Applying Broadcasting/Multicasting/Secured Communication to agentMom in Multi-Agent Systems

MSE Presentation 2
Chairoj Mekprasertvit

Overview

- Action Items
- agentMom Revisited
- Architecture Design
- Formal Requirement Specification
- Test Plan
- Formal Technical Inspection
- Implementation Plan
- Demonstration

Action Items

- Use Cases
- Cost Estimation

Action Items: Use Cases

Before

After

Action Items: Cost Estimations

- Pre: Estimation 1500 SLOC
- Function Point Analysis and COCOMO

<table>
<thead>
<tr>
<th>Type</th>
<th>Complexity</th>
<th>Function Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal Logical Files</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>External Interface Files</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>External Inputs</td>
<td>50</td>
<td>15</td>
</tr>
<tr>
<td>External Outputs</td>
<td>Set</td>
<td>10</td>
</tr>
<tr>
<td>External Inquiry</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>56</td>
<td></td>
</tr>
</tbody>
</table>

Effort = 3.2*(1.61)^1.05 = 5.3 person-months
Time = 2.5*(5.3)^0.38 = 4.7 months
Productivity = 1610/4.7 = 343 LOC-month
SLOC = 1612
Staff = 5.3/4.7 = 1.13 person
Boehm: 7hrs/day and 22 days/month
Time = 100 days with 7hrs/day
agentMom Revisited

- Communication framework for multi-agent systems
- Provides basic building blocks for building agents, conversation between agents and messages that are passed in conversation.
- Implemented in Java
- Using TCP/IP

Overall Design

Formal Requirement Specification

<table>
<thead>
<tr>
<th>OCL pre-post condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. OCL - pre condition</td>
</tr>
<tr>
<td>2. OCL - post condition</td>
</tr>
</tbody>
</table>

Unicast Conversation

- Association ConstructUnicast between Conversation and Message
- Association ReceiveUnicast between Conversation and Message

context Conversation:: sendMessage(m: Message)
pre cond_1: self.createdMessage = m
pre cond_2: m.isDefined
post cond_3: Conversation.allInstances -> exists(c: Conversation| c.Localhost = self.connectionHost and c.createMessage = m and c.receiveMessage = m)
implies c.Localhost = self.connectedHost
Formal Requirement Specification

- context Conversation::receiveMessage(): Message
  -- New received message is created
  post cond_1: self.receivedMessageoclIsNew = true
  -- New created received message is the same as sent Message
  post cond_2: self.receivedMessage.content = self.receiveMessage@pre.content
  -- Result of receiveMessage()
  post cond_3: result = self.receivedMessage
  -- Only the destined receiver receives message
  post cond_4: self.Localhost = self.receiveMessage@pre.createdByUnicast.connectionHost

Sequence Diagram

- Sending Unicast
- Receiving Unicast
- Sending Secured Unicast
- Receiving Secure Unicast
- Sending Multicast
- Receiving Multicast
- Sending Broadcast
- Receiving Broadcast
Sequence Diagram: Sending Multicast

Sequence Diagram: Receiving Multicast

Sequence Diagram: Sending Broadcast

Sequence Diagram: Receiving Broadcast

Test Plan
- Unit Testing
  - Sending unicast message
  - Sending multicast message
  - Sending Broadcast Message
  - Sending secured unicast message
  - Sending secured multicast message
  - Subscribe to multiple multicast group
  - Receiving unicast message
- Receiving multicast message
- Receiving broadcast message
- Receiving secured unicast message
- Receiving secured multicast message
- Encrypting message
- Decrypting message

Test Plan (cont.)
- Integration testing
  - Subscribe to multiple multicast group plus sending and receiving multicast message
  - Subscribe to multiple multicast group plus sending and receiving secured multicast message
- System testing
- Building simple multi-agent systems that use all types of communications.
- Compatibility Testing
- New agentMom is compatible with agentMom 1.2
Formal Technical Inspection

- 1. All the symbols used in the use case diagram conforms to the UML standard.
- 2. All the symbols used in class diagram conforms to UML standard.
- 3. All the symbol used in Sequence diagram conforms to UML standard.
- 4. If there is a message passing between objects in sequence diagram, association relationship in class diagram is defined. 
- 5. Each message in sequence diagram is a method in class diagram.
- 6. Use case scenarios and description are clear.
- 7. Class diagram and description are clear.
- 8. Sequence diagram and description are clear.
- 9. Names used in class diagram indicated their meaning.
- 10. The defined public attributes should be accessible to the outside class.
- 11. The defined private attributes should be accessible only within the class.

Implementation Plan

- Update document - 4 days
- Component Design - 7 days
  - completed class diagram and sequence diagram
- Coding - 10 days
  - Unit Testing - 4 days
  - Building stub module and performing test
- Integration Testing - 4 days
  - Building stub module and performing test
- System Testing - 4 days
  - Building simple multi-agent systems
- User Manual - 2 days
  - how to use the framework, source code and example
- JavaDoc - 1 day
  - standard JavaDoc API
- Assessment Evaluation - 1 day
- Project Evaluation - 2 days
- Formal Inspection Letters - 1 day
- References - 1 day
- Total – 40 days

Demo:

- Client agent request a prime number at position N, N is an input supplied to Server agents.
- Four types of communication, unicast, secured unicast, multicast, broadcast.
- Set up:
  1) one client agent
  2) three servers, two of them subscribe to the same multicast group.

Conclusion

- Action Items
- agentMom Revisited
- Architecture Design
- Formal Requirement Specification
- Test Plan
- Formal Technical Inspection
- Implementation Plan
- Demonstration