Chapter 6
Case Study: Game2D with Method Design

Objectives of Game2D
- Simple 2 player game
- To be played over the Internet
- The game board consists of a two-dimensional grid
- Dots randomly appear on the grid, and alternate between two colors (red and blue)
- Each player is assigned a color, either red or blue

More Objectives of Game2D
- The objective of the game is for each player to cause dots of his/her color to disappear by clicking on the dots
- A player loses if two neighboring dots of his/her color appears on the grid

Supported Activity List
- Allow registration of two players for each game
- Display a simple table (grid) of initially empty cells
- When two players have been registered and one of them clicks the start button, the game begins
- Randomly display dots in the cells at regular intervals. The dots alternate in color between red and blue
- Allow the player to adjust the interval at which dots are displayed

Supported Activity List Continued
- Assign one of two colors (red or blue) to each player so that each player is concerned with dots of that color only
- Respond to mouse clicks. If either player clicks on a dot, the dot disappears in response to that click
- Keep track of points (the score). Each time a player clicks on a dot of his or her color. He or she earns a point
- If two dots of the same color appear next to each other, the player corresponding to that color loses. Notify each player of their win or loss

Human-Computer Interface
Requirements Analysis

List of nouns
- Player
- Game
- Internet
- Table
- Cell
- Start Game button
- Dot
- Interval
- Color
- Mouse
- Mouse click
- Point (score)
- Loser

Analysis of List of Nouns

- Retained information
- Needed services
- Multiple attributes
- Common attributes
- Common operations
- Essential requirements

Primary Classes

- Player
- Game
- Cell
- Dot

Use Case Development

- Client communicates player sign-on
- Client communicates player action (mouse click)
- Server communicates that dot is removed from cell
- Server communicates that dot is set in cell
- Server communicates that game is won/lost

Use Case: Client Communicates Player Action

Main flow of events:
This use case begins when the player clicks the left mouse button in the applet window. The goal of each player is to erase as many dots of his or her associated color as possible. In other words, the blue player wishes to erase blue dots, while the red player wishes to erase red dots. A player erases a dot by pressing the left mouse button over a cell containing that dot. When the mouse button is clicked, the event is trapped and the coordinates of the mouse event are translated into a specific reference to a Cell object.

Use Case: Client Communicates Player Action (continued)

Main flow of events (continued):
The Cell object is queried to determine if a Dot is present in that Cell and what its color is. If no Dot exists in this Cell, the use case ends. If a Dot matching the player’s color is present, the player receives a point. If a Dot of any color is present, the Dot is removed from the Cell and the information to remove the Dot from the Cell is transmitted to the server. The use case then ends.
Use Case: Client Communicates
Player Action (continued)

Exceptional flow of events:
If a player clicks the mouse on an area of the applet where no Cell object is rendered, the use case ends with no information being transmitted to the server.

(see deliverable 6.11 for associated scenario)

Refined Class List
- Player
- Game2DServer
- Game
- Timer
- Cell
- Dot

An Initial Class Diagram (Partial)

State Machine for Game2D (Partial)

Use Case Diagram for Game2D

Revised Class Diagram for Game2D (Partial)
**Game2D: Interaction Diagram**

(see deliverable 6.20 for complete diagram)

- Game2DServer
  - unsetDot(row, col)
- Game2DClient
  - setColor(color)
  - setDot(row, col, color)
  - unsetDot(row, col)
  - win()
  - lose()

(see deliverable 6.22 for complete diagram)

**Game2D: Object Diagram**

- Game2DServer
- Game2DClient
  - Game2DClientListener
    - Applet
    - Socket
  - gridCell[row][col]
    - X=0
    - Y=0
    - width=15
    - height=15
  - color = 1
  - hasDot=false
  - color= 0

(see deliverable 6.22 for complete diagram)

**Game2D: Class Skeleton**

(see deliverable 6.23 for complete skeleton)

```java
public class Game2DClient extends Applet implements MouseListener, ActionListener
{
  // Class semantics and roles
  // This class embodies the GUI which the player interacts
  // Instance variables:
  private Graphical_Cell grid[][]; // 2D array of cells: game state
  private int color;  // the player’s color 1 = blue, 2 = red
  private Socket sock;  // Socket connection to game server
  private PrintWriter out;  // output stream built over socket connection

  // Nonstatic Methods
  public void paint(Graphics g)
    // Override Applet method to render the array of graphic cells
  public void setDot(int i, int j, int c)
    // send message to graphical cell object at row i, column j array grid
    // set color to Color.blue, if c = 1 otherwise set it to Color.red
    // precondition: 2D array of graphical cells has been allocated and c
    // c has value of 1 or 2. Postcondition: either a red or blue dot appears
    // in the graphic cell grid[i][j].
}
```

**Method Design**

- Techniques for specifying algorithms
  - Nassi-Shneiderman charts
  - Flowcharts
  - Pseudocode
  - Decision tables

**Decision Table and Pseudocode Example**

- Pick random number between 0 and MAX-1
- Set dot in cell grid[i][j] to color c

<table>
<thead>
<tr>
<th>grid[i+1][j] has dot color c</th>
<th>player c loses</th>
</tr>
</thead>
<tbody>
<tr>
<td>grid[i-1][j] has dot color c</td>
<td>player c loses</td>
</tr>
<tr>
<td>grid[i][j+1] has dot color c</td>
<td>player c loses</td>
</tr>
<tr>
<td>grid[i][j-1] has dot color c</td>
<td>player c loses</td>
</tr>
</tbody>
</table>
Creating Quality Methods

- Cohesive
- Loosely coupled
- Encapsulated
- Reusable