Simulation-based Architectural Design and Implementation of a Real-time Collaborative Editing System

2007 DEVS Integrative Modeling & Simulation Symposium (DEVS'07)

Jon A Preston
Xiaolin Hu
Sushil Prasad
Agenda

- Goals
- Background
- Simulation Design Process
- Models
- Simulation Configurations
- Results
- Conclusions
Goals

- Design and develop a Real Time Collaborative Editing System (RTCES) using simulation design
  - Multiple simultaneous users
  - Shared document
  - Distributed collaboration
- Reduce communication costs
  - Existing solutions are costly
Background

- RTCES are a subset of Computer Supported Collaborative Work (CSCW)
- Musts provide responsiveness in the editor
  - Thus replication of document is employed
- Operational Transformation (OT) use to ensure CCI
  - CCI defined (next)
Background: CCI Defined

1. **Convergence**: when all sites have executed the same set of operations, the copies of the shared document at all sites are identical.

2. **Causality-preservation**: for any pair of operations $O_a$ and $O_b$, if $O_a \rightarrow O_b$, then $O_a$ is executed before $O_b$ at all sites.

3. **Intention-preservation**: for any operation $O$,
   (a) both the local and remote execution effects of $O$ equal to the intention of $O$, and
   (b) if there exists an operation $O_x$ such that $O_x \parallel O$, then the execution effect of $O_x$ does not interfere with the execution effect of $O$, and vice versa.

**OT Example**

<table>
<thead>
<tr>
<th>Site 1</th>
<th>Site 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>State = “acd”</strong></td>
<td><strong>State = “acd”</strong></td>
</tr>
<tr>
<td>$O_1$=ins(1, ’b’))</td>
<td>$O_2$=del(1)</td>
</tr>
<tr>
<td><strong>State = “abcd”</strong></td>
<td><strong>State = “ad”</strong></td>
</tr>
<tr>
<td>$O_2$=del(1)</td>
<td>$O_1$= $O_1$’</td>
</tr>
<tr>
<td><strong>State = “acd”</strong></td>
<td><strong>State = “abd”</strong></td>
</tr>
</tbody>
</table>

*Incorrect w/out transformation*

*Correct with transformation*

$O_1$ is concurrent to $O_2$
Our Previous Work

- **Dynamic, hierarchical locking**
  - Manage a tree representing the shared document
  - Locks are automatically moved up and down the tree to allow multiple users access

- **Previous simulation demonstrated dynamic locking can reduce communication costs significantly**
  - But had write request fails (locks demoted to leaf nodes, but then rejected thereafter)
  - Caused some users to be blocked from editing
Previous Work: Document Tree

Title (t_{artif})
Paragraph A (p_a)
  Title A1 (t_{a1})
    Paragraph A1 (p_{a1})
      Paragraph A11 (p_{a11})
      Paragraph A12 (p_{a12})
    Paragraph A2 (p_{a2})
    Paragraph A3 (p_{a3})
Paragraph B (p_b)
...

[Diagram of a document tree structure with nodes and edges representing the hierarchical structure of the text.]
Message Types

- **Document Check-out (CO)** – the client would like to check out and become a reader of a document.
- **Document Check-in (CI)** – the client is no longer interested in the document and releases it.
- **Lock Request (LK)** – the client wants to write to a section of the document
- **Unlock (ULK)** – the client has left the section and no longer needs the ability to write to it
- **Promotion (P)** – informs the user that he now owns more of the document that he previously owned.
- **Demotion (D)** – informs the user that he now owns less of the document that he previously owned.
- **OT Added (OTA)** – signals a user within a section that another user has been added to the section and future changes must be sent to this new user
- **OT Deleted (OTD)** – signals a user within a section that a user has left the section and no longer needs to have changes sent to him
- **OT Join (OTJ)** – tells the user requesting a lock that he has been granted write access to a section that is already using OT; this message contains a list of the existing users within the section so that the new user can send future changes to these users
- **OT Modify (OTM)** – this message tells a client that the section has been modified and a local OT must be performed based upon the operation being communicated.
Simulation Design Process

- Simulate both clients and server
- Simulate clients and use real server
- Use real clients and real server
Models

- **Client**
  - Behaviors (states – reading or writing)
  - Internal state transition function changes
    - Based upon simulated user actions
  - External state transition function changes
    - Based upon messages from clients and server

- **Server**
  - Repository
  - Lock proxy (integrates with legacy repositories) and implements our algorithms
  - Only external state transition function implemented

- **Network**
  - Simple (all messages arrive in constant time)
  - Could make more complex (not focus)
  - Used to monitor and count message types
This configuration shows one server and three clients.
Simulation Configurations

- 6 different document structures
  - Varied depth and number of leaves
- 2 different number of clients
  - Used 3 and 9 clients
- 4 different client behavior patterns
  - Random (R)
  - Clustered (C)
  - Hybrid (H)
  - Uniform distribution of R, C, and H
Document Structures

<table>
<thead>
<tr>
<th>Document Structure</th>
<th>Number of Leaves</th>
<th>Maximum Depth</th>
<th>Average Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>3</td>
<td>2.75</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>4</td>
<td>2.875</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
<td>4</td>
<td>2.875</td>
</tr>
<tr>
<td>4</td>
<td>48</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>96</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>192</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Document structures 5 and 6 represent 4 and 8 page conference papers (respectively) if paragraphs are used as the leaf nodes in the document tree.
## Simulation Configurations

<table>
<thead>
<tr>
<th>Simulation Configuration</th>
<th>Number of Clients</th>
<th>Document Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>12</td>
<td>9</td>
<td>6</td>
</tr>
</tbody>
</table>
Results

- Since all messages are broadcast to all users other than the originating user in a pure OT system, we define the number of messages generated in a pure OT system as
  \[ M_{\text{PureOT}} = (n - 1)W \]
  where \( n \) is the number of users and \( W \) is the number of write requests (the number of times users modified the document).

- The relative message overhead, \( Mo \), of our dynamic lock OT system is defined as
  \[ Mo = \frac{LK + ULK + P + D + OTA + OTD + OTJ + OTM}{M_{\text{PureOT}}} \]

- We ignore \( LK \) and \( ULK \) since these are constant in both pure OT and our system.

- Thus a relative message overhead of 1 reflects the dynamic lock with OT system incurs the same number of communication cost as a pure OT system. \( Mo \) above 1 reflects our system incurs more communication that a pure OT system. \( Mo \) below 1 reflects our system incurs less communication than a pure OT system. Thus a lower value is a reduction in communication costs.
Results: Dynamic Message Overhead

Dynamic OT Message Rate = % messages dealing with OT relative to all messages

![Graph showing Dynamic OT Message Rate as Collaboration Density Increases]

- Document 1
- Document 2
- Document 3
- Document 4
- Document 5
- Document 6
Results: Communication Efficiency

- Odd simulations = 3 users
- Even simulations = 9 users
Results: 3 Users

Client Edit Behavior and Communication Efficiency Relative to Pure OT - 3 Users

<table>
<thead>
<tr>
<th>Document Structure</th>
<th>Number of Leaves</th>
<th>Maximum Depth</th>
<th>Average Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>3</td>
<td>2.75</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>4</td>
<td>2.875</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
<td>4</td>
<td>2.875</td>
</tr>
<tr>
<td>4</td>
<td>48</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>96</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>192</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
Results: 9 Users

**Document Structure**

<table>
<thead>
<tr>
<th>Document Structure</th>
<th>Number of Leaves</th>
<th>Maximum Depth</th>
<th>Average Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>3</td>
<td>2.75</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>4</td>
<td>2.875</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
<td>4</td>
<td>2.875</td>
</tr>
<tr>
<td>4</td>
<td>48</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>96</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>192</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
Results: 18 Users

Client Edit Behavior and Communication Efficiency Relative to Pure OT - 18 users

Document Structure

<table>
<thead>
<tr>
<th>Document Structure</th>
<th>Number of Leaves</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>48</td>
</tr>
<tr>
<td>5</td>
<td>96</td>
</tr>
<tr>
<td>6</td>
<td>192</td>
</tr>
</tbody>
</table>
Real Server
Real Client Editor

[Diagram of a software interface with labels:
- Editing Space
- Tree Structure
- Location State
- Messages from Server]
Conclusions

- Simulation design and implementation process useful
  - Allows rapid testing/validation of approach
- Communication costs can be reduced using dynamic locking
  - Larger documents
  - Clustered editing
  - Larger number of clients/collaborators