted op
Empirical Evaluation

- User study with 42 participants
- **USC** grad students in the Software Architecture class
- Participants divided into two groups
  - w/ PCD: FLAME *does* present conflict information
  - w/o PCD: FLAME *does NOT* present conflict information
- Conflicts were tracked from their creation to resolution
- Participants given *same length of design time*
- **Resulting model quality** was measured
User Study Details

<table>
<thead>
<tr>
<th>Target system</th>
<th>Next-Generation Climate Architecture (NGCA) *</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of participants</td>
<td>42</td>
</tr>
<tr>
<td>Number of teams</td>
<td>21 teams of 2</td>
</tr>
<tr>
<td>User study period</td>
<td>Span of 18 days</td>
</tr>
<tr>
<td>Modeling tool</td>
<td>GME</td>
</tr>
<tr>
<td>Conflict detection tool</td>
<td>XTEAM</td>
</tr>
</tbody>
</table>

- **Open-source system** chosen (NGCA)
- Collaborative design scenario based on design documents
- Scenario with 3 system properties
  - Memory usage, energy consumption, message latency
- Participants went through **4-week-long pre-training**
- 1-hour tutorial, 30-min main design, 30-min alt. design

* : Created based on NASA Computational Modeling Algorithms and Cyberinfrastructure (CMAC)
Conflict Notification

- Delivers conflict information with minimal interruption
- Color flags that present conflict information

All system requirements are met

Memory requirement not met
The NGCA Model
Empirical Eval. Takeaways

- With PCD
  - **More communication** between architects
  - **Earlier resolution** of higher-order conflicts
  - **Higher quality** resulting models

<table>
<thead>
<tr>
<th>FLAME mode</th>
<th>w/o PCD</th>
<th>w/ PCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session duration (minutes)</td>
<td>30.00</td>
<td>30.00</td>
</tr>
<tr>
<td>Number of operations performed</td>
<td>48.18</td>
<td>60.80</td>
</tr>
<tr>
<td>Number of communication activities</td>
<td>11.00</td>
<td>19.50</td>
</tr>
<tr>
<td>Higher-order conflict lifetime (seconds)</td>
<td>671.00</td>
<td>343.40</td>
</tr>
<tr>
<td>Proportion of unresolved conflicts at session end</td>
<td>33%</td>
<td>0%</td>
</tr>
</tbody>
</table>

The higher is better.

<table>
<thead>
<tr>
<th>Throughput: energy consumption (J)</th>
<th>8.18 M</th>
<th>8.55 M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput: memory usage (MB)</td>
<td>729.09</td>
<td>747.60</td>
</tr>
</tbody>
</table>
Hard Conflict Lifetimes

- Some conflicts are easier to resolve, possibly because they are already known to the architects
- Some conflicts were resolved early within 100 secs (27%)
- Participants resolved hard conflicts earlier with PCD

![Box plot showing lifetime (seconds) comparison between w/o PCD and w/ PCD]
Summary

• **Problem**: The inevitable higher-order design conflicts

• **Solution**: FLAME
  - Proactive conflict detection for collaborative design
  - An extensible, operation-based collaborative design framework that proactively detects higher-order conflicts

• With proactive conflict detection
  - More frequent communication between architects
  - Earlier resolution of higher-order conflicts
  - Higher quality resulting models in same amount of time
Contributions

• The first application of PCD for collaborative design
• The extensible collaborative design environment
• The first reported empirical evidence that PCD positively impacts collaborative design
Future Work

• Variations of FLAME
  • *Detection Engines* that derive other versions of model
  • *Detection Engines* that detect synchronization conflicts
  • FLAME GUIs presenting conflict information differently

• FLAME as a foundation
  • Conflict *cause assessment* – which modeling operations have caused the conflict?
  • Conflict *resolution support* by speculating possible resolution activities and proactively assessing the impacts of those
Thank you

Contact: Jae young Bang (jaeyounb@usc.edu)