Instructions for SWEPMUD

Michael Barth, Hubert Baumeister, Florian Hacklinger,
Piotr Kosiuczenko, Axel Rauschmayer

März 2003, Version 0.3

Contents

1 Business case

The company P2 is a telecom company that plans to increase the market share of network users in the age range from 15 to 40 years. To attract people in this age range to move to P2, a multi user dungeon game (SWEPMUD) playable by mobile phone should be implemented.

The communication costs for people playing the game should be kept low to attract people to join P2. The revenue will come from the use of standard services (phone calls, SMS, WAP/Internet, etc.). Further, the game should be easily scalable to several thousands of players at the same time (possibly using different instantiations of the game).

In particular this means that

1. most activities performed during the game should be handled locally on the mobile phones of the players, code may move from one mobile phone to another to achieve this;
2. most communication should happen between the players of the game themselves without involving a central server (peer to peer architecture).

2 Overview

The language used in this course is English.

SWEPMUD is a multi user dungeon (MUD) game, that is played with mobile phones. The player registers once for this game and can then use his cell phone to play SWEPMUD .

2.1 Sketch of the game structure

The "playing field" is partitioned into levels. Each level consists of several rooms. The rooms are connected by doors that allow a player to move from one room to another. Inside a room, a player can interact with other players and take, use or put down objects. To improve his abilities, a player interacts with other players or objects (see 3.1).

Every level contains two distinguished rooms. Namely a start room and a special (or final) room. A player enters a level through the start room. He can proceed to a higher level if he successfully performs a task in the special room.

2.2 The Goal

The goal of the game is to sequentially traverse all levels and to successfully complete the task of the final level's special room.

3 Detailed Instructions

3.1 The Player

Every player moves through the world of SWEPMUD . He has a name that is being chosen while registering. A player's condition is defined by the following attributes:

- live points,
- strength,
- agility,
- money.

The definition of a player's condition is performed by creating a profile. Doing so means to distribute a given amount of points between the attributes strength and agility. The player starts the game with a certain percentage of these maximum values. By going through experiences like fighting or progressing to a higher level a player usually improves his condition, but the condition never improves beyond the initially defined values. Progressing to a higher level increments the player's life points.

If the player's life points are zero he is dead. He has to start over the current level in the start room. He loses his objects, which remain in the room where he died.

3.2 Other Game Participants

The game is also populated by other players, part of them are human controlled (HC), part of them are machine controlled (MC) or automatic. A player can interact with HC players freely, MC players only allow very constrained interaction. MC players are either "good" or "evil". (Tentative names are *Data* and *Lore*.)

Evil MC players attack other players, good MC players never attack. Some good MC players repair broken objects. They might do that for free or charge for it. They are also very strong to prevent being attacked by HC players. MC players always stay in one room.

3.3 Interactions between Players

3.3.1 Talking

A player can talk to any other player in the same room. The conversation is free-form and can be terminated at any point.

An arbitrary number of players can participate in a given conversation. Non-participants cannot hear what is being said.

The communication with MC players is limited to picking from a predefined set of statements.

!!! DER FOLGENDE ABSATZ WIEDERHOLT FAST GENAU DAS, WAS OBEN STEHT!!! Free-Form Talking: A player can at any time of the game talk to another player in the same room. The player sends one or more sentences with the "talk-to" command to the other player. While talking, all other activities in the game are possible.

3.3.2 Exchange and Trade

Players can trade any object they own. Objects or money are used for bargaining. Negotiating the trade is done by the standard talking.

The actual trade is performed by an atomic "take-give sequence".

3.3.3 Fights and Robberies

Sequence of a Fight

Challenge to Fight: A player sends a "challenge to fight" to another player (who is in the same room) to initialize a fight against that player. The opponent (defender) has to confirm that call. He cannot carry out other actions in the game until he confirmed the challenge to fight.

Structure of an Attack: After the defender confirmed the attack, the attacker composes his attack. This is done by choosing two objects he wants to use in this attack (they must be in the attackers possession, of course).

This pair is transmitted to the opponent. E. g., he chooses to first strike with a sword and then with a knife.

Structure of a Defense: The opponent is allowed to see the *first* of the two offensive actions. Based on this information, the opponent composes two defensive actions by choosing which objects he wants to use as defense objects.

Common Poperties of Attacks and Defenses: Offensive and defensive actions consist of using objects (weapons). There is only one way of using an object. The two offensive actions need not necessarily use the same object. For instance, one can attack by first using a hammer and then a cudgel. Every object can be used offensively as well as defensively. Of course, some objects are more appropriate for an attack, others are better suited for defending.

Evaluation: The two pairs of attack and defense actions allow one to compute the result of that fight. Every action of attack can be countered by an action of defense. The evalution has the results "blocked" and "hit" .

The evalution is done in the following order:

1. The first action of attack is compared to the first action of defense.
2. Next, the first action of defense is compared to the second action of attack to find out if the defensive action had a negative impact on the attack. If so, the attacker has been hit.
3. Finally, the second action of attack is compared to the second action of defense.

As soon as an evaluation step leads to a hit, all further evaluation is stopped.

Consequences of a Hit: If a player is hit, he loses life points. The amount of the loss depends on the opponent's object used to accomplish the hit. The state of the used objects is being adapted. They lose status points.

Possible Actions for the Defendant: After one round of the fight, it's the defendant's turn to act: Before he can be attacked again, he must have the chance execute at least (any) one action. This action could be to attack the attacker, to attack somebody else, to leave the current room (that is, to flee), interact with other players, take an object, etc.

Death of a Player If a players dies, i. e., the amount of his life points is 0, he is transported to the start room of the current level. The dead players objects remain in the room of his death.

Weapons: In the game there are different weapons (or fighting objects). Every weapon is characterized by three attributes:

- The amount of strength points that are necessary to use the weapon.
- The amount of agility points that are necessary to use the weapon.
- The amount of life points the opponent loses in case of a successful attack.

Every fighting object can be used as a weapon for an attack, as well as a weapon for defense.

Table of Evaluation (Example)

A → D / D → A	O_1	O_2	O_3	O_4
O_1	x / x	x / 0	0 / 0	0 / x
O_2	x / 0	0 / x	x / x	0 / 0
O_3	0 / x	0 / 0	0 / x	x / 0
O_4	x / 0	0 / x	x / 0	x / x

The table describes the ways how two objects interact with each other. There are four objects defined.

At the evaluation there are two different situations: First, the evaluation if an attack can be warded off with a defense action (A → D), second the evaluation if a defense action has a negative effect on the subsequent attack action (D → A).

To evaluate the first situation look into the row where the object used to attack is listed, and then go to the column where to object for defense is listed. For the second situation look first for the object used for defense in the first row, then go to column for the object of the subsequent attack.

Table cells with an X denote a successfull action (i. e., the attacker hit the defender or the defender hit the attacker), table cells with a 0 denote a parade (i. e., the defender wards off the attack or the defense action has no negative effect on the following attack action).

Evaluation Examples At an attack with the actions (O_1, O_4) with the defense (O_3, O_3) the defender will be hit: (O_1 can be stopped with O_3, the defense O_3 has no negative effect for the second attack O_4, the attack O_4 can not be ward off with the defense O_3).

The attack (O_4, O_3) with the defense (O_2, O_1) leads to a hit against the attacker: O_4 can be stopped with O_2; O_2 has an negative effect for the second action of attack O_3).

The attack (O_3, O_2) vs. (O_1, O_2) ends tied (no player is being hit), all actions can be ward off resp. have no negative effect.

3.4 Objects

Objects can be found in rooms. They are used by players for certain purposes (e. g., healing potions give life points, weapons improve the chances of winning a fight, shields or armors reduce the impact of hits). Certain object demand minimum attribute values from player who wants to use them.

Objects have a state. The state indicates the condition of an object. The state of an object changes with use. Example: A sword loses state points while being used in battle. If the state value is 0 an object is broken, i. e., it can‘t be used anymore. A player can keep broken objects, and have them repaired by certain good MC players.

Examples of objects are: Sword, knife, shield, armor, healing potion, etc. Objects in the game should be variable and configurable.

3.5 Rooms

Rooms have doors, that connects them to other rooms. If a player enters a room, a description of that room is displayed. The structure of a level's rooms is an undirected graph. The graph's nodes are this rooms, the edges are the doors. Every level contains two distinguished rooms: the start room and the special room. The player enters a level in the start room. He can try to proceed to the next level in the special room. Once a player left the start room he can't go back. For details about the special (final) room see 3.6. When a player enters a room all other players inside of the room are notified of this event.

A player can perform a set of actions inside a room:

- Look: receive a detailed description of an object.
- Use objects (e. g., drink a healing potion).
- Interact with other players (talk, trade, fight, exchange objects)
- Enter a neighboring room.

3.6 Progressing to the next level

If a player wants to proceed to the next level, he must move to the special room. At a given time there can only be one player in this room. If other players want to enter special room that is already occupied, they get their own copy of the room.

Before a player can proceed he has perform a task, that consists of facing an evil MC player. The evil MC player tries to prevent the player from going to the next level.

If HC player defeats the evil player, he is transported to the next level's start room. His life points are then restored. If he fails, i. e., dies, he starts over in the start room of the current level. The door of the special room is marked to prevent players from accidently entering the room.

3.7 Refresh

Every instance of the game can be populated by a certain amount of players. As long as the current player count is less than a threshold value, new players can enter the game. The player count remains unchanged when a player goes offline.

To keep the game attractive for newly joining players, objects and MC players are periodically being "refreshed". (**Diskussion:** Ebenenabhängig?)

4 Description of the Device

The game that is to be developed during the SWEP runs on client-side on a mobile phone. Thus, the user interface is limited to the usual facilities of a typical mobile phone.

To be concrete:

Display in text mode In the text mode the mobile displays seven rows and 16 columns.

Display in graphic mode In the graphic mode pictures with a resolution of 101 by 80 pixels can be displayed. The number of colors is unlimited.

Keys The device has the following keys:

- A regular numeric block with the keys '1'-'9', '*', and '#'.
- A 5-way-navigation key. It allows the user to go left, right, up, down or to select (press center).
- One key to pick up and one key to hang up.
- Two keys directly underneath the display for context sensitive functions.

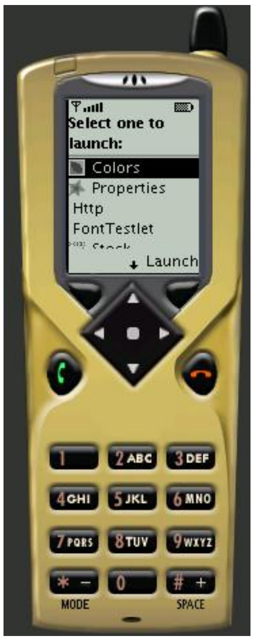

Figure 1: A mobile phone that corresponds to the specification in the text