agentToolIII (Static)

MSE Project Presentation 2

Presented By:
Deepti Gupta

Agenda

- Action Items from Presentation 1
- Project Plan
- Architecture Design
- Formal Specification
- Formal Inspection Checklist
- Test Plan
- Demo
Action Items from Presentation 1

- CVS Setup
- XML Standards for Agent Oriented Systems

Project Plan – Cost Estimate

Current Status
- 136 hours on the project totally (Phase 1 & 2)
- 75 hours coding/debugging
- 25 hours documentation
- 36 hours miscellaneous research/reading
- About 2000 LOC
- 40% features implemented
- 5 documents complete
**Project Plan – Cost Estimate**

- **Productivity**
  - 2000 LOC / 75 = 26.66 LOC/hour
  - 5 documents / 25 = 0.2 docs/hour

- **Remaining work**
  - 2000 / 0.4 = 3000 LOC
  - 4-5 documents

**Project Plan – Cost Estimate**

- **Remaining effort**
  - 3000 LOC / (26.66 LOC/hour) = 113 hours
    = 23 days (5 hours per day)
  - Estimated Testing and test documentation time 40 hours
    = 8 days
  - Estimated 40 hours for documentation
    = 8 days
Project Plan – Cost Estimate

- High level WBS
  - Coding/Debugging/Unit testing – 23 days
  - Overall testing – 8 days
  - Documentation – 8 days

- Total – 39 days

Architecture Design

- Use of GEF for agentToolIII (Static)
- Eclipse plugin
- GEF overview
- GEF / Plugin specific main classes
  - Editor class – entry point
  - Plugin class – makes the application an Eclipse plugin
- GEF links
Architecture Design

- agentTool III (Static) Package view

![Architecture Design - Model](image)
Architecture Design - View

- Drawn Figure
- Goal Figure
- Role Figure
- Agent Figure
- Organization Figure
- Actor Figure
- Capability Figure
- Service Figure

Architecture Design - Controller

- EditParts
- Edit Policies
- Requests
- Commands
Architecture Design

- Sequence Diagram

Formal Specification

```
--
-- Unique names. Goal Names should be unique
--
context o:Diagram
inv UniqueGoals:
o.hasGoals->forAll(g1, g2 | g1 <> g2 implies g1.name <> g2.name)
--
-- OPERATIONS
--
-- Adding a goal
-- Added object must be new
-- Object should be added to the list of goals in the diagram
--
context Goal::addGoal(o : Diagram): Set(Goal)
pre cond1 : o.hasGoals->excludes(self)
pren cond2 : o.hasGoals = o.hasGoals@pre->union(Set(self))
pren cond3 : result = o.hasGoals
```
Formal Specification

-- Deleting a goal
-- Object to be deleted must exist in the organization
-- Object should be deleted from the list of goals in the diagram
--
context Goal::deleteGoal(o : Diagram): Set(Goal)
  pre cond1 : o.hasGoals->includes(self)
  post cond2 : o.hasGoals = o.hasGoals@pre->excluding(self)
  post cond3 : result = o.hasGoals

Formal Specification

-- Adding a sub goal
-- The goals to be connected via the parent-child relationship should both exist in the diagram
-- The name of the sub-goal should become one more than the child number of the parent
-- The relationship type of all sub goals of the parent goal should be same.
-- AND / OR connections not allowed together.
context Goal::addSubGoal(o : Diagram, g: Goal, c: AndORRelationship)
  pre condpre1 : o.hasGoals->includes(self)
  pre condpre2 : o.hasGoals->includes(g)
  pre condpre3 : c.parentGoal = self
  pre condpre4 : AndORRelationship.allInstances->select(c1 | c1.parentGoal = self)->forAll(c2 | c2.type = c.type)
  post condpost1 : g.name = self.name->concat('.')->concat(self.currentChildNumber)
  post condpost2 : c.subGoal = g
### Inspection Checklist

- The symbols used in the class diagrams conform to UML standards
- The symbols used in the sequence diagrams conform to UML standards
- The classes in the class diagrams have a corresponding description provided in the architectural design document
- The descriptions of the classes in the Architecture Design document are clear and adequate.
- Classes in the Architecture Design (Section 6.1 – only relevant to Goal Diagram) are consistent with classes in the USE model
- The attributes in the USE model are consistent with the attributes of the corresponding class diagrams
- The associations in the USE model are present in the Class Diagram as association links.
- The multiplicities in the USE model are consistent with the Class diagram
- The sequence diagrams are clear and understandable based on the description in Section 8.
- All model elements and relationships as outlined in the Vision Document (Section 3.2) are present in the architecture document as classes.

### Test Plan

- Mainly functional black-box testing
- User actions will be simulated through a set of test scenarios.
- Debug logs inspection for testing correct working of object model manipulation
- Unit Testing of each diagram/plugin
- Overall testing based on scenarios/test cases
Test Plan

Sample test case

- Tasks
  - Bring up a diagram by clicking on it
  - Create a new diagram by right clicking on the root
  - Delete an existing diagram by right clicking on it

- Results expected
  - The selected diagram should appear on the canvas
  - The tester should be able to create a new diagram. An empty canvas should appear for drawing the diagram
  - The deleted diagram should no longer be visible in the hierarchy

Prototype Demonstration