Environment Model Building Tools

MSE Presentation 2
Esteban Guillen
Outline

- Action Items
- Project Plan
- Architecture Design
- Formal Specification
- Formal Inspection Checklist
- Test Plan
- Questions
- Demo (Terrain Builder)
Action Items

- Added detail to use case diagram
- Though more about searching
- Identified features to be implemented
New Use Case Diagram
Searching

- Libraries will be a hierarchy of directories searchable by a JTree
- A preview window will show a selected library item as a 3D thump-nail
Features to be Implemented

- Load Objects (search and preview)
- Load Terrain (search and preview)
- Move Objects
- View in 3D
- Save Model
- Load Model
EMB Building Surface

BUILDING SURFACE

OBJECT PREVIEW

TERRAIN PREVIEW
Features to be Implemented

EOB

- Build from primitives (move, resize, set properties)
- Build from saved objects
- Search and Preview
- View in 3D
- Save (collection of primitives)
- Load
EOB Building Surface
Features to be Implemented

**ETB**

- Modify Elevation
- Set properties (texture mapping)
- View in 3D
- Save (elevation map, triangle coords)
- Load
ETB Building Surface
Project Plan – Cost Estimate

- Current Progress
  - 152 total hours (Phase 1 & 2)
  - 90 hours coding/debugging/testing
  - 62 hours documentation
  - 1200 SLOC
  - 25% of implemented features
  - 5 Documents
Project Plan – Cost Estimate

- Productivity
  - 1200 SLOC / 90 hours = 13.2 SLOC/hour
  - 5 Documents / 62 hours = .08 Docs/hour

- Remaining Work
  - 1200 SLOC / .25 = 4800 SLOC
  - 4-5 Documents
Project Plan – Cost Estimate

- Remaining Effort
  - 3600 SLOC / (13.3 SLOC /hour) = 270.7 hours or 39 days (7 hours per day)
  - 56 hours for documentation = 8 days
Project Plan – WBS

- High level
  - Coding/debugging – 30 days
  - Testing – 9 days
  - Documentation – 8 days
Project Plan - WBS

- Deliverables for Presentation 3
  - Action Items (documentation)
  - User Manual (documentation)
  - Component Design (documentation)
  - Source Code (development)
  - Assessment Evaluation (testing)
  - Project Evaluation (documentation)
  - References (documentation)
  - Technical Inspection Letters (documentation)
Project Plan

- Development (6/10 – 7/15)
  - ETB
  - EOB
  - EMB
- Testing (7/16 – 7/28)
- Documentation (7/29 – 8/10)
Architecture Design – EMB Package Organization

EMB Application
EMB Controller
EMB Model
EMB View
Architecture Design – EMB Application Package

<table>
<thead>
<tr>
<th>EMBApplication</th>
</tr>
</thead>
<tbody>
<tr>
<td>init()</td>
</tr>
</tbody>
</table>
EMBController

- save(f : File)
- open(f : File)
- parseXML()
- exportXML()
- setZoomFactor()
Architecture Design – EMB View Package

EMBThreeDimensionalView
- add(s : Shape3D)

EMBTerrainPreview
- addTerrain()

EMBBuildingSurface
- paint()
- zoomOut()
- zoomIn()
- select(p : Point2d)

EMBDrawingView
- +threeD
- +canvas

EMBXMLView
- showXML()

EMBObjectPreview
- addObject()

EMBView
- +xml

EMBTerrainFinder
- searchDB()
- getCurrent() : EMBTerrain
- select()

EMBTerrainView
- display(s : Shape3D)

EMBObjectFinder
- searchDB()
- getCurrent() : EMBObject
- select()

EMBObjectView
- display(s : Shape3D)
Architecture Design – EMB Model Package

EMBModel
- getData(): EMBEnvironment
- addObject(o: EMBObject)
- addTerrain(t: EMBTerrain)
- deleteObject(o: EMBObject)
- deleteTerrain(t: EMBTerrain)

EMBBasicShape
- name: String
- x: Double
- y: Double
- z: Double
- writeXML(): String
- addShape(s: EMBBasicShape)
- move(p: Point3d)

EMBObject
- name: String
- x: Double
- y: Double
- z: Double
- length: Double
- width: Double
- height: Double
- writeXML(): String
- addShape(s: EMBBasicShape)
- move(p: Point3d)

EMBEnvironment
- name: String
- writeXML(): String
- addObject(o: EMBObject)
- addTerrain(t: EMBTerrain)
- deleteObject(o: EMBObject)
- deleteTerrain(t: EMBTerrain)

EMBEnvironmentLibrary
- getData(): Set(EMBEnvironment)
- addData(e: EMBEnvironment)

EMBTerrain
- name: String
- getHeight(x: Double, z: Double): Double
- writeXML(): String
- addObject(o: EMBObject)
- addTerrain(t: EMBTerrain)
- deleteObject(o: EMBObject)
- deleteTerrain(t: EMBTerrain)

EMBTerrainLibrary
- getData(): Set(EMBTerrain)
- addData(t: EMBTerrain)

EMBSphere
- radius: Double

EMBCylinder
- height: Double
- radius: Double

EMBBox
- length: Double
- width: Double
- height: Double

EMBObjectLibrary
- getData(): Set(EMBObject)
- addData(o: EMBObject)

EMBConic
- height: Double
- radius: Double
Architecture Design – EMB
Loading an Object

- User
  - select
  - display
  - select
  - display
  - addObject
  - addObject
  - addObject
  - addObject
  - display
  - addObject
  - paint
  - add

- EMBObjectFinder
- EMBObjectView
- EMBObjectPreview
- EMBModel
- EMBEnvironment
- EMBBuildingSurface
- EMBThreeDimensionalView
Architecture Design – EMB

Saving an Environment Model

- User
- EMBController
- EMBModel
- model: EMBEnvironment
- EMBTerrain
- EMBObject
- box1: EMBBox
- sphere1: EMBSphere

getData
save
writeXML
writeXML
writeXML
writeXML
Architecture Design – EOB
Package Organization
Architecture Design – EOB Application Package

<table>
<thead>
<tr>
<th>EOBApplication</th>
</tr>
</thead>
<tbody>
<tr>
<td>init()</td>
</tr>
</tbody>
</table>
Architecture Design – EOB Controller Package

EOBController

save(f : File)
open(f : File)
parseXML()
exportXML()
setZoomFactor()
Architecture Design – EOB View Package

EOBView

+threeD

EOBThreeDimensionalView
add(s : Shape3D)

EOBObjectPreview
addObject()

EOBObjectFinder
searchDB()
getCurrent() : EOBObject
select()

EOBObjectView
display(s : Shape3D)

EOBFrontDrawingView
paint()
zoomOut()
zoomIn()
select(p : Point2d)

EOBSDrawingView
paint()
zoomOut()
zoomIn()
select(p : Point2d)

EOBTopDrawingView
paint()
zoomOut()
zoomIn()
select(p : Point2d)

EOBXMLView
showXML()
Architecture Design – EOB Model Package

EOBObject
- name : String
- x : Double
- y : Double
- z : Double
- length : Double
- height : Double
- width : Double
- writeXML() : String
- addShape(s : EOBBasicShape)
- deleteShape(s : EOBBasicShape)

EOBox
- length : Double
- height : Double
- width : Double
- modify(l : Double, h : Double, w : Double)

EOBCone
- height : Double
- radius : Double
- modify(h : Double, r : Double)

EOBCylinder
- height : Double
- radius : Double
- modify(h : Double, r : Double)

EOBSphere
- radius : Double
- modify(r : Double)

EOBBasicShape
- name : String
- x : Double
- y : Double
- z : Double
- writeXML() : String
- move(p : Point3d)

EOBModel
- getData() : EOBObject
- addShape(s : EOBBasicShape)
- deleteShape(s : EOBBasicShape)

EOBObjectLibrary
- getData() : Set(EOBObject)
- addData(o : EOBBasicShape)
Architecture Design – EOB
Adding Two Box Shapes
Modifying Box Shapes

Architecture Design – EOB

User: EOBFrontDrawingView
- select
- getData
- paint

EOBSideDrawingView: getData
- move
- paint

EOBTopDrawingView: getData
- modify
- paint

EOBModel: box1: EOBBox
- select
- getData
- paint

EOBModel: box2: EOBBox
- select
- getData
- paint
Architecture Design – ETB
Package Organization

- ETB Application
- ETB Controller
- ETB View
- ETB Model
Architecture Design – ETB Application Package

<table>
<thead>
<tr>
<th>ETBApplication</th>
</tr>
</thead>
<tbody>
<tr>
<td>init()</td>
</tr>
</tbody>
</table>
Architecture Design – ETB Controller Package

<table>
<thead>
<tr>
<th>ETBController</th>
</tr>
</thead>
<tbody>
<tr>
<td>save(f : File)</td>
</tr>
<tr>
<td>open(f : File)</td>
</tr>
<tr>
<td>parseXML()</td>
</tr>
<tr>
<td>exportXML()</td>
</tr>
<tr>
<td>setZoomFactor()</td>
</tr>
</tbody>
</table>
Architecture Design – ETB Model Package

ETBModel
- getData() : ETBTerrain

ETBTerrain
- name : String
- writeXML() : String
- updatePoint(i : Integer, p : Point3d)

ETBTerrainLibrary
- getData() : Set(ETBTerrain)
- addData(t : ETBTerrain)

+ terrain 1
+ terrains 1..n
Architecture Design – ETB Modifying Elevation

User

: ETBBuildingSurface

: ETBModel

: ETBTerrain

setElevation

modify

data

updatePoint
Formal Specification

--
--Unique names of objects in Object Library
--
context o : EMBObjectLibrary
inv UniqueNameObjectLibrary:
o.objects->forAll(p1,p2 | p1 <> p2
    implies p1.name <> p2.name)
--
--Unique names of terrains in Terrain Library
--
context t : EMBTerrainLibrary
inv UniqueNameTerrainLibrary:
t.terrains->forAll(p1,p2 | p1 <> p2
    implies p1.name <> p2.name)
Formal Specification

-- Unique names for all shapes of an object
context obj : EMBObject
  inv UniqueNameObjectShapes:
    obj.shapes->forAll(p1,p2 | p1 <> p2 implies p1.name <> p2.name)
--
-- Every box has positive length, width and height
context b : EMBBox
  inv BoxPositiveLength:
    b.length > 0
  inv BoxPositiveWidth:
    b.width > 0
  inv BOXPositiveHeight:
    b.height > 0
Formal Specification

--Deleting an object must remove it while the other object are unchanged
context EMBEnvironment::deleteObject(o : EMBObject)
  pre  Current: objects->includes(o)
  post Deleted: objects = objects@pre->excluding(o)

--Added objects must be unique
context EMBEnvironment::addObject(o : EMBObject)
  pre  Current: objects->excludes(o)
  post Added:   objects = objects@pre->including(o)
1. The symbols used in the class diagrams conform to the UML standards
2. The symbols used in the sequence diagrams conform to the UML standards
3. The class diagrams have a corresponding description provide in the architectural design document
4. The descriptions of all class diagrams are clear and makes sense
5. The messages passed between objects in the sequence diagrams can be found in the corresponding class diagram as public methods
Formal Inspection Checklist

- 6. All classes in the Environment Model Builder (sections 2.2-2.5 of Architecture Design) are found in the USE model (section 5 of the Architecture Design)
- 7. The role names and multiplicities in the USE model match with the role names and multiplicities of the UML diagrams for the Environment Model Builder (sections 2.2-2.5 of Architecture Design)
- 8. The attributes in the USE model match with the attributes of the corresponding class diagrams (sections 2.2-2.5 of the Architecture Design)
- 9. The operations in the USE model match with the corresponding methods in the class diagrams (sections 2.2-2.5 of the Architecture Design)
Test Plan

- Mainly functional testing
- Describe a scenario consisting of test cases that test specific requirements
- Unit testing with JUnit
- Integration testing will be difficult
  - Nobody developing 3D Viewer and Environment Simulator
Test Plan

Scenario for Environment Object Building Tool

- The object builder is intended to build object from primitive shapes. For testing the tester will stack all four types of primitive shapes on top of each other. This collection of primitives will be saved to the database and then reused to build a new shape. The test cases below will provide a step by step process for the scenario.
Test Plan

Test Case 1 – Testing SR12 and SR13

Task(s)
- Load a cone, sphere, box, and cylinder onto the drawing surfaces.

Verification
- The shapes should be visible on the drawing surface.
Test Plan

Test Case 3 – Testing SR15 and SR16

- Task(s)
  - Change the box to red and a weight of 5kg.
  - Change the cylinder to yellow and a weight of 2kg.
  - Change the cone to orange and a weight of 4kg.
  - Change the sphere to blue and a weight of 2kg.

- Verification
  - The box is visibly red and its property box shows a weight of 5kg.
  - The cylinder is visibly yellow and its property box shows a weight of 2kg.
  - The cone is visibly orange and its property box shows a weight of 4kg.
  - The sphere is visibly blue and its property box shows a weight of 2kg.
Questions
