IT 2931 Game Development Techniques

Technical Document

Game Title:	Dungeon Revival		
Company name:	Mikonos		
Company Website:	http://www.geocities.com/hkzr/Mikonos		
Team Member:	Hansel Koh	050664U	(Project leader)
	Chen Ren Hao	052663Z	(Audio Person + Marketing)
	Nigel Low Wei Yang	052365B	(Lead programmer + Tester)
	Melissa Lim	054131H	(Designer + Artist)

Version # 2.00
Thursday, August 10, 2006

Content

Content

2. Task allocation list

3. **Schedule**

Game functionalities

Path-finding

- **Improvement**

We will be implementing the AI path finding to avoid walls in the future, as well as altering it to change with time so that faster computers will not make the monsters too aggressive.

Inputs: target position, invoker's position

Function: Unplanned based movement of the monsters in the game.

Returns: Updated position consistent with man-time

Process: Target position is checked if visible, if not visible another sister function will be called to help move the invoker around the obstruction before continuing straight to target position.

Improvement: Currently, the AI for our monster is able to detect the player's direction and detect if the player is visible through the confines of the maze.

Advanced Collision Detection

- **Improvement**

We are improving the collision detection to reduce the bug in the game to the minimum. Currently the player is bounced off the walls

Inputs: Player's position, 3D-array storing game level's data

Function: Allows player to collect objects, trigger events with monster and bounce off the walls upon contact.

Returns: Updated flags to allow events/situations to occur.

Process: Position of player is checked every frame. If one of the flags returns true, the desired event/situation is activated.

Improvement: Currently the bouncing effect pushes the player back in the direction of his key inputs, which can be abused to pass through walls, as the bouncing process do not check if it is colliding with walls.

Level timer in level

- **Enhancement**

We decided to implement a level time for every level of the to create a feel of rush to collect all the coins as fast as possible, by making quick moves around the game maze and avoiding monsters.

Inputs: Game level (maximum time allowed)

Function: Limits the time allowed to be spent on each level. If the time is up, game over. Player must complete the level before time-up.

Returns: Updated flags to allow events/situations to occur.

Process: Amount of time passed since start of level is checked with the maximum time allowed.

Enhancement: Currently there is no timer in the game. Thus the player can collect the coins in the level with ease and it can get boring. Timer is introduced to ensure player will want to move around the maze with haste and dexterity. This will also make the player lose if he is unable to find the last few stray coins he had missed out carelessly during game-play.

Narrower passage way of bigger monster (Redesign of map)

- **Improvement**

We will redesign the map to balance the game making it slightly more challenging for the gamer to get the coin by putting the monster to guard the coin in certain area of the maze. Also, our map will be small in the first level, and slowly expand as the levels increase. This will increase the difficulty in the game and reduce the chances of character running pass the monster with out any damage. This is a design issue that sprouts from technicalities of the game.

Inputs: Text file containing game level data

Function:

Returns:

Process:

Improvement: Currently the map is very spacious and very disorientated. As game level design is the essence of Gameplay, we will try to minimize unnecessary wastage and spaces as well as ensuring the game's progression remains at a good and stable pace.

Game functionalities

Improvement of gem's functionally

- **Improvement**

Improving of gems function so as make the gem more useful to the player rather then just killing the monster. The new function that we will be adding will be when gem is collected then the player can see where the rest of the coins are and implementing rules like red gems can only kill red enemies etc.

Inputs: Flags from collision detection, and parameters for setting the functionalities.

Function: Gems are the extra depth to the game, a form of aid so that the player can get a temporal advantage over the game.

Returns: More flags to activate other events/situations.

Process: Varies depending on rules and function. Flags will activate the gem's effect on the player, and parameters will notify the type of behavior desired by the Gameplay.

Improvement: The gems are now for a single function only, which is to kill the monster in one hit. To make the game more in-depth we are now looking into implementing new features and rules for the gems.

Power-ups

- **Enhancement**

We will be implementing power-ups for the character. Power-ups can be in the form of increase the movement speed, health and etc. to boast the player's morale and increase depth of the game.

Inputs: Flags from collision detection, and parameters for setting the functionalities.

Function: Power-ups are the extra depth to the game. Unlike gems, power-ups are a direct form of temporal aid to help ease the game.

Returns: Updated flags to allow events/situations to occur.

Process: Varies depending on rules and function. Flags will activate the gem's effect on the player, and parameters will notify the type of behavior desired by the Gameplay.

Enhancement: With power-ups, the character can last longer in this unforgiving maze. Right now there is no such things in the game, which means the Gameplay is quite linear and expected, making the game shallow.

Game functionalities

Lighting

- **Implementation**

We will use candles in the game. This will be the source of dynamic lighting and adds to the dungeon feel of the game. Light will emit from these candles, which will be just an excuse for the presence of lights. Models will be made in outside software.

Inputs:	Model from loader, positioning in game level from map data, color and intensity of light source.
Function:	To light up the game dynamically and solve certain lighting bugs issue at game initialization.
Returns:	
Process:	Places the candles' model and in-game real-time lighting.
Implementation:	Lightings are for the mood of the game. Staying true to the dungeon universe, we will use candles to illustrate the presence of lights in some corners and at the same time implementing dynamic lighting and shadows to add to the depth of the game.

Different Soldiers

- **Enhancement**

We are intending to add Elemental Patrols. Basically, these are Patrols that cannot be hurt by normal gems. Players must collect special colored gems to defeat these guys. Red Patrols must be defeated using Red Gems, etc.

Implementing different monster using different colors to represent its ability of the monster such as its area of sight and intelligence of it path finding.

We will also increase the difficulty by adding in 'mixed' soldiers, which combine the 2 above-mentioned changes. Not only that, their element will also change with time, so players are tested on their speed and decision making.

Inputs:	Flags from collision detection and path finding, and parameters for setting the functionalities, player's position, monster's position, effective range of the monster and lastly, the model.
Function:	Basic monster class is done, now is an extension to the monster handler to include more depth to the game by including more functions and features into the monsters.
Returns:	Updated flags to allow events/situations to occur.

Game functionalities

Process: Varies depending on rules and function. Flags will activate the effects by the player, and parameters will notify the type of behavior desired by the Gameplay, events will be tested against the rules and the result is returned in the form of updated flags to change the situation of the game.

Enhancement: Right now the monster is very basic, it will chase the player if in range and in sight, upon losing sight of the player it returns to original position. If the gem's effect on the player is active it will not chase. So now we will make it more challenging to kill the monster by making the monster change element with time, and the corresponding type of gem is needed to kill it at the correct time.

Animations

- Implementation

We intend to animate the movement of the Hero, and the Patrols. We intend to do so by loading our models by parts and animating each of them, or animate them using Maya or other programs, and them load it using MD2 format.

Inputs: Model from loader.

Function: To make the game more lively.

Returns:

Process: Use the md2 format and animation through vertex animation.

Implementation: Animations are for the mood of the game. This is possibly a compound task to do and on the technical side, an MD2 loader for the game is needed.

Traps

- **Enhancement**

We would like to place traps here and there about the maze, to add the element of surprise and depth to the game. Traps would probably include random teleportation, so players might end up next to a coin, or a Patrol. It will also include spikes to reduce the health of the player, or blind the player, not allowing him/her to see the mini-map for a period of time.

Inputs: Model from loader.

Function: To make the game more lively.

Returns:

Process: Use the md2 format and animation through vertex animation.

Game functionalities

Implementation: Animations are for the mood of the game. This is possibly a compound task to do and on the technical side, an MD2 loader for the game is needed.

More levels and fixed story line

- Implementation

An official frames our hero in the kingdom, and he is thrown into the deepest level of the dungeon to live out the rest of his days. The guard who was guarding over him agreed to let him escape, provided he paid an agreed price. However, this lift can only go up to the next level. So in order to get out, our Hero must bribe all the guards in the levels.

Inputs: Flags from the game to trigger events, game level data and game states

Function: To give a conclusion to each level.

Returns: Event flags, and new game level.

Process: When activated by the flag, which is when the player has completed the game level, a lift door opened by the guard allows the player to move on to the next level.

Implementation: This will give a dramatic and conclusive transition between the game levels every time a level is completed successfully.

Create installation file (Optional)

- Implementation

To make the game easier to setup and packages the game up in a more professional manner.

Loading game and scoring system

- Implementation

When the player presses quit, we will show him/her the password to that level. When the player presses the load game button in the menu, he/she will be required to enter the given password. If a level is unlocked, the password to that level will be shown in the load game menu.

Inputs: Game level, game state

Function: To allow users to start the game from their progress.

Game functionalities

Returns: A password so that the player can access his/her progress at a later time.

Process: Checks the game state, if player quits properly, returns a password, which can be used to load the game level at a later date from the 'load' button.

Implementation: This is to implement the functions for the 'load' button in the game's main menu.

Portal

- **Enhancement**

Adding portal so that user can move to certain part of the maze, with this function in the game, it is able to create a bigger maps that have different floors.

Inputs: Flags from the game to trigger events, positioning in game level.

Function: To allow player to transition between different levels.

Returns: Other game map.

Process: When activated by the flag, which is when the player has entered the portal, the player is moved to the target area of the map.

Enhancement: This will better separate the game levels to make it easier to give each section of the map its own characteristics which in turn gives depth to the game.

Loading screen (Optional)

- **Enhancement**

Display a screen while the game is loading.

Frustum culling (Optional)

- Implementation

The current version of the game is running on graphic accelerated graphics cards like nivida of ATI. The game is also able to perform on integrated graphic card, but we would like to implement frustum culling so to get a even better performance on integrated cards.

Task allocation list

Project leader: Hansel Koh

More levels and fixed story line (3 days)

Create installation file (2 days)

Power-Ups (2 days)

Loading game and scoring system (4 days)

Audio Person + Marketing: Chen Ren Hao

Traps (4 days)
Improve gem's functionality (2 days)

Different Soldiers (5 days)

Level timer in level (1 day)

Lead programmer + Tester: Nigel Low Wei Yang

Path Finding (5 days)

Collision detection (3 days)

Lighting (3 days)

Design of map (1 day)

Designer + Artist: Melissa Lim

Animations (5 days)
More levels and fixed story line (3 days)

Graphics (4 days)

Schedule

Milestones

1.) Clean up game framework:
- Advance collision detection.
- Game level timer.
- Various stories screens.
- Lighting.
- Loading and saving game.
- Loading screen.

2.) Improvement visual/design:
- Animate character movement.
- Animate monster movement.
- Gem functionality.
- Game level timer animation.
- Enhance graphics and model.

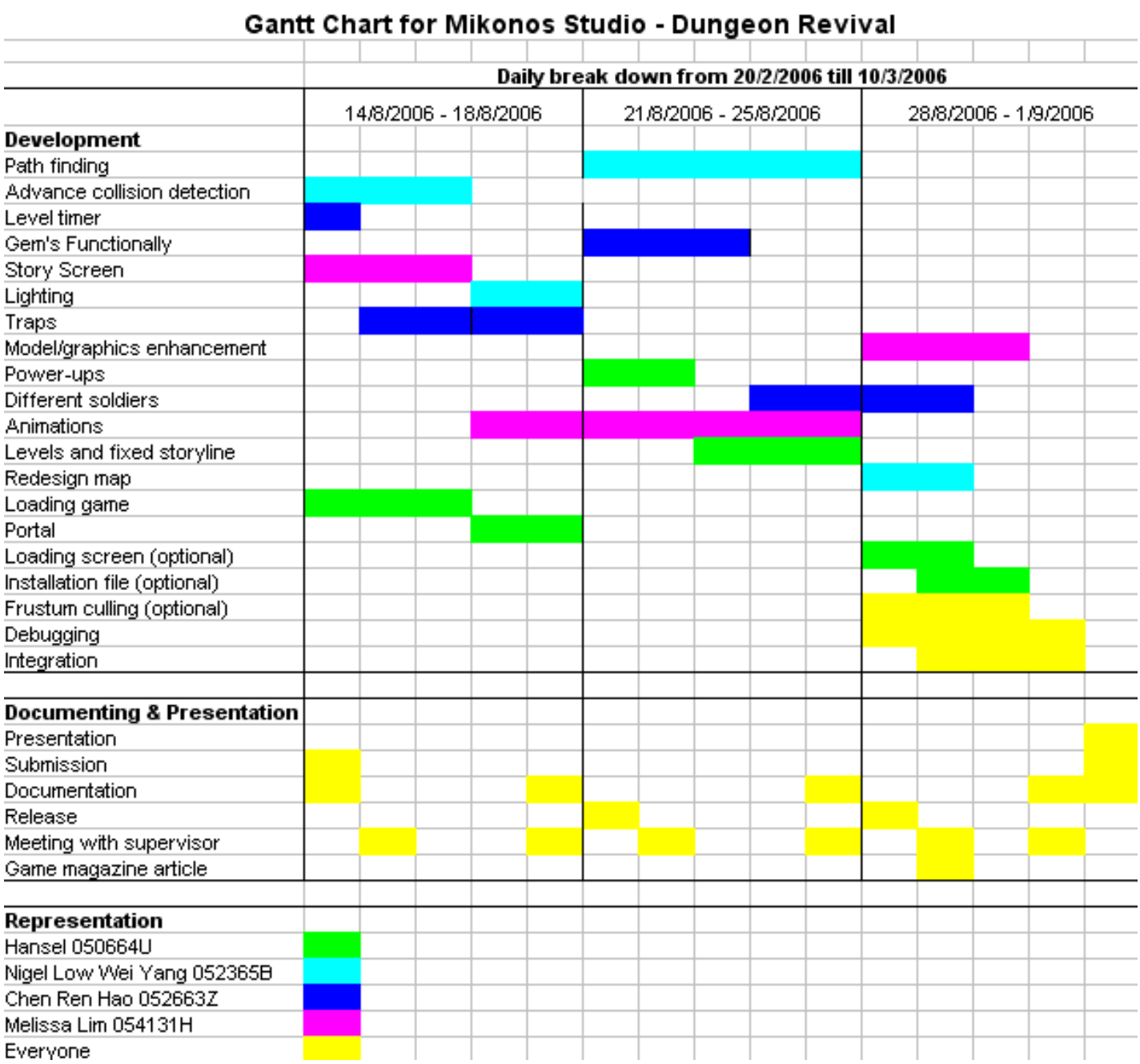
3.) Alpha version release, complete all game features (Sounds and special effects), A.I. path finding and create multiple monsters in game level.

4.) Game expansion:
- Adding power ups
- Adding traps.
- Adding monsters.
- Adding 2 more levels and storyline.
- Map design.
- Path finding.

5.) Debugging and clean up:
- Integration.
- Clean up bugs
- Debugging.
- Optimizations.
- Installer file.
- Submit game magazine article

6.) Beta version release, complete all game features (Sounds and special effects), A.I. path-finding and create multiple monsters in game level.

Gantt Chart


Gantt Chart for Mikonos Studio - Dungeon Revival
Daily break down from 20/2/2006 till 10/3/2006
14/8/2006 - 18/8/2006
21/8/2006 - 25/8/2006
28/8/2006 - 1/9/2006
Development
Path finding
Advance collision detection
Level timer
Gem's Functionally
Story Screen
Lighting
Traps
Model/graphics enhancement
Power-ups
Different soldiers
Animations
Levels and fixed storyline
Redesign map
Loading game
Portal
Loading screen (optional)
Installation file (optional)
Frustum culling (optional)
Debugging
Integration
Documenting & Presentation
Presentation
Submission
Documentation
Release
Meeting with supervisor
Game magazine article
Representation
Hansel 050664U
Nigel Low Wei Yang 052365B
Chen Ren Hao 052663Z
Melissa Lim 054131H
Everyone