```
-- Description: Formal Requirement Specification based on agentMom's
-- Architecture design using UML/OCL methodology.
-- We want to formalize to show that our model holds the following properties by
-- defining the pre and post conditions:
-- 1.) Unicast conversation
-- 1.1) Only the specified address receives the unicast message
-- 1.2) Sent message is the same as received message
-- 2.) Multicast conversation
-- 2.1) Only the specified group receives the multicast message for that group
-- 2.2) Sent message is the same as received message
-- 3.) Broadcast conversation
-- 3.1) Only the conversations holding the same broadcast address receive the
-- broadcast message
-- 3.2) Sent message is the same as received message
-- In this model we assume that the underlying physical communication is
-- reliable.
-- Project: Applying Broadcast/Multicast/Secured Communication to agentMom in
-- Multiagent Systems
-- Author: Chairoj Mekprasertvit
-- File: agentMom_ocl.use
-- Course: CIS895 MSE Project 2003
-- Project Advisor: Dr. Scott A. DeLoach
-- Department of Computing and Information Sciences
-- Kansas State University
-- version 1.1 11-23-2003
model agentMom

class MomObject
attributes
name: String;
port: Integer;

broadcast_port: Integer;
secure_unicast_port: Integer;
operations
end

class Agent < MomObject
attributes
operations
end

class Component < MomObject
attributes
operations
end

class MessageHandler
attributes
operations
end

class Message
attributes
content: String;
force: String;
host: String;
```

```
inreplyto: String;
language: String;
ontology: String;
performative: String;
port: Integer;
receiver: String;
replywith: String;
sender: String;
end

class Conversation
attributes
m: Message;
Localhost: String;
connectionHost: String;
connectionPort: Integer;
operations
sendMessage(m: Message)
receiveMessage(): Message
end

class MulticastConversation
attributes
multicastPort: Integer;
m: Message;
join: Boolean;
multicastAddress: String;
operations
sendMessage(m: Message)
sendJoin()
sendLeave()
receiveMessage(): Message
end

class BroadcastConversation
attributes
broadcastPort: Integer;
m: Message;
broadcastAddress: String;
operations
sendMessage(m: Message)
receiveMessage(): Message
end

class SecureUnicastConversation
attributes
Localhost: String;
connectionHost: String;
connectionPort: Integer;
m: Message;
operations
sendMessage(m: Message)
receiveMessage(): Message
end


-- Associations
```

```
association Agent-Conversation between
      Agent[1] role agent
      Conversation[0..*] role unicastConversation
end

association Agent-MulticastConversation between
      Agent[1] role agent
      MulticastConversation[0..*] role multicastConversation
end

association Agent-BroadcastConversation between
      Agent[1] role agent
      BroadcastConversation[0..*] role broadcastConversation
end

association Agent-SecureUnicasttConversation between
      Agent[1] role agent
      SecureUnicastConversation[0..*] role secureUnicastConversation
end

association ConstructUnicast between
      Conversation[0..1] role createdByUnicast;
      Message[0..1] role createdMessage;
end

association ReceiveUnicast between
      Conversation[0..1] role receivedByUnicast;
      Message[0..1] role receivedMessage;
end

association ConstructMulticast between
      MulticastConversation[0..1] role createdByMulticast;
      Message[0..1] role createdMessage;
end

association ReceiveMulticast between
      MulticastConversation[0..1] role receivedByMulticast;
      Message[0..1] role receivedMessage;
end

association ConstructSecureUnicast between
      SecureUnicastConversation[0..1] role createdBySecured;
      Message[0..1] role createdMessage;
end

association ReceiveSecureUnicast between
      SecureUnicastConversation[0..1] role receivedBySecured;
      Message[0..1] role receivedMessage;
end

association ConstructBroadcast between
      BroadcastConversation[0..1] role createdByBroadcast;
      Message[0..1] role createdMessage;
end

association ReceiveBroadcast between
```

```
      BroadcastConversation[0..1] role receivedByBroadcast;
      Message[0..1] role receivedMessage;
end


-- Constraints

constraints

-- Pre - Post Conditions
-- Send unicast pre-post condition
-- Only Specified agent receives message
context Conversation::sendMessage(m: Message)
-- unicast conversation associates with the Message parameter
      pre cond_1: self.createdMessage = m
-- Message must be well defined before sending
      pre cond_2: m.isDefined
-- Only the destined address and port receive the message.
      post cond_3: Conversation.allInstances->
                              exists(c: Conversation|
                                    ((c.Localhost = self.connectionHost
                                    and
                                    c.agent.port = self.connectionPort)
                                    implies
                                    c.receivedMessage = m)
                                    and
                                    (c.receivedMessage = m
                                    implies
                                    (c.Localhost = self.connectionHost
                                    and
                                    c.agent.port = self.connectionPort)))

-- Receive unicast pre-post condition
-- Received message is the same as sent message
context Conversation::receiveMessage(): Message
-- New received message is created
      post cond_1: self.receivedMessage.oclIsNew = true
-- New created received message is the same as sent Message
      post cond_2: Conversation.allInstances->
                              exists(c: Conversation|
                                    ((c.connectionHost = self.Localhost
                                    and
                                    c.connectionPort = self.agent.port)
                                    implies
                                    c.createdMessage = self.receivedMessage)
                                    and
                                    (c.createdMessage = self.receivedMessage
                                    implies
                                    (c.connectionHost = self.Localhost
                                    and
                                    c.connectionPort = self.agent.port)))
-- Result of receiveMessage()
      post cond_3: result = self.receivedMessage

-- Send secured unicast pre-post condition

context SecureUnicastConversation::sendMessage(m: Message)
```

```
-- secured unicast conversation associates with the Message parameter
      pre cond_1: self.createdMessage = m
-- Message must be well defined before sending
      pre cond_2: m.isDefined
-- Only the address that the message is destined to receives the message.
      post cond_3: SecureUnicastConversation.allInstances->
                              exists(c: SecureUnicastConversation |
                                    ((c.Localhost = self.connectionHost
                                    and
                                    c.agent.secure_unicast_port =
                                          self.connectionPort)
                                    implies
                                    c.receivedMessage = m)
                                    and
                                    (c.receivedMessage = m
                                    implies
                                    (c.Localhost = self.connectionHost
                                    and
                                    c.agent.parent.secure_unicast_port =
                                          self.connectionPort)))

-- Receive secured unicast pre-post condition
context SecureUnicastConversation::receiveMessage(): Message
-- New received message is created
      post cond_1: self.receivedMessage.oclIsNew = true
-- New created received message is the same as sent Message
      post cond_2: SecureUnicastConversation.allInstances ->
                              exists(c: SecureUnicastConversation |
                              ((c.connectionHost = self.Localhost
                                    and
                                    c.connectionPort =
                                          self.agent.secure_unicast_port)
                                    implies
                                    c.createdMessage = self.receivedMessage)
                                    and
                                    (c.createdMessage = self.receivedMessage
                                    implies
                                    (c.connectionHost = self.Localhost
                                    and
                                    c.connectionPort =
                                          self.agent.secure_unicast_port)))
-- Result of receiveMessage()
      post cond_3: result = self.receivedMessage

-- Send multicast pre-post condition
context MulticastConversation::sendMessage(m: Message)
-- Multicast conversation associates with the Message parameter
      pre cond_1: self.createdMessage = m
-- Message must be well defined before sending
      pre cond_2: m.isDefined
-- Need to subscribe to the multicast group first
      pre cond_3: self.join = true
-- All conversations that have the same multicast address and port receives the
-- message, including itself.
      post cond_4: MulticastConversation.allInstances->
                              forAll(c: MulticastConversation|
                                    ((c.multicastAddress = self.multicastAddress
```

```
                                    and
                                    c.multicastPort = self.multicastPort)
                                    implies
                                    c.receivedMessage = m)
                                    and
                                    (c.receivedMessage = m
                                    implies
                                    (c.multicastAddress = self.multicastAddress
                                    and
                                    c.multicastPort = self.multicastPort)))

context MulticastConversation::sendJoin()
-- Not in the group
      pre cond_1: self.join = false
-- New received message is created
      post cond_2: self.receivedMessage.oclIsNew = true
-- All conversations that have the same multicast address receives the join
-- groupmessage, including itself.
      post cond_3: MulticastConversation.allInstances->
                              forAll(c: MulticastConversation|
                                    ((c.multicastAddress = self.multicastAddress
                                    and
                                    c.multicastPort = self.multicastPort)
                                    implies
                                    c.receivedMessage = self.receivedMessage)
                                    and
                                    (c.receivedMessage = self.receivedMessage
                                    implies
                                    (c.multicastAddress = self.multicastAddress
                                    and
                                    c.multicastPort = self.multicastPort)))
-- Now join the group
      post cond_4: self.join = true

context MulticastConversation::sendLeave()
-- Already in the group
      pre cond_1: self.join = true
-- New received message is created
      post cond_2: self.receivedMessage.oclIsNew = true
-- All conversations that have the same multicast address receives the leave
-- groupmessage, including itself.
      post cond_3: MulticastConversation.allInstances->
                              forAll(c: MulticastConversation|
                                    ((c.multicastAddress = self.multicastAddress
                                    and
                                    c.multicastPort = self.multicastPort)
                                    implies
                                    c.receivedMessage = self.receivedMessage)
                                    and
                                    (c.receivedMessage = self.receivedMessage
                                    implies
                                    (c.multicastAddress = self.multicastAddress
                                    and
                                    c.multicastPort = self.multicastPort)))
-- Not in the group
      post cond_4: self.join = false
```

```
-- Receive multicast pre-post condition
context MulticastConversation::receiveMessage(): Message
      pre cond_1: self.join = true
-- New received message is created
      post cond_2: self.receivedMessage.oclIsNew = true
-- New created received message is the same as sent
      post cond_3: MulticastConversation.allInstances->
                              exists(c: MulticastConversation|
                                    ((c.multicastAddress = self.multicastAddress
                                    and
                                    c.multicastPort = self.multicastPort)
                                    implies
                                    c.createdMessage = self.receivedMessage)
                                    and
                                    (c.createdMessage = self.receivedMessage
                                    implies
                                    (c.multicastAddress = self.multicastAddress
                                    and
                                    c.multicastPort = self.multicastPort)))
-- Result of receiveMessage()
      post cond_4: result = self.receivedMessage


-- Broadcast message is received by all broadcast conversation that has the same
-- broadcast address, which is the same local network.
context BroadcastConversation::sendMessage(m: Message)
-- Broadcast conversation associates with the Message parameter
      pre cond_1: self.createdMessage= m
-- Message must be well defined before sending
      pre cond_2: m.isDefined
-- All conversations that have the same broadcast address and port receive the
-- message, including itself.
      post cond_3: BroadcastConversation.allInstances->
                              forAll(c: BroadcastConversation|
                                    ((c.broadcastAddress = self.broadcastAddress
                                    and
                                    c.broadcastPort = self.broadcastPort)
                                    implies
                                    c.receivedMessage = m)
                                    and
                                    (c.receivedMessage = m
                                    implies
                                    (c.broadcastAddress = self.broadcastAddress
                                    and
                                    c.broadcastPort = self.broadcastPort)))

-- Received broadcast message is the same as sent message
context BroadcastConversation::receiveMessage(): Message
-- New received message is created
      post cond_1: self.receivedMessage.oclIsNew = true
-- New received message is created
      post cond_2: self.receivedMessage.oclIsNew = true
-- New created received message is the same as sent
post cond_3: MulticastConversation.allInstances->
                              exists(c: BroadcastConversation|
                                    ((c.broadcastAddress = self.broadcastAddress
                                    and
```

```
                                    c.broadcastPort = self.broadcast Port)
                                    implies
                                    c.createdeMessage = self.receivedMessage)
                                    and
                                    (c.createdMessage = self.receivedMessage
                                    implies
                                    (c.broadcastAddress = self.broadcastAddress
                                    and
                                    c.broadcastPort = self.broadcastPort)))
-- Result of receiveMessage()
      post cond_3: result = self.receivedMessage
```