Project Number: RL1-P208

Dreamspace

Design and Development of a First-Person, Story-Driven Puzzle Game

Interactive Media and Game Development

A Major Qualifying Project Report

Submitted to the faculty of

WORCESTER POLYTECHNIC INSTITUTE

by

Albert Lo and Joshua Walkowski

Advised by

Professor Robert W. Lindeman

April 30, 2009

Abstract

for the

Design and Development of a First-Person, Story-Driven Puzzle Game

by

Albert Lo and Joshua Walkowski

This is an Interactive Media and Game Development Major Qualifying Project. This project focused on the design, implementation and release of a game modification using Valve Software's Source engine. The game, *Dreamspace*, is a story driven single player game filled with obstacle-based puzzles with emphasis on a disconcerting atmosphere that is unsettling to the player.

Dreamspace is a single player adventure game set in a derelict space freighter, the *MSC Jacob*. The player must journey to the bridge of the ship and discover why he is on the *Jacob* and how it got into its current state.

This document discusses the design of *Dreamspace* and covers its gameplay, story and puzzle design, as well as its artistic design and technical architecture. It also documents the process of implementing the designs and the testing and evaluation used to get feedback on the game.

Table of Contents

List of Figures

List of Tables

1 Introduction

Dreamspace is a single player survival horror/adventure game set in a derelict space freighter, the *MSC Jacob*. The player must journey to the bridge of the ship and discover why he is on the *Jacob* and how it got into its current state.

This document discusses the design of *Dreamspace* and covers its gameplay, story and puzzle design, as well as its artistic design and technical architecture. It also documents the process of implementing the designs and the testing and evaluation used to get feedback on the game.

1.1. One Sentence description

Dreamspace is a story driven first person adventure game set in space made using the Source Engine.

1.2. One paragraph description

In the bowels of the space freighter *MSC Jacob* a man wakes up with no recollection of what has transpired on the ship since the accident that has put him in the infirmary. After searching the rest of the ship, he notices that something has changed about the once familiar hallways and decks he inhabited. He sets out to find the video logs which record all happenings on board. In his search for answers he experiences odd occurrences which reveal to him a shocking truth.

1.3. Gameplay Overview

Dreamspace is a first person adventure game, focusing on storyline and challenges that will impede the player's progress. The player is guided by a mysterious entity over an intercom whose voice is coming from all the speakers on the ship. Beginning in the infirmary, the game will include three different chapters which consist of different portions of the ship.

To tell the story of the protagonist, surreal "dream sequences" are used. During these sequences, the player will be engaged in much more combat than the rest of the game. The enemies will be some sort of horror amalgamation which will be related to his past. We are striving for visceral, tense feel for the combat so weapons will be everyday objects on the ship. Finally, during non combat phases of the game, there will be a proliferation of puzzles throughout the ship. These will mostly consist of logic puzzles.

1.4. Controls

The controls for the player are standard keyboard controls for an FPS. Movement is controlled by the WASD keys on the keyboard and moving the mouse allows the player to look in all directions. The detailed controls are shown in Table 1.

Table 1 – Input Controls

Input	**Behavior**
Move Mouse	Perspective follows mouse
Press Left Click	Use equipped item
Press/hold W	Move forward
Press/hold A	Strafe left
Press/hold S	Move backwards
Press/hold D	Strafe right
Press Space	Jump
Press E	Pick up item

1.5. Puzzle Design

Although there were several types of puzzles proposed for the original design of *Dreamspace*, obstacle-based puzzles are the only type of puzzles used throughout the game. Implementation of logic based puzzles was proposed and examples were created in design, but were ultimately halted due to time constraints.

Examples of implemented obstacle-based puzzles include the tutorial puzzle, which requires the player to find a wheel for the door to open it, giving the player access to the rest of the infirmary. Also, there is a puzzle in the living quarters puzzle, where the player is given a group of numbers throughout the area and must punch the numbers into a keypad in a particular sequence to gain access to a room. The room contains an item the player needs to continue the story.

1.6. Combat

Combat in *Dreamspace* is sparse and mainly concentrated in the dream sequences. There are only two weapons in the game, the pistol and the crowbar. Using the pistol throughout the game would take away from the feel and story in the game so the pistol is only available in the opening room with limited ammo. The pistol is destroyed when the player picks up the crowbar weapon in the infirmary. This makes the combat very visceral.

1.7. Health

The player‘s health is not displayed on the HUD to add to the realism of the game. Health also does not regenerate so the player must be careful when entering a situation where possible attack or injury could occur.

1.8. Saving

The game utilizes the save ability already built into the Source engine. The player can save at any point in the game, preserving the environment and objects around them. The location of the player is also saved so when the saved game is loaded, the player is in the same spot when the game was saved.

1.9. Objective

The player’s goal is to get to the bridge to find out what actually happened on the ship while he was incapacitated. To do this the player must traverse the infirmary, living quarters and maintenance area. The player must solve various obstacle-based puzzles in each area to gain access to the next area.

2 Artistic Design and Vision

The artistic theme of *Dreamspace* is very futuristic, as the game is set somewhere in the future where space travel is commonplace. A majority of the game will take place on the spaceship *MSC Jacob*. The overall appearance of the ship is worn from years of use and is even further worn down due to the fact it is a transport freighter. The style of the ship will be inspired by spaceship interior/exterior designs from the *Alien* movies, *Doom* and other space movies such as *Star Wars*. At other points in the game the player will slip into dream sequences where the setting will be shifted from the ship to other locations such as the protagonist's home.

2.1. Level Design

Since *Dreamspace* is a story driven game, all sections of the ship are created with linearity so that the player does not wander around the ship without a sense of what to accomplish to further the playing experience. The three main areas (infirmary, living quarters, and maintenance area) are located on the ship. The infirmary is populated with objects that a person would find in a hospital, but with a dilapidated look. The living quarters contains the crew's sleeping quarters, locker room and bathroom. The objects in these rooms (furniture, appliances, etc.) also have a worn down look and feel to fit the game's style. The overall worn down look of all the levels (except the bridge), and the size of the ship, gives the player the sense that the ship was once heavily populated, but is now deserted. The scope of the ship is shown in Figure 1.

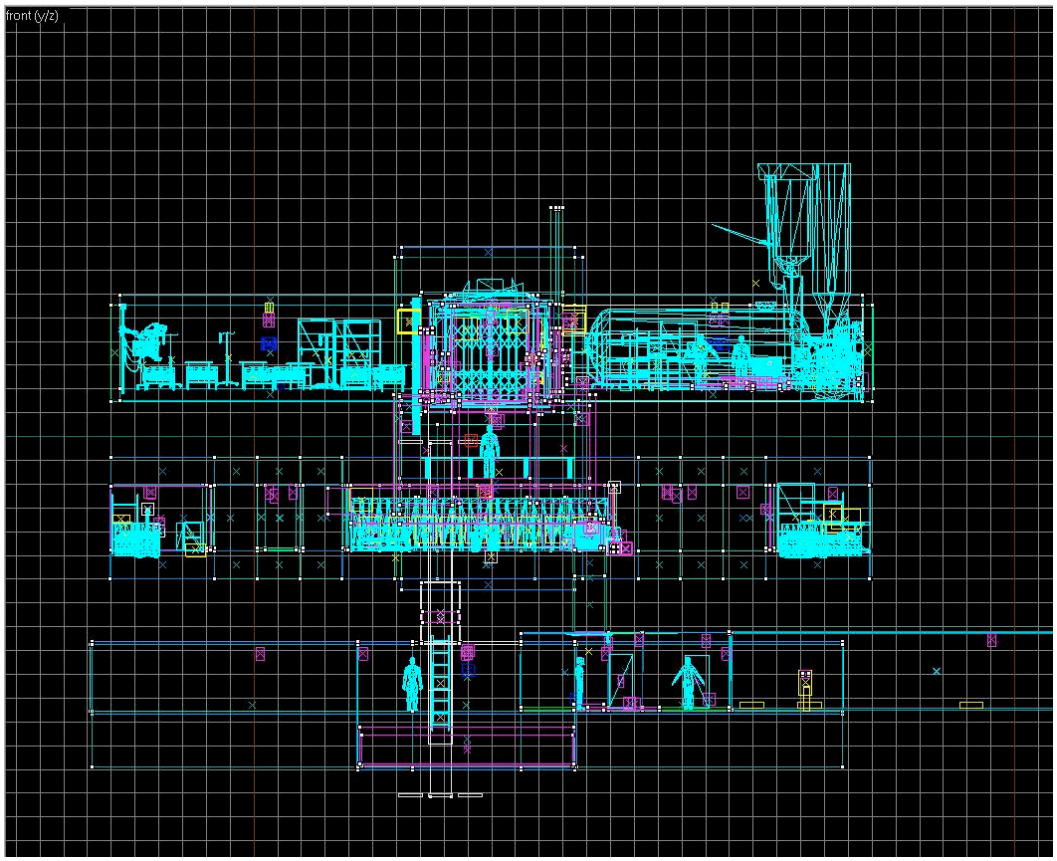

Figure 1: Map-Side View Hammer

The last area of the ship, the bridge, will be completely different from the rest of the ship. The pristine, bright textures used in the bridge differ from the rest of the ship to signify the end of the journey. All of the surfaces will be very shiny and polished in the light. The white effect gives the player a sense of surrealism, which is exemplified when the player realizes that he/she has been dead the entire time.

2.2. Dream Sequence Design

The dream sequence utilizes a different style from the rest of the ship to signify to the player that the dream sequence is not located on the ship but is rather generated by the player's memories. The style of the dream sequence is heavily surreal, using out of place textures and areas that would not be found on a ship. The original plan called for two dream sequences, but due to time constraints, the second dream sequence was dropped.

The dream sequence starts in a room similar to a modern day living room, as would be seen in an everyday house. A woman, revealed by the intercom to be the protagonist's wife, runs away

from the player, giving the player the sense that there are some unresolved issues with the protagonist's family. After the dream sequence, the player is instantly brought back to the ship in the maintenance area, which was previously unreachable. The startling end of the dream sequence is shown in Figure 2

Figure 2: Dream Sequence

2.3. Heads-Up Display

The HUD follows a minimalist design. There are no visual representations of the player's health or status. The visual display that appears when a weapon is picked up is also removed. The HUD was removed to give the game a cinematic effect, adding to the story.

2.4. Artistic Tools

Dreamspace's map was created using both the Source SDK's *Hammer* editor and *Autodesk's Maya* and *3D Studio Max* (3DSMax). Hammer was used to create the large portions of the level

such as the walls, floors and ceilings. Hammer was also used to integrate modeled or prefabricated objects with entities. Maya was used to model all of the props and objects that could not be made inside the *Hammer* editor, shown in Figure 3. 3DSMax was used to import the Maya objects and then export the objects as a file that could be read by the Source engine.

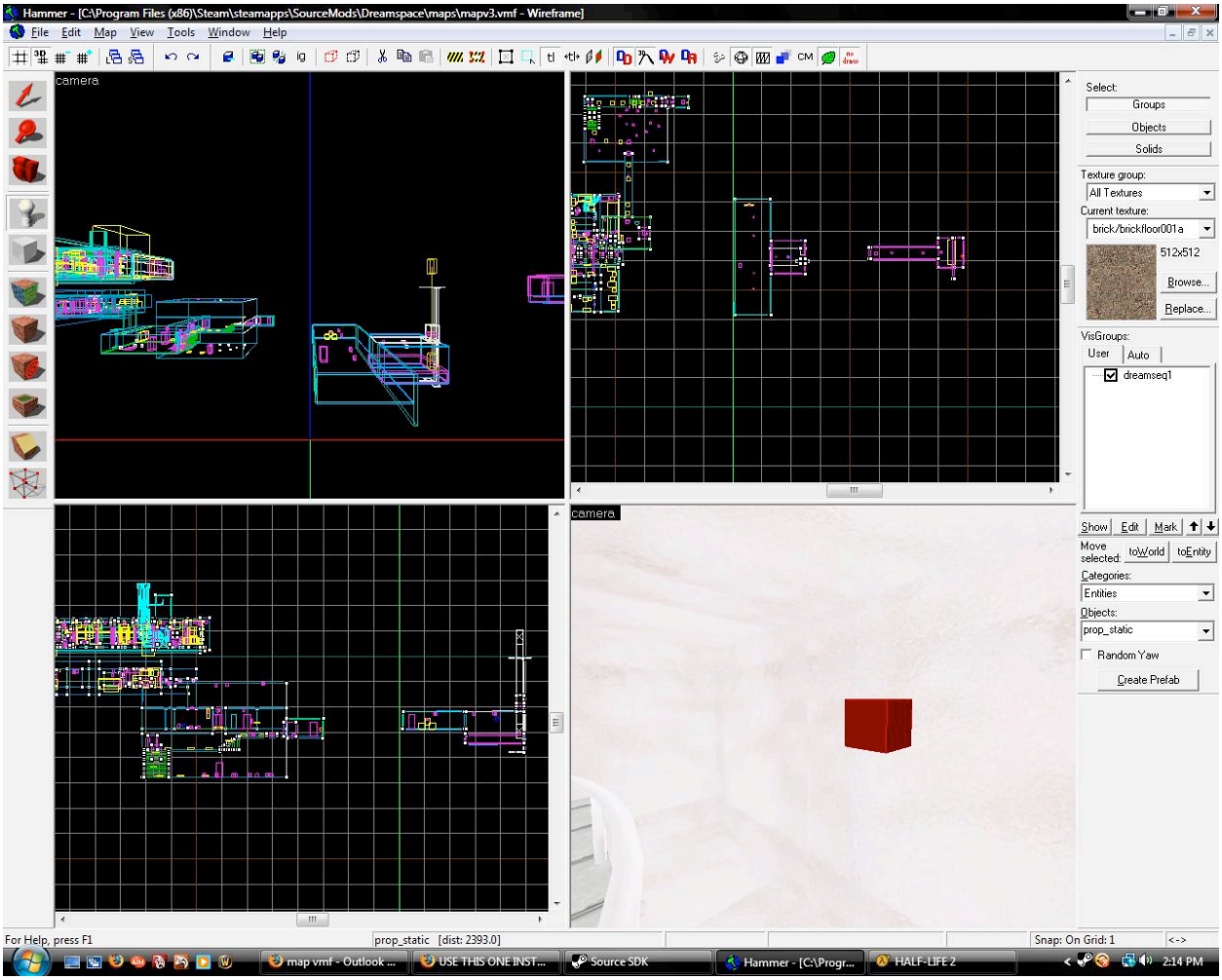

Figure 3: Hammer Editor

3 Technical Design

3.1. Engine Choice Rationale

The project was built as a game modification using the Source Engine, specifically the *Half Life 2 Single Player* code.

3.2. Specifications

3.2.1. Hardware Requirements

Valve's website (steampowered.com), these are the minimum and recommended system hardware requirements for *Half Life 2: Episode 2*:

- **Minimum:** 1.7 GHz Processor, 512MB RAM, DirectX® 8 level Graphics Card, Windows® Vista/XP/2000, Mouse, Keyboard, Internet Connection
- **Recommended:** Pentium 4 processor (3.0GHz, or better), 1GB RAM, DirectX® 9 level Graphics Card, Windows® Vista/XP/2000, Mouse, Keyboard, Internet Connection .

3.2.2. Software Performance Targets

Dreamspace will meet the following performance targets:

- Players: 1
- Video frames per second: Stable 60FPS on recommended-level hardware, 30FPS on minimum, without significantly compromising visual detail

3.3. Technical tools

Dreamspace was developed using Microsoft Visual Studio 2005. Although there is only one developer on the project, development still utilized a version control system (SVN). This allowed the project to revert back to an earlier release in case updates cause things to break or otherwise make the project unstable.

4 Story Overview

4.1. Backstory of World

Earth has become one of the thousands of planets colonized by humans. Each colony must receive goods from respective warehouse planets which store food, tools, and other essentials. The *MSC Jacob* is a freighter transporting the necessary goods to a far off planet. Faster-than-light travel has been invented so there is no need for technology such as hibernation chambers.

4.2. Characters

4.2.1. Protagonist

The unnamed protagonist signed up on the *MSC Jacob* as a simple technician. His duties include making routine repairs to the ship, keeping its life support systems well maintained and other day to day tasks. He has taken this job to try to forget his family. He has abandoned his wife who he no longer loves but is racked with guilt because he has left his kids without a father. He finds that his life on the ship has not allowed him to forget his past as easily as he thought and has now fallen into a deep depression. A short time before the events in the game the protagonist is in an accident that results in him ending up in the infirmary.

4.2.2. Voice on the Intercom

At first the voice on the intercom seems like the only other survivor of whatever happened to the ship. The voice guides the player through the game, outlining the goals for the player and also drives the plot.

4.3. Story Detail

The story will unfold through three acts, with Act 1 consisting of two areas and the other two acts both consisting of a single area, as shown by Figure 4.

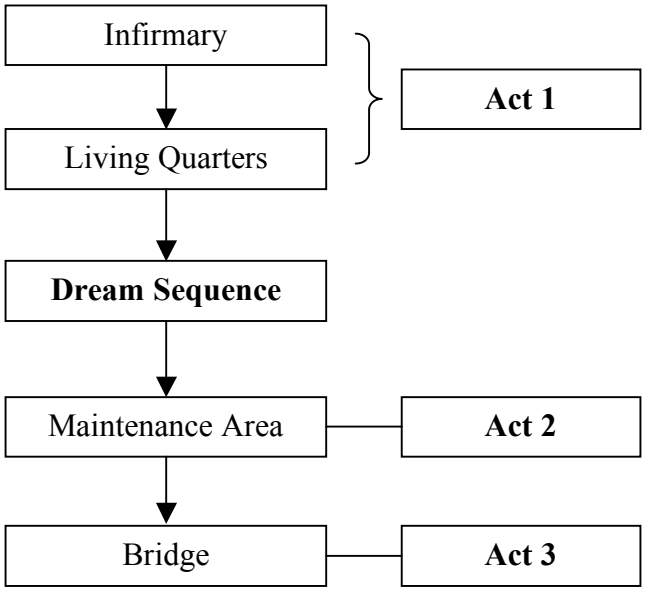

Figure 4: Chapter Flow

4.3.1. Act 1: The Awakening

The protagonist wakes up alone in the infirmary. With help from the voice on the intercom, he makes it out of the infirmary (shown in Figure 5) and into the living quarters. It is here he discovers that the ship is empty save for himself and a few insane crewmembers. The voice on the intercom tells him to come to the bridge, where they can figure out a solution together. In order for the protagonist to reach the bridge he must get through the maintenance area. The protagonist pieces together a code scrawled on the walls of the living area. This code allows him to unlock a room containing a blowtorch. The blowtorch lets the protagonist cut through a door leading to the locker room. The voice on the intercom beckons him to go through the vent in the room.

Figure 5: Infirmary

4.3.2. Dream Sequence

Falling through the vent, the protagonist finds himself in a room. His wife is sobbing next to a doll in the middle of the room. Realizing that the protagonist is there, she leads him down a hallway. The two fall into a large room with a staircase. The protagonist's wife beckons for him to traverse the staircase with her. At the top of the stairs, she runs into a bright light and disappears. The room is suddenly flooded with demons. The voice on the intercom tells the protagonist that he must defeat all his demons to continue. After the protagonist has done this, the voice tells him to move into the light.

4.3.3. Act 2: The Ascension

The protagonist finds himself in the maintenance area and finds a critical area flooded. To proceed he must drain the water. The protagonist manages to drain the affected area and makes his way to the bridge.

Act 3: The Revelation

The protagonist finally finds himself on the bridge. It is pristine and sleek unlike the rest of the ship. The protagonist makes his way down the shiny, almost too new hallway and reaches the room where the voice is coming from.

5 Puzzles

The gameplay of *Dreamspace* is focused on puzzles. The game opens with a tutorial puzzle to get the player comfortable with the games controls. The rest of the puzzles then focus on removing obstacles from the player's path and mainly need to be solved linearly.

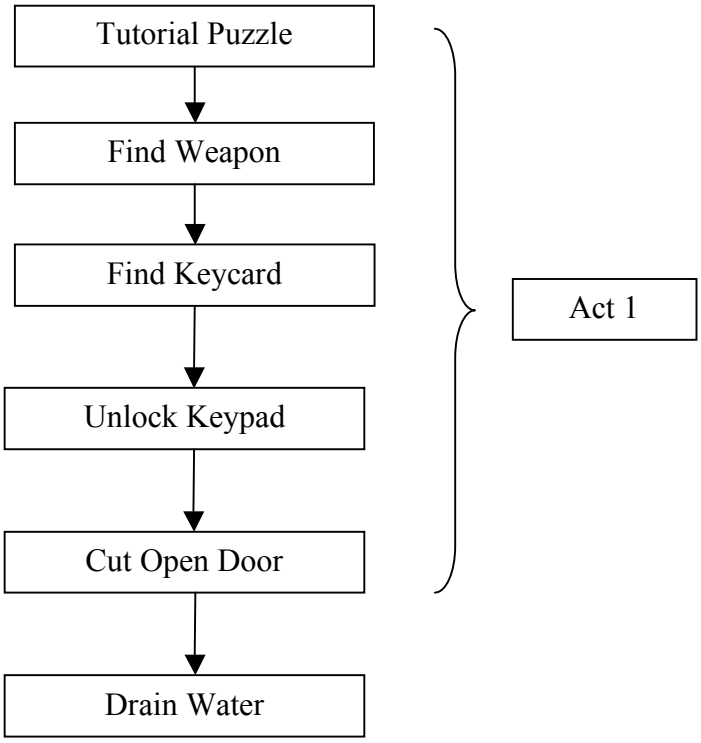

Figure 6: Puzzle Flow

5.1. Puzzles in Infirmary

5.1.1. Get out of first room

The player wakes up in the infirmary to the voice on the intercom. The voice informs the player that the room is locked and the player must find a way out.

5.1.2. Solution

The player picks up a gun near him and shoots apart the vent.

5.1.3. Get out of second room

The voice informs the player that the door in this room is jammed and he must find the crank that operates it.

5.1.4. Solution

The crank is behind a machine. The player must slot this crank into its holder and turn the crank to open the door.

5.1.5. Finding a weapon

The voice tells the player that he must make his way to the bridge by taking the elevator. However, in order to access the elevator, the player must have a keycard which is behind a jammed door.

5.1.6. Solution

There is a room off the main hallway of the infirmary that has frayed wires on its locking mechanism. The player must use the wrench, which is found in an adjacent unlocked room, on the frayed wires which will open the door. A crowbar is inside the room.

5.1.7. Getting out of infirmary

The door connecting the rest of the infirmary to the ship requires a keycard which the player doesn't have. The player must find the ship's doctor (shown in Figure 4) and get the keycard from him. The door to the doctor's office is jammed shut.

Figure 7: Doctor - Infirmary

5.1.8. Solution

The player must use the crowbar to break the door to the doctor's office. The doctor, who has been driven insane, attacks the player. After defeating the doctor, the player can pick the keycard off the doctor's desk.

5.2. Puzzles in the Living Quarters

5.2.1. Unlocking the Keypad

When the player enters the living quarters, he will notice a series of numbers scrawled on the walls.

5.2.2. Solution

The numbers in the living quarters are the digits in the code that unlocks the door to a room that has a blowtorch.

5.2.3. Getting out of the Living Quarters

The door to the locker room leading to the maintenance area has been welded shut.

5.2.4. Solution

The blowtorch allows the player to cut his way through the welded door.

5.3. Puzzles in Maintenance Area

5.3.1. Drain Water

The water flooding the maintenance area must be drained in order to gain access to the ladder leading to the bridge.

5.3.2. Solution

There is a valve nearby that the player can use to drain the water.

6 Project Workflow and Management

Dreamspace's design and development, which took place over three terms, required both team members to be constantly working on new tasks for the game. Only having 21 weeks to design and implement a full game required the team to keep in mind our goals each week so that we would stay on track until completion.

6.1. Task scheduling and assignment

The team used Sourceforge to keep track of tasks and assignments. All members of the team were free to assign each other tasks as they see fit.

6.2. Meetings and Communication

During the design phase of the *Dreamspace*, the team met at least twice a week. Each meeting lasted at least an hour. The first meeting each week was with the project advisor, who went over the week's work and commented on what needed to be focused on for the next week. These informal meetings usually started with the team updating the advisor on the development of the game. This was usually followed by discussion of any problems or setbacks that were encountered during the week or comments on design choices. The advisor usually closed with questions on design choices/advice and proposed work for the next week.

More formal meetings with the advisor took place once every few weeks. These were more formal status reports which took the form of PowerPoint presentations. The team updated the advisor on the status of the whole project along with an updated design document. The presentation consisted of a formal introduction, slides on the planned design of the game, and a summary of what the game will contain.

All other meetings were internal team meetings. During these meetings, the design and story of the game was planned out and action items were assigned.

7 Implementation

7.1. Changes to Design

7.1.1. Changes due to Time Constraints

There are two major tech features in *Dreamspace* that were unable to be implemented due to time constraints:

- An inventory system that allows the user to drag and drop items to combine them
- A mini map that would allow the player to see points of interest on the current level

The main reason other than time that these features were not implanted was unfamiliarity with the *Source SDK*. Many hours were spent trying to figure out how to make custom heads-up display (HUD) elements, such as the inventory, which do not appear in the current version of the Source engine, and making them appear on screen.

Development of the inventory was stopped at the beginning of the implementation phase. The player must now use the item they currently hold. The player can not hold more than one object at a time.

An alternative to the inventory system was suggested but due to time constraints was unable to be implemented. Items that needed to be picked up were to be classified as "new weapons" that would be put into pre-existing weapon "buckets". The items would then be treated as normal weapons that would be able to be picked up and stored in the weapon HUD element and accessed with the number keys as per-usual in *Half Life*.

The mini map system was close to becoming implemented. Behind the scenes, the map is completed because all necessary classes, scripts, and files have been created and linked. However, the map refuses to show up in-game. This is probably due to a wrong pointer or script but because of the lack of documentation for the SDK. The minimap was not a major feature of the project, and due to the time constraint, it was dropped.

There were also some major art features in *Dreamspace* that we could not get into the game due to time constraints:

- Animated cut scenes for the dream sequences
- Fully skinned character model

Both of these features were cut for lack of time.

The same problem was evident for the animations, which would take at least four to five days each to export models and props from Source to 3DStudio Max and create animations for every scene. Animating a simple scene with props is time consuming in itself, but a scene with fully rigged biped characters would be even more difficult because the animations would have to seem very human-like and that would take much more time.

7.1.2. Changes due to Technical Constraints

Due to unimplemented technical features, namely an inventory system, certain aspects of the game needed to be changed. As it stands now, the player cannot pick up more than one item at a time. Fortunately the impact of this on puzzles is relatively minor. Instead of combining items in the inventory itself, items that need to be combined are "slotted" together in the game world and the resulting item can be picked up by the player and used.

The only puzzles that were removed because of this were the dog puzzle, where the player would have to combine a pheromone vial with a tennis ball to get the attention of a dog blocking a certain door, and the gas room puzzle. The gas room puzzle required the player to pick up a mask and a towel so that the player could use the wet towel and the mask to gain access to the room that is filled with gas.

The art half of the team ran into major technical problems with the Maya to Source art pipeline. The original pipeline proposed in the design phase, which started directly from Maya, looked like this:

- Create Model in Maya-> Export SMD file (Source's file format for storing the objects list of vertices and edges)of reference object and physics hull-> Create both texture files that the Source engine recognizes(VTF and VMT) that can be used in Source->Create QC file (list of locations for the texture files and SMD files -> Compile QC file, SMD files, and texture files-> Create MDL (model file that can be used in Source and Hammer)

The art team realized once the implementation phase started that the pipeline took some tweaking but was still confident it could work, despite the fact that the only working exporter for Maya to Source was written over 8 years ago for compatibility with Maya versions 4-7. After many weeks of unsuccessful attempts, the art team looked to using an idea suggested by a fellow student. The idea was to export from Maya to 3Ds Max and then export the SMD files from there. The successful pipeline from Maya to 3Ds Max is as follows:

- Create Model in Maya-> Export as FBX to 3Ds Max-> Create physhull around model and make the entire model 1 individual smoothing group-> Export SMD file of reference object and physhull-> Create texture files (VTF and VMT) that can be used in Source->Create QC file-> Compile QC file, SMD files, and texture files-> Create MDL

7.2. Artistic Implementation

7.2.1. Props

Since the game has three large areas to the map, specific props were needed to fill each section. For example, several pieces of hospital equipment were made to fill up the rooms in the infirmary. Twenty-seven props are implemented in the three levels of the game. All props fit the style of the area they are in, and were created with a low-polygon count to keep the frame rate at a playable level.

Figure 8: IV Render

All of the smaller props are physics props and will react accordingly when hit with another object. The QC file that is compiled to create the model file used in Hammer allows the developer to input a material type for Source to recognize and give the appropriate attributes such as the sound made when an object is hit. This means that if a material type of metal was given to a prop, the prop would sound metallic when hit with other objects.

7.2.2. Textures

All textures were either created by the art team in Adobe Photoshop, or taken from cgtextures.com and modified to be used in the Source Engine. To get the textures working in the Source Engine either a custom image or a preexisting image was used as the base for the texture. A new Photoshop document that has a size equivalent to a power of two (128x512, 512x1024, etc.) was created and was filled with a previously made pattern. The image was offset and fixed so that it would tile without a seam. It was then saved as a 24-bit Targa file and opened in the program VTFEdit, created by "Wunderboy" (Wunderboy.org) to create VTF (Valve Texture Files) and VMT (Valve Material Type) files. In VTFEdit, the VTF is opened and then saved as a VMT to be used in the Source Engine. The textures were all created with the location within the

level in mind. Everything placed in the area of the infirmary, living quarters and maintenance area was given an old, worn down, rusted look to go along with the overall style of the game.

The textures were mapped mainly with the use of planar mapping to different faces of the object. For some of the cylindrical objects in the game such as the legs to the hospital tray or the robot laser, cylindrical mapping was used.

7.3. Technical Implementation

7.3.1. Source SDK Extension

Source SDK extension was minimal. An effort was made to extend the Source SDK to allow for a mini map HUD element. Changes were made in MapOverview.cpp, HudLayout.res, and HudAnimations.txt to accommodate the map element. A level overview texture was also made. However, some file or call in the process did not work and the mini-map does not show up in game.

7.3.2. Scripting

Scripting was done in Valve's Hammer World Editor. A map or level running on the Source Engine is populated with objects called entities. These entities represent everything from walls and NPCs to objects that govern AI behavior to Player and NPC spawn points. All these entities can be scripted to interact with each other through input and output functions. Manipulation of these functions allows for behaviors and interactions that go beyond what an entity may originally have been coded to do. All puzzles were created using Hammer.

7.3.3. NPC Behavior

Two Non-Playable Characters (NPC's), and four spawning groups of enemies were implemented. To make NPCs interact with the environment and player, AI behavior patterns had to be scripted. The *ai_goal_actbusy* entity was used to give a behavior pattern to an NPC that it would carry out while not interacting with the player. The behavior is guided by the placement of hint nodes that would tell the NPC what to do. Each node would tell the NPC what to do in a certain area (play a idle animation, sit in a chair, etc.)

To govern interaction with the player, the *ai_relationship* entity was used. This entity governs the actions NPCs have with other NPCs or the player.

The wife NPC in the scripted dream sequence was heavily governed by the *scripted_sequences* entity.

7.3.4. Scripted Objects

Objects which move in game such as the elevator or blowtorch weld path were mainly governed by the *func_tracktrain* entity.

7.3.5. Scripting Examples

7.3.5.1. Doctor Behavior

The doctor behavior was created by using the *ai_relationship* entity and *ai_goal_actbusy* in conjunction with other trigger entities. The *npc_combine_*s entity was used as a base for the doctor NPC. The *ai_goal_actbusy* entity was used to tell the doctor to move through the hint nodes that were placed in the room he patrolled. These nodes told the doctor that he could move in those areas. An *ai_relationship* entity was used that changed the default neutral behavior of the NPC to a hate behavior, used with the *npc_combine_s* entity.

The doctor patrolled around the room when a trigger in the hallway was activated by the player. When the player enters the room, a trigger that covered the floor of the room set the doctor's behavior to its default hate behavior. As long as the player stayed within the room, the doctor chased them. When the player has left the room, the trigger changes the doctor's behavior to neutral and tells the doctor to begin patrolling again. This prevents the doctor from chasing the player out into the hallway where there are no nodes to govern doctor movement.

7.3.5.2. Door to locker room

The door to the locker room can be cut open by the blowtorch object. Getting these two objects to work together took many entities and triggers to work in conjunction. The blowtorch was created from scratch in the Hammer editor using brushes.

After the blowtorch was made, an *env_steam* entity was attached to the end of it to give it the effect of shooting flames. An *env_sprite* entity was attached to a *func_tracktrain* entity to allow the player to see the starting point of the cutting path on the door. The *func_tracktrain*'s movement is governed by *path_track* entities, points that tell other entities where to go. Each

path_track entity holds the name of the next *path_track* entity on the path. In this case, *path_track* entities were placed on the three remaining corners of the door.

When the blowtorch interacted with a trigger that was attached on the *func_tracktrain*, it sent messages to enable the *env_steam*, *func_tracktrain,* and *env_spark*. This made the weld point move around the door while shooting sparks and smoke while the blowtorch was in contact with it. An *env_physexpolosion* entity was used to blow the door backwards when the *func_tracktrain* reached the end of its path.

8 Evaluation

When we first began this project we had a few main goals we wanted to meet:

- **Story –** We wanted to give the player an interesting story that would make them want to explore the ship
- **Environment -** Have a futuristic environment that looks realistic and lived in instead of the generic shiny newness that most science fiction have
- **Puzzles -** Have puzzles that would drive the story along and challenge the player

8.1. Story

Unfortunately, much of the story elements in the game are missing. Out of two planned dream sequences, we only managed to fit in one. Letters from the protagonist's wife and other background information on the protagonist did not make it into the game either due to technical shortcomings or lack of time.

8.2. Environment

The team feels that they have succeeded in creating a realistic feeling environment for the player to interact with. The ship look rusted and worn and the player can interact with many elements of the ship.

8.3. Puzzles

Although not all puzzles that were originally planned made it into the game, a majority of them are present in some form. The puzzles in the infirmary area are complete as are most of the puzzles in the living quarters and maintenance sections.

8.4. Testing

Testing of *Dreamspace* consisted of an internal test. The game was played by six testers who were friends of the team. None of the testers had previously played the game before, including one that had never played a first person shooter. Players played through the most complete area of the game, which was the infirmary.

During internal testing, it was noted that hints needed to be given more liberally and that certain items of interest were hard to spot. These issues are addressed in the final submitted build of *Dreamspace*. Hints are now given more often to the player and a glow was added to items that testers found hard to find.

The overall feeling towards the game was positive. Testers stated that the story piqued their interest and wanted more of the game. Puzzle difficulty needed to be tweaked however, as the more experienced game players found the puzzles were too easy while the less experienced testers needed more hints.

Appendix A – Reference Material

The following works contain useful information used in the implementation of the project:

"Half-Life 2 News and Tutorials." Interlopers. URL: http://developer.valvesoftware.com/wiki

"SDK Wiki." The Valve Developer Community. URL: http://developer.valvesoftware.com/wiki

"Source Tools." Wunderboy. URL: http://wunderboy.org

Appendix B – Original Project Pitch

8.5. One Sentence description

Dreamspace is a story driven first person adventure game set in space made using the Source Engine.

8.6. One paragraph description

In the bowels of the space freighter GMS Jacob a man wakes up with no recollection of what has transpired on the ship since the accident that has put him in the infirmary. After searching the rest of the ship, he notices that something has changed about the once familiar hallways and decks he inhabited. He sets out to find the video logs which record all happenings on board. In his search for answers he experiences odd occurrences which reveal to him a shocking truth.

9 Gameplay Overview

Dreamspace is a first person adventure game, focusing on storyline and challenges that will impede the player's progress. The user will be guided by an entity over an intercom that will be on his wrist. Beginning in the infirmary, the game will include three different chapters which consist of different portions of the ship. Between each chapter there will be a dream sequence. During these sequences, the player will be engaged in much more combat than the rest of the game. The enemies will be some sort of horror amalgamation which will be related to his past. We are striving for visceral, tense feel for the combat so weapons will be everyday objects on the ship. Finally, during non combat phases of the game, there will be a proliferation of puzzles throughout the ship. These will mostly consist of logic puzzles.

10 Artistic Design and Vision

The artistic theme of *Dreamspace* is very futuristic, as the game is set somewhere in the future where space travel is commonplace. A majority of the game will take place on the spaceship *MSC Jacob*. The overall appearance of the ship will be worn from years of use and will be further worn due to the fact it is a transport freighter. The style of the ship will be inspired by spaceship interior/exterior designs from the *Alien* movies, *Doom* and other space movies such as *Star Wars*. At other points in the game the player will slip into dream sequences where the setting will be shifted from the ship to other locations such as the protagonist's home.

10.1. Level Design

The levels will all be on the ship and take the player through three different sections of the ship. The living quarters, which will contain the crew's sleeping quarters, bathrooms, and infirmary. The objects in these rooms (furniture, appliances, etc.) will be very basic in shape and form. This will allow the game to display the uniformity between all the quarters and give the player the feeling that the ship they are on is fairly large and at one time, supported many crew members. The textures for the ship's interior will need to be created to show years of use. The maintenance area will be created to give the feel that all the major work done to the ship is done in this area. Several pipes and terminals will be used to signify this. The last area of the ship, the bridge, will be completely different from the rest of the ship. While the player has been living through a nightmare in the first two sections of the ship, the player has the truth revealed to him and realizes that he is dead. The bridge will be completely white and extremely clean. All of the surfaces will be very shiny and polished in the light. The white effect will give the player the idea of heaven after they realize that the protagonist is already dead.

The dream sequences will appear very surreal and will give the player the feeling he or she is either unconscious or asleep. The camera will have a constant blur effect in the dream sequences to add to the feeling of surrealism to the player. The two locations in the dream sequences are very different. The first sequence takes place in the protagonist's home on his home planet. The setting will be very bright to give the appearance of a warm, inviting house. The player will get the feeling in the sequence that the home the protagonist had was once happy. The second dream sequence will be much more phantasmagoric. The setting will be a beam of light, upon which the character will walk towards a door of light while trying to get past NPCs who are demon like in appearance.

10.2. Character Design

The protagonist in the story is a male, approximately six feet tall and in his 30's. He has dark brown hair and wears an outfit similar to a flight suit/mechanics jumpsuit for the entire game. His design will appear very realistic as we will likely be using a model from the Source Engine and retexture it. The character's movement will be realistic and lifelike. There will only be a few opportunities for the player to see the protagonist in the game. In the bathroom in the living

quarters the protagonist can see himself in the mirror in the bathroom and at the end of the game the player will see himself on the bridge.

10.3. Concept Art

Both concept art and reference art will be used to create the look of the spaceship interior, NPC's and the protagonists. The demon NPCs will borrow from several pieces of reference art of demon-like creatures. The main goal of the demon's appearance is to be very menacing, with bone or tissue exposed on their body, an overall scary look on the demon's face, and when dealing with the movement of the demon, a very creepy walk or run.

The outfit for the main character will be borrowed from several examples to merge into a single outfit. The protagonist's outfit should look as though it belongs to a member from the crew who works with his hands repairing things on the ship. Because of this the outfit should look like a mechanic's jumpsuit, but since the game is set in space, the outfit should also have some flight suit qualities to it.

10.4. Heads-Up Display

The goal of the HUD in *Dreamspace* is very minimalist. While playing, the screen will be completely empty. The players health will be represented by a clear screen (if the player is unharmed), or a red flash around the perimeter of the screen, coupled with a loud, quickened heartbeat (if the player is being harmed). The inventory system will be a window that pops up on screen. The inventory will consist of a simple drag-and-drop system and will display all of the items that the player has picked up so far in the game. In the inventory system the player will be able to combine, use or equip any item located in there.

The map will also be available to the player as a smaller window in the inventory system. This means that when the player opens up the inventory system, they will be able to see the map in the lower right hand corner of the screen, and will have the option of seeing an expanded version of the map.

10.5. Artistic Tools

Everything that has to do with art in *Dreamspace* will be created using both Source's SDK Hammer editor and Autodesk's Maya. Maya will be used for the overall layout of the levels while Maya will be used to create the smaller objects in the levels.

The models for the protagonist and the NPCs will be created using the models that already exist within Source. While the models will be recycled, the textures will all be independently created in Adobe Photoshop. The HUD and menu system will be created using Photoshop as well. Audacity and Digital Performer will be used to create and edit the sound and voiceovers used.

11 Technical Design

11.1. Engine Choice Rationale

The project will be built as a total conversion mod using the Source Engine, specifically the *Half Life 2 Single Player* code.

11.2. Specifications

11.2.1. Hardware Requirements

Valve's website (steampowered.com), these are the minimum and recommended system hardware requirements for *Half Life 2: Episode 2*:

- **Minimum:** 1.7 GHz Processor, 512MB RAM, DirectX® 8 level Graphics Card, Windows® Vista/XP/2000, Mouse, Keyboard, Internet Connection
- **Recommended:** Pentium 4 processor (3.0GHz, or better), 1GB RAM, DirectX® 9 level Graphics Card, Windows® Vista/XP/2000, Mouse, Keyboard, Internet Connection

Dreamspace will not be as visually intensive as *Half Life 2: Episode 2* so it may be able to run comfortably on configurations closer to the stated minimum requirements.

11.2.2. Software Performance Targets

Dreamspace will meet the following performance targets:

- Players: 1

- Video frames per second: Stable 60FPS on recommended-level hardware, 30FPS on minimum, without significantly compromising visual detail

11.3. Technical tools

Dreamspace will be developed using Microsoft Visual Studio 2005. Although there is only one developer on the project, development will still utilize a version control system, such as Subversion(SVN). This will allow the project to revert back to an earlier release in case updates cause things to break or otherwise make the project unstable.

Appendix C – Asset Lists

Texture List

Name	*Description*	*Sources*	*Completion %*
Metal 1	512x512, for props	HL2, photography	100%
Metal 2	512x512, for props	HL2, photography	100%
Metal 3	512x512, for props	HL2, photography	100%
Metal 4	512x512, for props, level	Photography	100%
Metal 5	512x512	Photography	
Concrete 1	512x512, for floor and props	HL2, photography	100%
Blue Fabric	512x512, for Living Quarter beds	HL2, photography	100%
Rusted Metal	512x512, for props	HL2, Photography	100%
Cardboard	512x512 for props	HL2, Photography	100%
Comp Texture	512x512, for props	HL2, Photography	100%
Black Vidscreen	For several props	Original creation	100% (Original)
IV Bag	For Hospital IV prop	Original creation	100% (Original)
Duct Tape	For props	Original creation	100% (Original)
Robot Metal	For robot	Original creation	100% (Original)
Porcelain	For props	Photography	100%
Yellow Button	For props	Original creation	100%
Bridge White1	For bridge area	Original creation	100%
Bridge White 2	For bridge area	Original creation	100%
Satellite Metal	For props	Photography	100%
Keycard	For props	Photography	100%
Red Button	For props	Original creation	100%
Tire tread	For props	Photography	100%

Towel	For props	Photography	100%
Transformer Gauge	For Props	Photography	100%
Wheel Black	For props	Original creation	100%
White fabric	For props	Photography	100%
Warning sign	For props	Photography	100%

Model List

Name	*Completion %*
Buzzsaw	100%
Cabinets	100%
Doc key card	100%
Doctor desk	100%
Dream sequence table	100%
Duct tape roll	100%
Fence	100%
Hospital bed	100%
Hospital IV	100%
Hospital tray	100%
Infirmary cabinet	100% (Original)
Living Quarters bed	100% (Original)
Duct tape	100% (Original)
Stand	100% (Original)
Robot complete	100%
Robot damaged	100%
Robot satellite	100%

Robot laser	100%
Robot power supply	100%
Robot wheel	100%
Safety goggles	100%
Scalpal	100%
Sink	100%
Stat machine	100%
Storage shelf	100%
Syringe	100%
Towel	100%
Transformer	100%