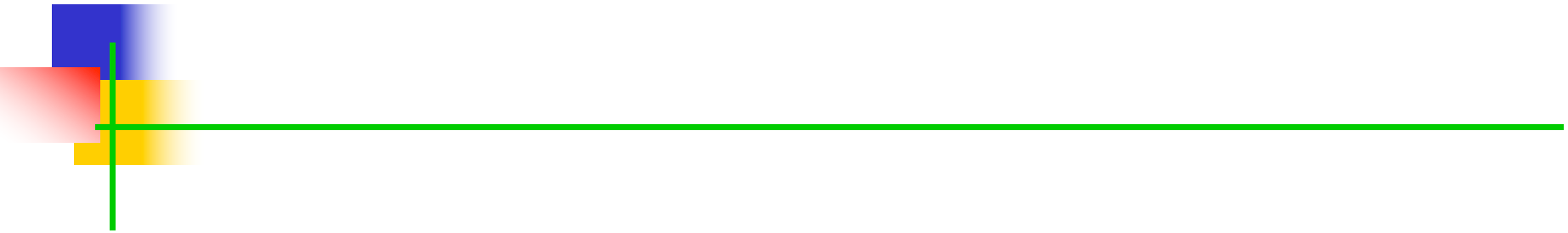
Assignment 2 – Team Project: A Distributed Game



Dr. Rajkumar Buyya and Dr. Egemen Tanin

School of Computing and Information Systems
The University of Melbourne, Australia

Distributed Scrabble

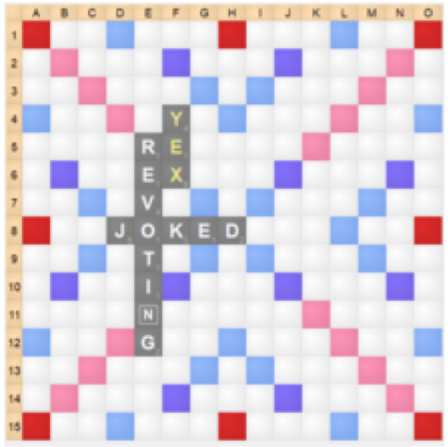
- You will implement a Java application that runs on a Distributed System for the second project.
- Team Size: 4 (Strongly recommended)
  - Note: Each member contribution will be evaluated and marked.
- General help: Ask tutors during/after tutorials in person. Also use the associated Discussion Board in LMS.
- Total Marks Allocated: 25

Scrabble Game Characteristics

- You need to create a 20x20 grid where users on different PCs, min of 2 people, but allowing more if there are more players, will place letters on tiles of this grid to make words in turns.
- We ask you to implement a specific version of this game as follows but you are welcome to read about Scrabble game online which is a popular game with numerous versions.
- Users will take turns to place a character in a tile of the fixed grid mentioned above.
- When tiles that touch each-other make a word, then the person who placed the associated letter will get points equal to the total length (number of tiles) of the word.
- A proper word is judged when all players accept the word through a visible voting GUI.
- The game ends when all users have a turn and cannot find a word/extension to make, basically when all players say pass when their turn comes through a GUI.
- Words are written in English and can only be read from left to right or top to bottom.
- The game should have appropriate GUI for allowing appropriate controls and information to play as well as to follow it by all players.
- All users should be able to see the same view of the game on their machines without differences or errors between them. Start and end game should be clear.
- Users should be able to logout from the game any time which would lead to end game as well.
- There should be a game membership pool where users come in when they login to the game application and see other potential players.
- You are welcome to have simplifying assumptions for membership and game start management such as having at most one game at a time and not allowing people to join games at any time but only at the beginning of a game.
- You need to let any player initiate a game and invite others who can then accept this.
- You can get creative and implement a more complex game if you agree as a team.

Characteristics Contd.

- An example game GUI is given below, from Wikipedia. Please have a read on what Scrabble is if you have no idea about this game and its versions. You only need to implement our version described here.
- The system must be implemented in Java but you can choose the technology you want to use to build your distributed application:
  - Sockets
    - TCP or UDP?
    - Message format? (JSON?)
    - Exchange protocol?
      - E.g., server broadcasts a message with updates to all other clients, other clients reply acknowledging the message.
      - Peer to peer?
  - Java RMI?



Challenges

- Dealing with concurrency
  - Regardless of the technology you use, you will have to ensure that access to shared resources is properly handled and that simultaneous actions lead to a reasonable state. The game logistics leads to some application level order which you need to implement so that the game board is altered by one person at a time for example. There are other cases where you need to do more to solve conflicts.
- Structuring your application and handling the system state
  - For example you can have one or more servers that communicate with each other, use a P2P or client server approach.
- Dealing with networked communication
  - You have to decide when/what messages are sent across the network.
  - You may have to design an exchange protocol that establishes which messages are sent in which situation and the replies that they should generate.
  - If you use RMI, then you have to design your remote interface(s) and servants for example.
- Implementing the GUI.
  - You can use any Java library for this implementation but no outside packages that implements the core features of the Distributed Application or the Game Interface.

Some Further Requirements

- Users must provide a username when joining the game. There should be a way of uniquely identifying users, either by enforcing unique usernames or automatically generating a unique identifier and associating it with each username.
- All the users should see the same image of the game and should have the same player capabilities and no difference between player capabilities.
- When displaying a board, the user interface should show the usernames of other users who are currently in the system and their scores etc. You cannot assume implicit knowledge of other players or that they are sitting next to you.
- Users may connect and disconnect at any time and the game should not crash but stop gracefully such as announce a winner of the game at that state.

Further Requirements Contd

- Users should be able to play and view the game state in real time, without delays. A change on one player should be visible on others immediately from a user point of view.
- There is no need to authenticate users that want to access the system accept a simple system login page to get names etc and then listing online members at that time.
- Important: you are NOT allowed to use ANY code taken from an existing shared game implementation.

Simplification

- In real life boot strapping such a game would need elaborate mechanisms such as finding other users and downloading the application etc.
- In our setup, you are welcome to explicitly enter IP addresses, port info etc in a simple game control interface to declare the location of others in the game. Thus we do not ask you to implement a web service or a P2P mechanism to locate others over the internet.
- More elaborate schemes on this front are acceptable and encouraged.

Deliverables

- There are **two deadlines** for this assignment:
  - Deadline 1 – Milestone 1 (Progress Review): Teaching **Week 10, during the tutorials**
  - Deadline 2 – Milestone 2 (Final Submission): **Week 12, Sunday (Oct 14) at 11:59pm**
  - **Demo format/location to be announced.**

Deadline 1: Progress Review

- You will have a quick discussion with your tutor and explain the design of your system and show your code as much as you can.
- You will show your tutor a demo of the functionality you have already implemented:
  - You should have implemented at least a single-user playing the game
  - The rest of the game implementation is designed and work distributed among members of the team and you are ready to start coding the rest
  - This is the minimum stage you should be in at week 10, but we advise you to aim for more and basic communication between players could be working at this stage – this is encouraged by the following marking scheme.
- **5 marks** will be given at deadline 1 when we see your progress at this time as follows. All team members should attend the first demo.
  - 5 marks will be given to teams who are at the above expected min stage and at least have implemented something more than the min on the distributed front.
  - 4 marks will be given if your team meets min. 3 if a problem is noticed there.
  - 2 marks is given when multiple problems occur with min. 1 and 0 means, effort only and no effort at all accordingly.

Deadline 2: Final Submission

- Report
  - You should write a report that includes an intro about the game and the system architecture, communication protocols and message formats, design diagrams (class and interaction), other implementation details, and a section that outlines the contribution of each member.
- You need to submit the following via LMS:
  - Your report in PDF format *only*.
  - The *executable jar* files used to run your system's clients/server(s).
  - Your source files in a .ZIP or .TAR archive *only*.

Deadline 2: Final Submission Contd

- Demonstrations
  - You will showcase your system and discuss your design choices during the demos (as done for assignment 1).
  - Dates and venues will be announced closer to the submission date.
  - You will be required to bring your own laptops and a printed copy of the report.

  - The final submission accounts for 20 marks. Marking guide to be announced soon.

Penalties for late submissions of assignments

- Assignments submitted late will be penalized in the following way:
  - 1 day late: -3 marks
  - 2 days late: -6 marks
  - 3 days late: -9 marks
  - 4 days late: -12 marks
  - 5 days late: -15 marks
  - No submissions after the 5^{th} day is possible.