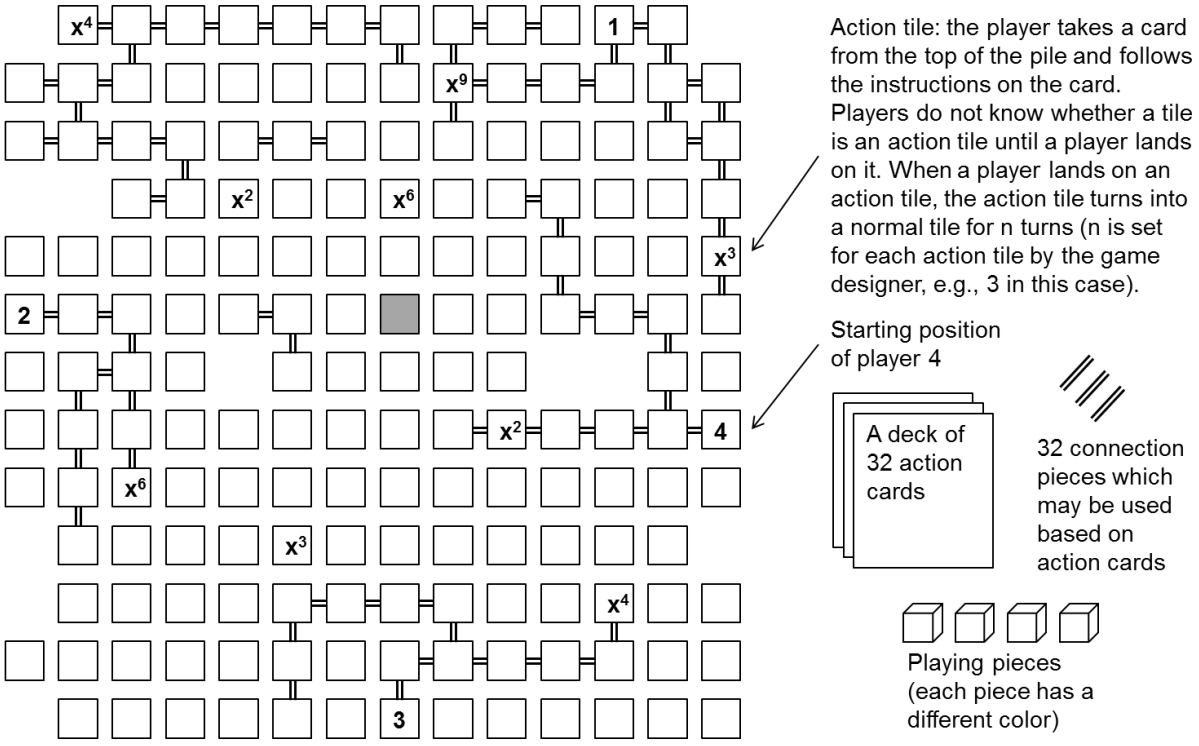
Project Overview

You will create the ***Tile-O*** application that first allows a game designer to design a board game and then allows players to play the game. The objective of the game is to find the hidden tile (indicated in gray in the figure below but shown as a regular tile to players during a game). Two to four players take turns by moving their playing pieces along connected tiles based on the roll of a die.

A designer first defines the whole game including the layout of the game board. The designer places the tiles on the board and connects them with connection pieces. In addition, the designer indicates the hidden tile (shown in gray), the starting positions of each player, as well as the location of action tiles.



The designer also defines a deck of 32 action cards by choosing from the following predefined choices:

- Roll the die for an extra turn
- Connect two adjacent tiles with a connection piece from the pile of spare connection pieces
- Remove a connection piece from the board and place it in the pile of spare connection pieces
- Move your playing piece to an arbitrary tile that is not your current tile
- Lose your next turn

Players take turns, with Player 1 starting the game, followed by Player 2, Player 3 (if applicable), and Player 4 (if applicable). The player whose turn it is rolls the die and them moves her playing piece along connected tiles. If the player lands on any tile, the color of the tile changes from white to black to indicate that the tile has been visited during the game. If the player lands on the hidden tile, the game ends and the player wins the game. If the player lands on an action tile, the player takes the first action card from the deck of action cards and follows the instructions on the action card. In addition, the action tile turns into a regular tile for a number of turns as specified by the game designer.

1 Deliverables

You will deliver the ***Tile-O*** application in several iterations throughout the term. For each iteration, one or more deliverables are to be submitted as described below. The project is done in a ***team of six students***. If the number of students in the class is not divisible by six, then some teams may consist of five or seven students with the instructor's permission. Generally, the whole team is responsible for each deliverable, except for Parts 2/3/7 where individual team members are responsible for some specific deliverables.

Deliverables for Part 1 – Domain Model (4%) (due Monday, January 30, 2017 23:30)

- Use Umple to define the domain model showing all concepts and relationships
- Generate code from domain model

Deliverables for Part 2 – UI Mockup, Sequence Diagrams, Controller Interfaces (4%) (due Friday, February 10, 2017 23:30)

- Assign the development of the application features to your team members, i.e., each team member is individually responsible for a set of features (inactivity period of an action tile and action card "Lose your next turn" are not required for this deliverable)
- For each feature, create a mockup of the user interface, define a sequence diagram showing how the feature will be implemented, and specify the Controller interface as needed to realize the feature

Deliverables for Part 3 – Implementation (8%) (due Friday, February 24, 2017 23:30)

- Implement the assigned features individually and as a team in Java as described in Part 2

Deliverables for Part 4 – State Machines (4%) (due Friday, March 10, 2017 23:30)

- Use Umple to describe two features (inactivity period of an action tile and action card "Lose your next turn") with state machines for the domain classes
- For each feature, create a mockup of the user interface, define a sequence diagram showing how the feature will be implemented, and specify the Controller interface as needed to realize the feature
- Generate code from the state machines

Deliverables for Part 5 – State Machine Integration (4%) (due Friday, March 17, 2017 23:30)

- Implement the new features in Java as described in Part 4

Deliverables for Part 6 – Expand Application (4%) (due Friday, March 24, 2017 23:30)

- Come up with a new action card for the game
- Describe what is needed from the user interface to realize the new action card
- If applicable, update the domain model and state machines to realize the new action card
- Implement the Controller code for the new action card in Java
- Share the description of the new action card, the description of the UI, the updated domain model and state machines (if applicable), and the Controller code with all teams

Deliverables for Part 7 – Final Application (8%) (due Thursday, April 6, 2017 23:30)

- Select one different action card for each team member from the new action cards from Part 6, i.e., each team member is individually responsible for one new action card
- Update the domain model and state machines as needed for the chosen action cards
- Implement individually and as a team all chosen action cards in Java

Deliverables for Part 8 – Demo (4%) (due Friday/Monday/Tuesday April 7/10/11, 2017 during tutorial/class)

- Give a demo of your application to the class
- Time slots for the demo will be available for selection on Friday, March 10, 2017 10:00 on a first-come-first-serve basis

2 Technology Constraints

Your ***Tile-O*** application must be implemented in Java with a suitable Java framework for the user interface. Your domain model and state machines must be specified with Umple and code generated from them with Umple. You must use the generated code in your application. The data of your application must be saved in an XML file with the help of the provided XStream libraries.

3 General Rules

Project Reports: Clearly state the course name and number, term, team number, and team members on the title page of your project report. The project report is to be formatted with single line spacing, Calibri or Times New Roman 11pt font, and normal margins (2.54cm all around). If you are using an application other than MSWord for your report, convert your report first to either a PDF file or a DOC(X) file.

Submission of Source Code: While your team is required to work on the deliverables in the repository assigned to your team in the GitHub organization of this course, ***all project deliverables must be submitted in myCourses***. Submit the source code of your application as a zip file. Use the Export feature of Eclipse to create an Archive File (Export – General – Archive File) of the project. If you realize that you need to make changes to the submission of one of your deliverables, do not resubmit only the file(s) that have changed, but rather resubmit another complete zip file.

Member Contributions: Each team member must contribute to each project deliverable. A team member who does not contribute to a project deliverable receives a mark of 0 (zero) for that deliverable. A team member may optionally email a confidential statement of work to the instructor ***before the due date*** of the project deliverable. A statement of work lists in point form how team members contributed to the project deliverables. In addition, the statement of work also describes whether the work load was distributed fairly evenly among the team members. A statement of work may be used to adjust the mark of a team member who is not contributing sufficiently to the project deliverable. It is not necessary to email a statement of work, if a team distributed the work for the project deliverable fairly evenly and each team member contributed sufficiently.