Distributed Systems Spring 2013

CTF - Architecture 2.0

For Example

(Formally The Engineers & Floppy Disk)

MEMBERS:

Name	Email
Gurwinder Singh	gsinghny@bu.edu
Alejandro Pelaez Lechuga	apelaez@bu.edu
Amelia Martinez	mely91@bu.edu
Yan Olshevskyy	yanolsh@bu.edu
John Martin	jcmartin91@bu.edu

Game Logic

Game Specs

Game

- **When does a game starts ?**
    - *Need a min of 2 player to start game.*
    - *Next player added to the 'other' team.*
- **When should we allow players to join a game?**
    - *Connect auto joins the current game*
    - *They can join any game, if the game is at max cap do a connection refuse*
- **What happen to the players when the game is over?**
    - *The game resets, which means that all the players from the old game are auto joined into the new game.*

Players

- **All players can occupy the same location**
    - *Issues: Displaying the player info for many players on the same cell*
        - *Fix: Your player is highlighted to be green*
    - *Issues: Which player picks up item first if multiple players on same item cell*
        - *Fix: First person to do the RPC call gets it; all others will get an error message*
- **Players cannot move out of the board bounds**
- **Maximum number of players <= min (total number of home cells, total number of jail cells)**

Teams

- **There are two teams: Team 1 (RED) & Team 2 (BLUE)**
    - *First player is assigned to RED first, then the next player is assigned to BLUE, and so on.*
- **New players are assigned to the team with the least players. If both teams have the same amount of players, the new player is assigned to Team 1.**

Cells

- **There are 7 types of cells**

    - *“ “ - floor cell*
    - *“#” - wall cell*
    - *“h”, “H” - home cell*
    - *“j”, “J” - jail cell*
    - *“f”, “F” - flag*
    - *“p”, “P” - Player*
    - *“s” - shovel*
- **Adjacent cells are cells above, below, left, and right.**
- **Diagonal cells are not adjacent**

Home Location

- **Each team has a home location**
- **Spawning location of players**
- **Players members cannot enter their opponent's team home**

Tagging

- **2 sides RED and BLUE, split evenly based on the map dimensions and home location**
- **You can only tag the other team on your side of the map.**
    - *X can only tag Y players on the X side of the map*
- **Tagging sends the player to the jail on the side they were tagged on.**
- **A player gets tagged when he is in the opposite team’s side of the map and he is in the same cell as an opposite team’s member**

Jail

- **Player goes to jail if it gets tagged by an enemy team’s player.**
- **Tagging happens automatically when the condition mentioned above occurs; the tagging player does not need to execute a “tag” command**
- **A tagged player is automatically sent to the opposite team’s jail**
- **Players in jail cannot move away from the jail cells**
- **Players are freed from jail when an ally player (a player from the same team) that is not in jail moves to a cell adjacent to an opposite team’s jail cell**
- **When players are freed from jail, they are moved to their team home**
- **The player that “frees” is not moved back to his team home**

Items

- **A player can only hold one item**
- **The flag IS an item**
- **When a player holding an item is tagged, he drops the item.**
- **When an item is dropped, it returns to a random location on the side of the map where such item belongs to, and the player is moved to jail**
    - *The random return location cannot be occupied by a player, item, home cell, jail cell, or wall cell*
- **There are only two items (so far)**
    - *Flag*
    - *Shovel*

Flag

- **Each team has a flag**
- **"f" is Team 1's**
- **"F" is Team 2's**
- **The flags start at random locations within their corresponding teams side of the map**
- **Players cannot pick up their team's flag**
- **Players cannot move to the cell where their team's flag is on (for displaying purposes)**
- **A player cannot voluntarily drop the flag**
- **Flags are always visible (Game version 1.0)**

Shovel

- **A player holding a shovel can walk through walls**
- **A player holding a shovel can break "breakable" wall cells; the wall cell becomes a floor cell**
    - *Note: some notes might not be breakable.*
- **When a player holding a shovel breaks a wall by moving into it**
- **When a player holding a shovel pickups a flag, he drops the shovel**
- **Shovels are always visible (Game version 1.0)**

Game Ending Conditions

- **Tie**
    1. *When all players disconnect without meeting the winning condition*
- **Winning**
    1. *Have the enemy's team flag on your side of the field and no players on your team in the enemy team's jail and on your side of the field.*
    2. *All players on the other team disconnect, as long as the game has started.*

Structs

Server

Global Game Struct

- array: Player* struct (array of all the players)
- "boolean" Global gameStart
- int Game Version (gets updated at game start of a game)
- int Game State Version
- array: Item* (array of all the items "shovels" and "flags")
- array: int [team1, team2] (number of players on each team)

Cell struc

- Enum cell Type
    - " " free cell so movable
    - "#" wall cell so immovable
    - "h", "H" - spawn point for team
    - "j", "J" - jail cell
    - "p", "P" - Player
    - "i" - items ("i" is a placeholder, the actual character to be used is specified in the item struct)

- Position struct (x,y)
- Breakable? [Y(es) or N(o)]
- Player*
- Item*

Player Struc

- int ID ()
- int Team (whose team the player belongs to)
- Position struct (x,y)
- State (Jailed, Tagged, Free)
- *Item

Item Struc

- enum type
    - “f” - (Team 1 flag)
    - “F’’ - (Tram 2 flag)
    - ‘s’ - (shovel. All players can pick it up)

Maze Struc

- int numfloor
- int numJailCells(x2)
- int numwall
- int dimensions (int X, int Y)
- array of cells (the actual map)
- *Update with code. Code version is probably more complete*

Position Struc

- int x
- int y

Client

To fully specify all the client structs we might need to explore how to display the map

GameData

- Player id: long int
- Game State Version: unsigned long int
- Game Version: unsigned long int

Maze

- Maze rows: int (maze y dimension)

- Maze columns: int (maze x dimension)
- Maze: Char*

Communication

RPCs

Connect RPC

- Joins game
    - Error cases - can not join due to max capacity
- Returns player id number or -1 in case of error

Move RPC

- Up, Down, Left, Right
    - Returns valid or invalid
- Parameters sent:
    - ‘U’ - move up
    - ‘D’ - move down
    - ‘L’ - move left
    - ‘R’ - move right
    - int ID. player id
- Note on move: Move. When a players move, both the source and destination cell are locked

Disconnect RPC

- Disconnects the player from the game.
    - Removes the player from correct structs.

Event Channel Message Format

Delta of the map changes (Need to define a good algorithm for this)

[OBJECT][PLAYER][MAP] - FULL STATE
Should only sent items that change into body, deltas sent via header encoding.

Event Channel Marshalling (Format)