Project Number: RL1-P008

# Petrified

Design and Development of a Multiplayer Survival/Horror Game

Interactive Media and Game Development

A Major Qualifying Project Report
submitted to the faculty of
WORCESTER POLYTECHNIC INSTITUTE
In partial fulfillment of the requirements for the
Degree of Bachelor of Science

by

Skyler Clark, Chris Drouin, Matt Fabian

Advised by

Professor Robert W. Lindeman

# Abstract for the

# Design and Development of a Multiplayer Survival/Horror Game

by
Skyler Clark, Chris Drouin, Matt Fabian

This is an Interactive Media and Game Development Major Qualified project, focused on designing, implementing, testing, and releasing a full-conversion game modification based on Valve Software Corporation's *Source* Engine. The game, titled *Petrified*, is a team-based multiplayer game, with a heavy emphasis on unsettling atmosphere and challenging, engaging play for players on both teams.

*Petrified* revolves around two distinct teams of players: the human Mortals, who are attempting to survive until dawn, and the supernatural Watchers, who are attempting to stalk the Mortal players and attack them, thereby converting the attacked players to the Watcher team. While Mortals have relatively free range of movement, Watchers are largely restricted to possessing statues that can teleport within a limited range, so long as they are not visible to Mortals on either end. To attack, Watchers must leave their statues by going into an astral form; in this state, they can be stunned and delayed by Mortals wielding flash-sticks – a brand of high intensity narrow beamed flashlight. Watchers that stray from their statues for too long are subject to penalties. The game ends when all Mortals have fallen to the Watchers, or when dawn breaks (i.e. after a set time limit).

The project consisted of three distinct phases. The team made use of iterative development methods throughout all three phases in both its designs and technical implementations.

The first phase was the design phase, in which the team documented its technical, artistic, and gameplay-related plans. These plans were revised multiple times based on peer review and advisor recommendation, and were ultimately compiled in the form of a design document.

In the second phase the team implemented the game, working with the *Source* engine to define game rules and behaviors and creating artistic assets to match.

Finally, with the completion of the implementation, the team released the game for controlled beta-testing and evaluation. This testing took place both in-person on the WPI campus and online. The online testing was coordinated through a *Petrified*-centric community group within Valve's gamer-oriented social network, *Steam*. For both sets of tests, testers were asked to give feedback on their experience and note any bugs or glitches that they encountered. The team took the feedback and bug reports received and polished and updated the game, while staying within the bounds of the original design. With *Petrified* tested and polished, the team released the game as a free download available through www.petrifiedmod.com; the game can be played by anyone with a Steam account, a copy of *Half-Life 2: Episode 2*, and an appropriately equipped computer.

# Table of Contents

## List of Figures

## List of Tables

# 1 Introduction

*Petrified* is a multiplayer survival/horror game, set in a graveyard guarded by possessed statues known as Watchers. The Watchers form one of two teams; their goal is to sneak up on and attack the human Mortal players who have trespassed in the graveyard, thereby converting the Mortals into Watchers. The Mortals must attempt to survive the Watcher onslaught until dawn, using only their quick reflexes and flash-sticks for defense.

This document discusses the design of *Petrified*, covering its gameplay mechanics, artistic design, and technical architecture through sections 2, 3, and 4. It also details the process of implementing that design, in section 6, and the testing and evaluation process used to gather feedback on the game in section 7.

## 1.1 One Sentence Description

Don't turn your back, or you may find your heart turned to stone – and in *Petrified*, if you want to survive until dawn, you'll need to keep a keen eye on the waiting statues, lest they add your soul to their ranks.

## 1.2 One Paragraph Description

Can you last until dawn? In Petrified, a Source-powered multiplayer game, a band of humans is pitted against an array of statues possessed by evil Watchers—the protectors of the graveyard. The Watchers approach while the Mortals' backs are turned, intending to steal their enemies' souls and swell their own numbers. As a Watcher you can enjoy the thrill of the hunt waiting for the right time to strike. As a Mortal you'll be running around a chilling landscape that never moves yet constantly changes. Whichever side you're on, look forward to endless quick, atmospheric action.

# 2 Gameplay Design

In *Petrified*, Mortals try to survive until dawn, while Watchers, spectral beings that inhabit statues, wait to sneak up on them and turn them to stone. Players control both sides, but the selection of which player lands on which team is determined in part by player performance in-game.

When a round begins, there will only be one player on the Watcher team. Throughout each round, Watcher players will attempt to attack the Mortals. When an attack is successful, the Mortal disintegrates, and the affected player is transferred to the Watcher side, in control of a statue.

The end of the round is signified by dawn, or by the elimination of the Mortals. If there are any Mortals remaining when the sun rises, the Mortals win the round, otherwise, the Watchers win.

## 2.1 Mortals

Mortal players begin each round in a random location in a graveyard. As they begin to move and play, the world will appear stationary, but the location of the statues will keep changing. From the Mortal player's perspective, the game will shift between periods of uneasy calm, in which the Watchers approach but do not reveal themselves or attack, and frantic moments of fight-or-flight as the Watchers attack and multiply. Because of the sound effects associated with Watcher attacks, the Mortal players will always be aware of the general location of aggressive Watcher players; this knowledge helps raise player tension and excitement.

### 2.1.1 Objectives

The primary objective of each Mortal player is to evade Watcher attack until dawn, which was originally to be represented by the lightening sky. If the Mortal players cannot survive, they are encouraged to evade death as long as possible.

This goal does not encourage teamwork, but in keeping one's teammates alive a player will be able to survive longer, because he or she will have other players to watch his or her back. So while survival is a selfish goal, this aspect of survival will encourage the use of teamwork.

### 2.1.2 Abilities

The Mortal player's primary tool in managing and thwarting the progress of the Watcher-possessed statues is simply his or her field of view; any Watcher-possessed statues visible to the Mortal become unable to teleport (though they can still swap to other statues, as described in section 2.2.2).

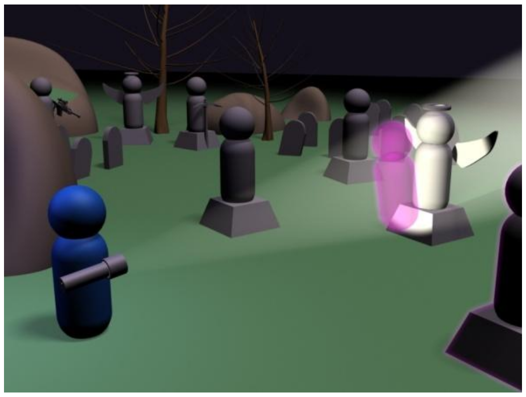


*Figure 1 – Concept Render – Flash-stick Attack*

Mortals have another defensive ability: they carry flash-sticks, which let out short bursts of light. During the design phase, the team created a series of mock-ups to demonstrate some of the gameplay mechanics, one of which (Figure 1) shows a Mortal player hitting a Watcher ghost with a flash-stick. These bursts can stun the Watchers' astral form[1], which is used for attack, providing time for the Mortals to escape or re-position. The flash-sticks are not wholly reliable, however; there is an internal system of logic that determines whether or not they work, which is shown in Figure 2.

[1] Hereafter the "astral form" of a Watcher will be referred to as a "ghost".

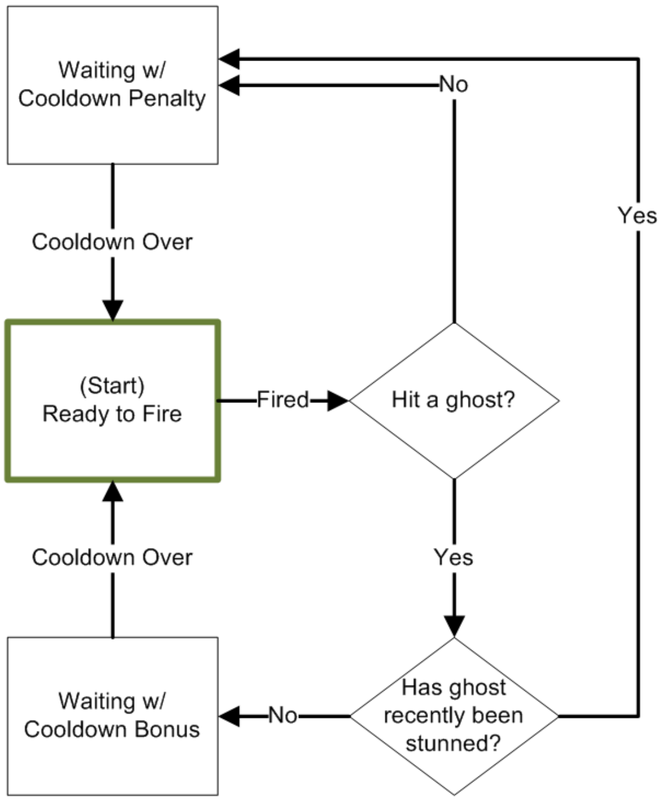

Waiting w/ Cooldown Penalty
No
Yes
Cooldown Over
(Start) Ready to Fire
Fired
Hit a ghost?
Cooldown Over
Yes
Waiting w/ Cooldown Bonus
No
Has ghost recently been stunned?

*Figure 2 – Flash-stick mechanics flow chart*

If the flash-stick is within its cool-down phase and a Mortal player attempts to use it again, the stick will malfunction, spewing out purple sparks. As shown in Figure 2, however, successive hits on different ghosts will temporarily decrease the cool-down phase, enabling skilled players to take on multiple attacking ghosts if they can time their bursts appropriately.

Mortal players additionally have the option to use voice communication that will address all members of their team. They can also communicate through a text-based chat system (for players lacking a microphone or with limited-bandwidth connections). Some information is blocked from the Mortals' view, however. Mortal players do not receive updates on other players' deaths. While they can hear the sound effect associated with death, they are not told who has been turned to the Watcher team.

### 2.1.3 Controls

Mortals are controlled through relatively standard first person shooter (FPS) controls by default. The standard configuration will be familiar to anyone who has played a Source-based game, such as *Half-Life 2*, *Counterstrike: Source*, or *Team Fortress 2.* The controls are shown in *Table 1*.

*Table 1 – Default Mortal Control Scheme*

| **Input (key or action)** | **Behavior** |
|---|---|
| Move mouse | Perspective follows mouse (enables turning, looking up/down) |
| Click left mouse button | Attempt to use flash-stick (see Figure 1 for behavior) |
| Press/hold [W] | Walk forward |
| Press/hold [S] | Walk backward |
| Press/hold [A] | Strafe (walk sideways) left |
| Press/hold [D] | Strafe (walk sideways) right |
| Hold [Shift] | *Originally* 'Run', *removed so that Mortals have only one consistent movement speed* |
| Press [Space] | Jump |

Mortals players can re-configure this default control mapping by accessing the "Options" window from the main menu (which is itself accessible by hitting [Esc] at any time in-game).

### 2.1.4 Projected Player Strategies

The most natural initial strategy for the Mortal players will be to bunch together, seeking safety in numbers and increased vision coverage. Even two Mortals back-to-back, as shown in Figure 3, can cover each others' blind spots to some extent.

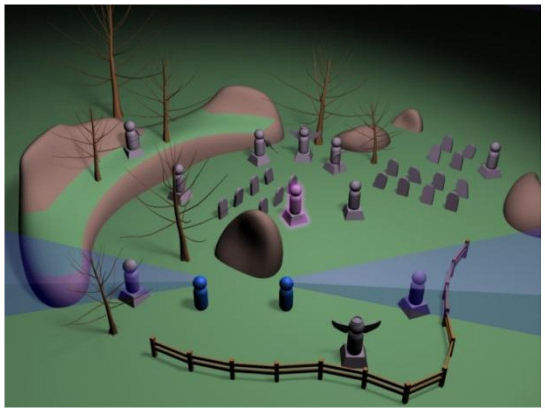


*Figure 3 – Concept Render - Grouped Mortal Defense*

Mortal players may seek to move in coordinated formations that cover wide spaces with their combined field of view, effectively freezing any nearby Watchers. Such behavior, however, will ultimately contribute to the Watcher team's arsenal of abilities, as doing so will help Watcher players gain powers through the Fear Point system (see section 2.2.3).

Many Mortal players may also seek shelter in physical features of the environment in order to limit the ability of Watchers to approach. They may position themselves against walls or in corners. While the Watcher statues themselves cannot travel through physical obstacles, their ghosts can pass through some of them. Basic Ghost attacks can travel through physical obstacles like walls, rocks, and trees. Experienced Mortal players will probably not rely on physical shelter, but they will need to know the map layout to efficiently escape from attacking statues.

Another strategy is for Mortal players to position themselves in regions with few statues. This may tend to separate the Mortals, however, making it easier for Watchers to track them down and eliminate them. Small, coordinated groups of Mortals may be able to strike the best balance between vision coverage, mobility, and Fear Point marginalization.

## 2.2 Watchers

Watchers are supernatural beings that typically inhabit or possess statues. They are guardians, seeking to protect the area entrusted to them. They can leave their statue shells to attack trespassing Mortals; though emerging as ghostly astral projections makes them visible and vulnerable to attack. In the original design, Watchers derived some of their capabilities from the statues that they inhabited; presently, all statues have the same abilities.

The Watcher team begins with only a single player, and inhabitable statues placed in strategic and atmospheric locations around the map. For the first game on a new server, the Watcher player is chosen randomly from the players. In all subsequent games, the initial Watcher is chosen randomly from amongst the pool of surviving Mortals, if there are any, or purely randomly if there are no surviving Mortals.

Under the original design, the player who was picked was to be given a choice regarding his or her role as a Watcher in the following round, and if this player declined, a new one would be offered the opportunity instead. This behavior was not implemented, due to time constraints. In practice, as rounds proceed rapidly, no player will be stuck in an undesired role for an extended period of time.

The player who is selected to be the first Watcher starts in a statue located some distance from the starting cluster of Mortals. As the round progresses, the Watcher team will grow in size as Watcher players attack and petrify Mortals, converting them to the Watcher team. The tide of the game turns much as it does in a "zombie" style game, with an ever-increasing number of "infected" players turned against their former comrades, trying to convert them in turn.

Unlike the Mortals, who view the game entirely from a first person perspective, Watchers switch from a third-person perspective, when they are inhabiting a statue or using their regular attack, to a first-person perspective when they perform their lunging special attack. Also, unlike the Mortal view, which is entirely free of interface elements, the Watcher view contains cues related to knowledge that Watchers should logically have, such as teleport destinations, inhabitable statues, and fear points.

### 2.2.1 Objectives

The Watcher players' goal is to capture all of the Mortals – that is, successfully attack them so they are switched to the Watcher team. It is only possible to lose as a Watcher if the team allows Mortals to survive until dawn.

As the Watcher team grows, strategy and teamwork will become important to efficiently capture the remaining Mortals, who will likely rank amongst the more-skilled and experienced players. On a more abstract level, the Watchers also serve as the puppet-masters or directors of the game; as the antagonists, they deliver the most direct frights to the Mortal players, and their actions prior to attack (teleportation and swapping) serve to develop the atmosphere.

### 2.2.2 Baseline Abilities

Watchers all possess three basic abilities. They can teleport, moving the statue that they inhabit. They can swap statues, leaving their current statue uninhabited. Finally, they can leave their statues to attack.

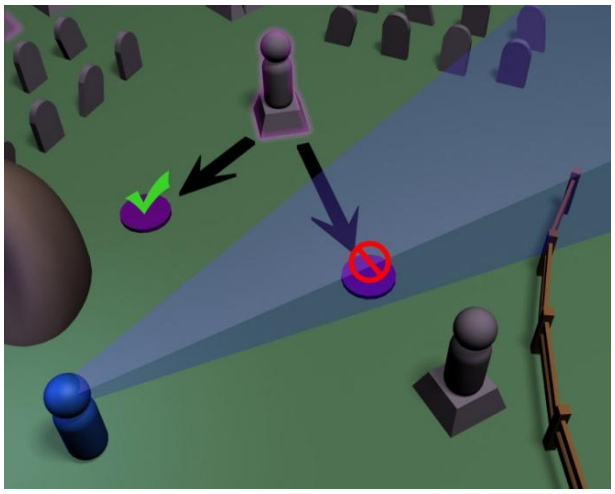


*Figure 4 – Concept Render – Watcher Teleportation Mechanics*

Teleportation enables Watchers to quickly close the distance between themselves and the Mortals. Watchers can teleport within a limited range, though only when out of sight of any Mortal players. Furthermore, even if the Watcher player is not within the Mortal's field of view, he or she cannot teleport into a region that is visible to a Mortal, as demonstrated in Figure 4.

To teleport, Watcher players will select the region on the ground that they wish to teleport to with the mouse; the third-person camera tracking the statue will pan left and right to follow the mouse as it moves across the terrain. Figure 5 shows how this works in-game: the Watcher statue itself is displayed as transparent to the Watcher player, so that the player can see the terrain and passing Mortal players without resorting to awkward camera manipulation.

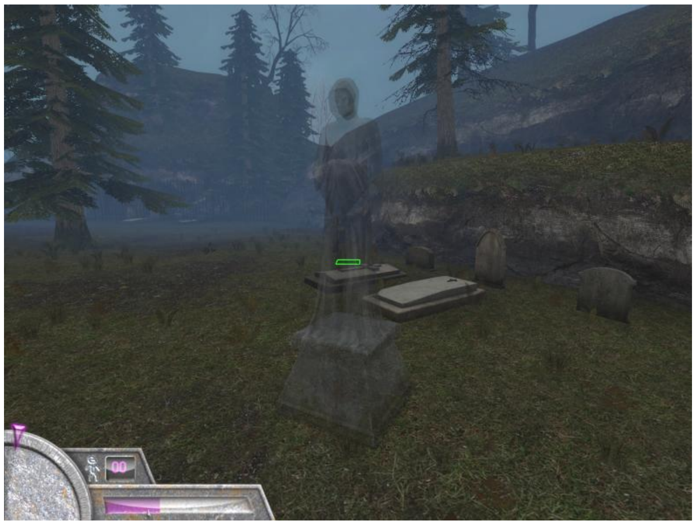


*Figure 5 – Screenshot – Watcher Interface with Teleport Icon*

In the event of blocked teleportation, the teleport cursor, initially green, changes to red to indicate the problem. Under the original design, the game was to allow Watcher players to “lock in” a teleportation destination, forcing a teleport as soon as the destination and the origin are out of sight. This was not implemented, however; it remains easy to teleport during brief breaks in the Mortal sight-line by tapping the teleport key repeatedly. Watcher players cannot cover vast distances with a single teleport, and they must wait a short period between teleports. This yields a chase dynamic, where Watcher players must opportunistically sneak after Mortal players, trying to outpace them through clever avoidance of obstacles.

So long as they inhabit a statue, Watcher players also have the ability to swap control into another uninhabited statue anywhere in the game. They can do this even if they are currently being watched by a Mortal player. This swap ability is performed by selecting a statue on the mini-map, as seen in Figure 6.

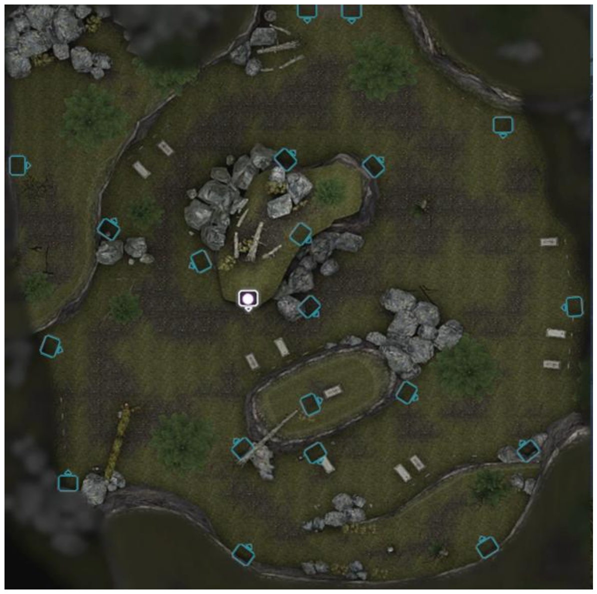


*Figure 6 - Watcher Mini-map*

The Watcher player is shown as a statue with a white ball in it, while statues that are inhabited by other Watcher players (and therefore are un-selectable) are represented with purple balls in them. The mini-map also shows statue orientation and the position and orientation of Mortal players (who are represented via blue arrows).

In the original design, there were additional classes of statues, and Watcher players could not swap into statues of the additional classes without first accumulating sufficient Fear Points. In the actual implementation, there is only one standard class of statues, so this feature was not implemented.

Watchers are able to attack Mortals by projecting their ghost form; when the ghost passes through a Mortal, they are converted to the Watcher team. Figure 7 outlines the state-logic that the ghost form attack uses.

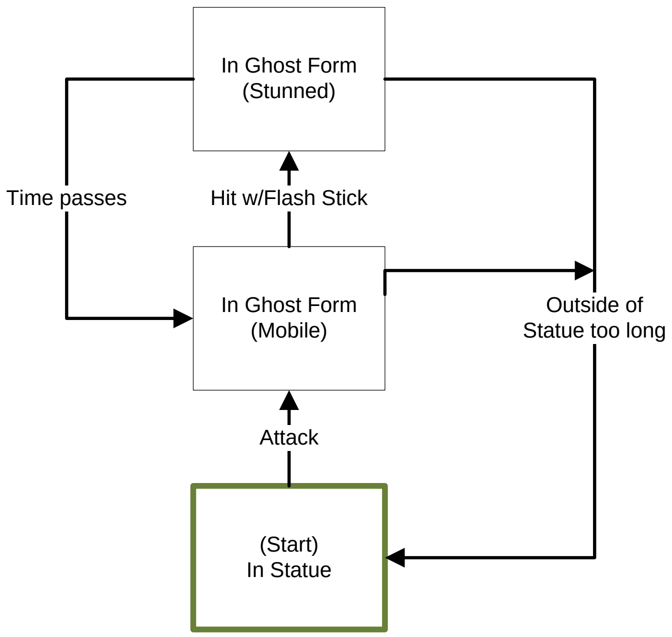


*Figure 7 – Watcher Attack States*

Ghost-form Watchers are vulnerable to flash-stick attacks, and can only last for a limited time (and, by extension, range) outside of their statue shells. Watchers can attack when they are in view of Mortals, but they are in greater risk of being stunned by a flash-stick attack if they do so. When stunned, Watcher ghosts are temporarily immobilized, but time still passes. Whether stunned or mobile, Watcher ghosts will eventually be called back to their statues after a short period (one second for normal attacks, three seconds for lunge attacks).

Like the Mortals, the Watchers have a dedicated voice chat channel to enable them to coordinate and strategize. The original design called for Watcher players to be able to hear Mortals talking, with the volume dependant on the player's distance, but this proved to be difficult to implement technically. Unlike Mortal players, Watcher players are notified when Mortal players are converted, and they have full access to the listing of which players are on which team; this allows them to strategize based on the known habits and skills of the surviving Mortal players.

### 2.2.3 Fear Point System

Mortal players holding in formations can effectively ward off basic attacks, and in general stalemated rounds are not fun for either side. The fear-point system helps mitigate this problem by providing Watchers locked in stalemated games with a long-range and hard-to-avoid secondary attack.

Each Watcher player has an individual fear-point counter, similar to the one shown in Figure 8.

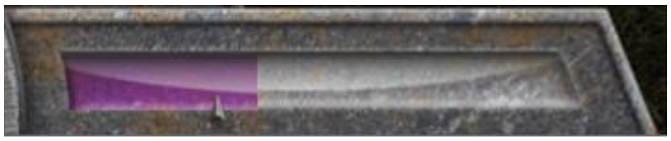


*Figure 8 – Watcher Fear Point Bar*

This bar slowly fills up with a purple fluid when the game has gone for more than 15 seconds without a Mortal death. When the fluid passes the arrow-shaped marker, roughly one third of the way along the bar, the Watcher player gains the ability to unleash a special long-range attack by right-clicking.

Invoking this long-range attack, known as a 'lunge' attack, temporarily transfers the Watcher player to a first-person perspective, as shown in Figure 9.

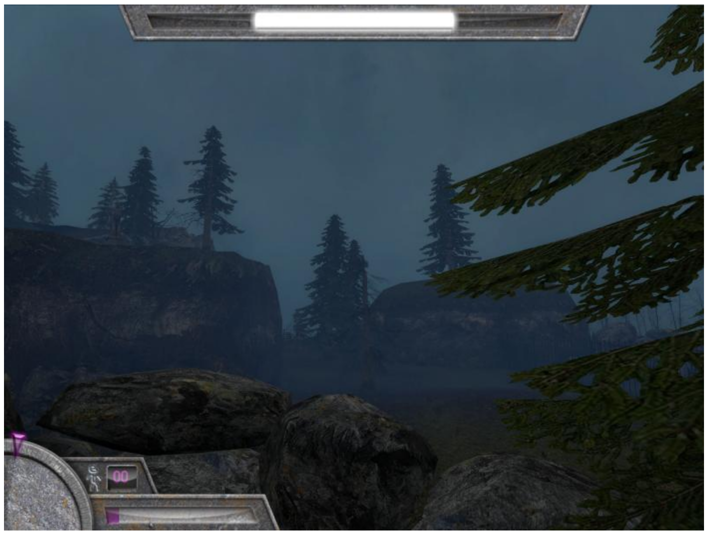


*Figure 9 – Screenshot – Watcher Lunge Attack*

In this state, the Watcher ghost moves forward rapidly, killing and converting any Mortals that it strikes in the process. Unlike the basic attack, the player can control this attack directly by moving the mouse, and he or she can end the attack by right-clicking a second time.

This last ability is important, because the player is under a strict time limit, represented by a shrinking bar shown on the top of Figure 9. If the player runs out of time, the player will be penalized all of his or her remaining fear points; if the attack is ended in time, the player will lose only the third of the fear bar required to engage in the attack to begin with.

The lunge attack was adapted from a more complex system of additional statue classes and attacks specified in the original design. These additional classes were left unimplemented, as creating the necessary artistic assets and implementing the behaviors in code would have delayed start of testing and evaluation.

### 2.2.4 Controls

The control scheme for Watchers inhabiting statues is very different from the standard Mortal control scheme. The default controls for the Watchers are shown in Table 2.

*Table 2 – Default Watcher Control Scheme*

| **Input (key or action)** | **Behavior** |
|---|---|
| | **Teleport** |
| Move mouse | Move teleport destination target within field of view, pan camera left/right |
| Press [q] or middle-mouse button (scroll wheel) | Teleport if: not in sight of Mortal, not moving to spot seen by mortal, not cooling down from previous teleport, and not teleporting out-of-range. |
| | **Swap/View Mini-Map** |
| Hold [c] | Opens the map while [c] is held. |
| Left-click over statue on map | Swap Watcher into selected statue (if statue is unoccupied) |
| | **Basic Attack** |
| Left-click as statue | Initiate ghost-form attack if possible; switches to first-person view. This applies to all statue classes. The attack movement is automatic for the basic attack. |
| Move mouse | Perspective follows mouse (enables turning, looking up/down). |
| Pass through Mortal | Petrify the Mortal, converting him or her. |
| | **Lunge Attack** |
| Right-click as statue | Engage lunge attack if there are sufficient fear points (switches to first person) |
| Right-click as ghost | Return to statue and third-person |

| | |
|---|---|
| | perspective before time limit ends, to preserve remaining fear points (happens automatically when time runs out, but the player loses all remaining fear points) |
| Move mouse | Change direction that the attack is moving in (allows turning, etc.) |
| Pass through Mortal | Petrify the Mortal, converting him or her. |

Like Mortal players, Watchers can re-configure their controls via the options menu (accessible at all times by hitting [Esc] and selecting the "Options" menu item).

#### 2.2.5 Projected Player Strategies

Most Watcher players will initially move in close to Mortal players and attack when their backs are turned. This basic maneuvering will always work to a degree, depending on player skill, but it may not ultimately be the most effective strategy. Still, careful teleportation will be the key to Watcher strategy; the short-range basic attack makes it necessary to be in close proximity to Mortal players to successfully attack them.

More advanced players will make great use of the swapping mechanism, trying to position themselves within statues that will have better opportunities to strike. Waiting patiently is the key to this strategy. If a Watcher can lull the Mortals into thinking that there is nothing in the area, it will have an easier time attacking them. The major issue here is that passively waiting for opportunities wastes time and any time wasted brings the night closer to dawn and to a Mortal victory.

The statues positioned throughout the map can also block a Mortal player's movement. Fast-acting Watcher players, especially those with good teamwork, will be able to block off potential escape routes by lining up extra statues between existing physical barriers.

Many Watcher players may also bide their time in an effort to boost their fear points and prepare for multiple lunge attacks. This can work if all Watcher players agree to avoid attacking, but even one Mortal death will cause all Watcher players to stop accumulating fear points.

## 2.3 Balancing Issues

There are several behaviors and mechanics in this design that could only be properly balanced through playtesting:

- **Distribution of statues –** Too many statues makes it difficult for Mortals to navigate, and it can also make it difficult for Watchers to use the minimap mechanism. Initial tests with the final map had too few statues; Mortals found areas that were not easily accessible by the existing Watcher statues and hid in them. This was rectified by placing additional statues near or on the hiding places. There are 21 statues placed throughout the final map.
- **Time until dawn –** This factor is itself dependent on the number of players involved in a given match. For small matches of three to four players, it appears that ninety-second-long rounds provide an even balance of Mortal and Watcher wins. For medium-sized matches with five to eight players, two-minute-long rounds provide an even balance. For larger matches, the time until dawn could be increased, but all rounds tend to end quickly once the Watcher team reaches a certain critical mass.
- **Flash-stick cool-down –** Most Mortal players found alternate uses for the flash-stick (they used it as a pointing device and locator more frequently than as a defensive weapon). Because of this, the cool-down on the flash-stick attack did not have a major bearing on game balance.
- **Fear point gain/loss rates –** The fear point system works optimally when Watcher players gain approximately a full fear bar after a minute of no Mortal deaths. If they gain points faster than this, Watcher players tend to abuse the lunge attack, and the strategy of positioning statues for successful basic attacks is lost. This degrades the game for both Mortal and Watcher players.
- **Lunge attack time limit –** This limit was initially set to five seconds; it was shortened to three to somewhat limit the attack's range and require that Watcher players use reflexes and judgment to avoid being penalized the remainder of their fear points.
- **Mortal movement speeds –** The default player movement speed in the Source Engine was set to a value of 450 (no units available). This speed did not fit well with the slower-paced atmosphere of *Petrified*; it was reduced first to a value of 250, and is presently set at 225, exactly half of the original speed.
- **Watcher teleport range** – The team experimented with several possible ranges for Watcher teleportation. At distances of roughly half the width of the map, Watcher players never needed to chase after Mortal players; they could simply teleport ahead of the Mortal's trajectory and attack. At distances of roughly a quarter of the width of the map, Watchers could just keep up with Mortals, which led to engaging chase sequences. For values less than this, Watcher players gave up on chasing, instead preferring to lie in wait in convenient

locations.  The team chose to keep the quarter-width range, in order to promote the chasing behavior.

## 3 Artistic Design and Vision

*Petrified*'s gameplay is complimented by the spooky, dark atmosphere of its art and level design. The game takes place inside a small, isolated graveyard in a temperate forested area. The style borrows from several forms of artistic media, such as horror films and games, in which the outdoor scenes are designed to inspire fear and unease.

### 3.1 Level Design

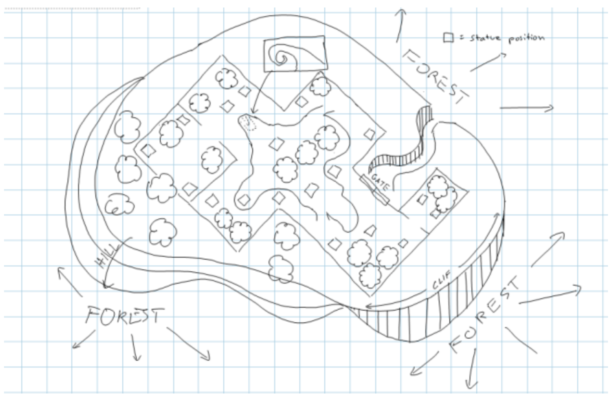


*Figure 10 – Concept Drawing – Early Level Layout*

*Petrified* was built around a single level, or "map." The team took this approach to ensure maximum visual quality and eliminate the issue of balancing for multiple map configurations. The final map is in some ways based on original design concepts, shown in Figure 10. The map contains a mixture of high and low terrain, to give Watchers sheltered locations that they can maneuver in, and is bounded by a combination of natural obstacles (cliff walls, rocks, trees) and man-made barriers (fences and locked gates). The original design called for the engine to place statues randomly at the start of each match, but it proved more atmospherically effective to place the statues by hand, and simply give Watcher players control of a random statue at the beginning of each round.

Figure 11 shows a wireframe view of the final map as it appears in *Hammer*, Valve's level editor.

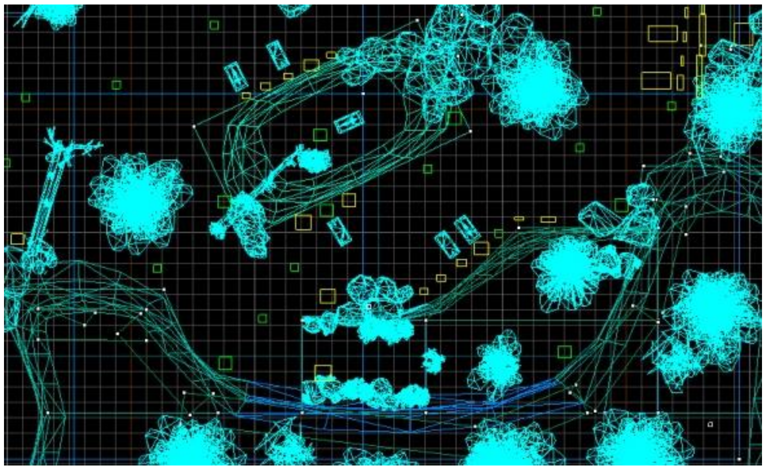


*Figure 11 – Screenshot – Final Map in Level Editor*

The final map differs greatly from the concept in layout and features.  Where the original map was pictured as set on top of a hill surrounded by forest, the final map is set as though in a depression, bounded largely by cliffs and boulders.  Most areas in the concept would have been accessible by a Mortal simply walking around; here, Mortals must jump and climb to reach some areas, and even then cannot approach some of the higher points on the map.

The final map also makes use of a visual technique favored by Valve Software in their game series *Half-Life*.  The map centers on a looming butte that is visible from almost any position within the map.  The screenshot in Figure 12 is taken from a spot perched right on the top of this butte, next to an all-seeing statue.

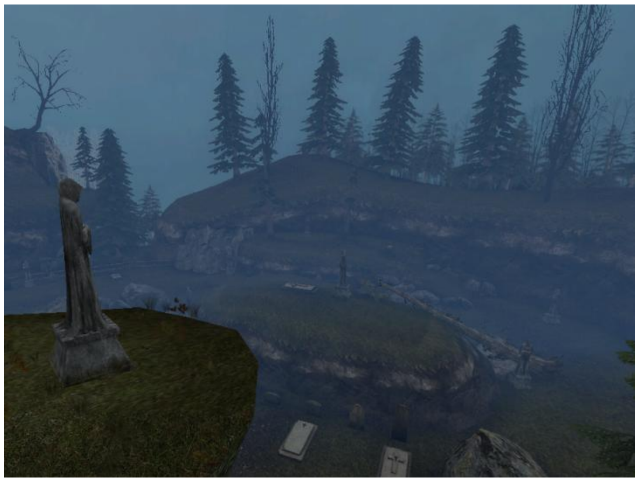


*Figure 12 – Screenshot – Vantage on Final Map*

Players can use the butte as a visual reference for where they are and where they are headed; it can be particularly useful for Watcher players, who must constantly re-orient themselves following swaps into new statues.

In addition to its role as a landmark, the central butte (and other raised areas, such as the mound visible in the center of Figure 12) serves to divide the map into corridors. These corridors provide the easiest paths across the terrain, but they also serve as chokepoints. Watchers can teleport their statues into the corridors, slowing or even completely blocking Mortal progress. If they choose to do so, however, they risk losing the ability to effectively cover the more open areas, where Mortals can easily block teleportation by looking around.

Many of the static models visible in Figure 12 are actually borrowed assets from *Half-Life 2: Episode 2* and the earlier *Half-Life 2*. Both of these games contain numerous outdoor scenes which required many samples of vegetation and rocks, and *Half-Life 2* contained one scene set in a large graveyard, complete with tombstones and statues. Because these needs were already met, the team focused on modeling the terrain of the level itself and assets such as the iron fence components shown in Figure 13 , which the aforementioned games did not provide.

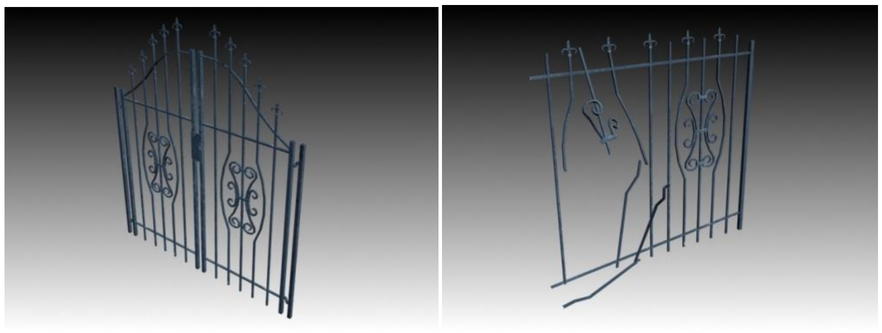


*Figure 13 – Render – Gate and Fence Models*

The map is also populated with certain visual and aural effects. Past a certain distance from the player's perspective, the renderer applies a layer of fog to the terrain, adding to the atmosphere. This fog does not fully obscure any map features, but it adds an element of mystery to events happening in the distance. Some areas of the map, including the north-most corridor, are also tagged with ambient audio effects: players walking through the corridor will hear wind whistling between the cliff faces.

## 3.2 Art Development

*Petrified's* art assets underwent three or four distinct phases:

- **Reference Images** – Pictures taken either in person by team members or gathered from Creative Commons-licensed works available on image sites like Flickr.
- **Concept Art** – Initial sketches and mock-up renders to lay out character poses and features
- **Draft Art** – In-development art, lacking textures, animations, or other important features
- **Final Art** – Fully realized artistic assets ready for use in-game

Not every asset made it through every phase – some assets were cut at the concept art or draft art stages, and not every finished asset was based directly or indirectly on the original reference photos.

### 3.2.1 Watcher Statues

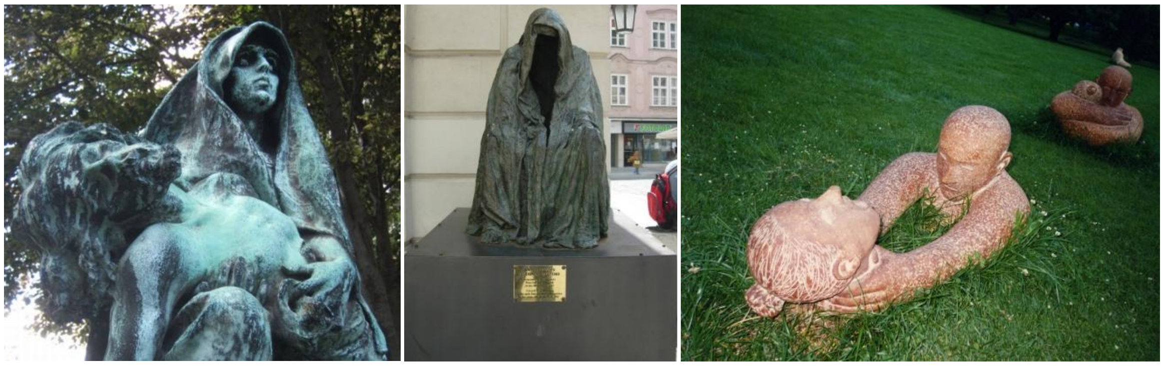


*Figure 14 – Statue Reference Photographs*

For the Watcher statues, the team looked for reference photographs depicting hooded religious figure statues, such as the piece in Figure 14 (far left) showing a shrouded Mary Magdalene cradling the body of Christ. The team also looked for depictions of gargoyles and grotesque statues. At this point in development, the team was still looking to create several classes of Watcher statues, and so the reference photographs were translated into a variety of drawn statue concepts for four different types of statues (Figure 15).

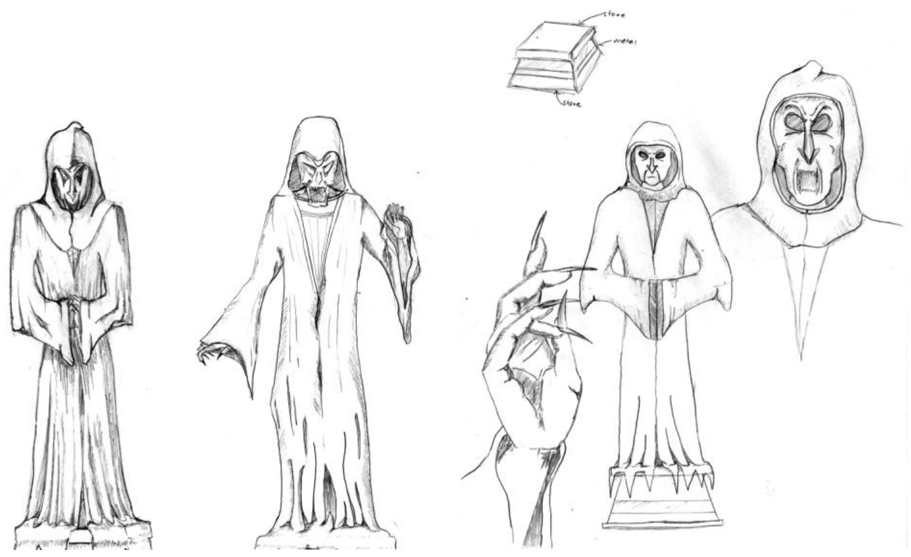


*Figure 15 – Concept Drawing – Basic Statue Concepts*

The basic statue went through a number of design revisions during the concept art stage. The statue was initially depicted as leering and demonic, as shown on the left of Figure 15, but gradually transitioned to a stern but relatively human character (on right).

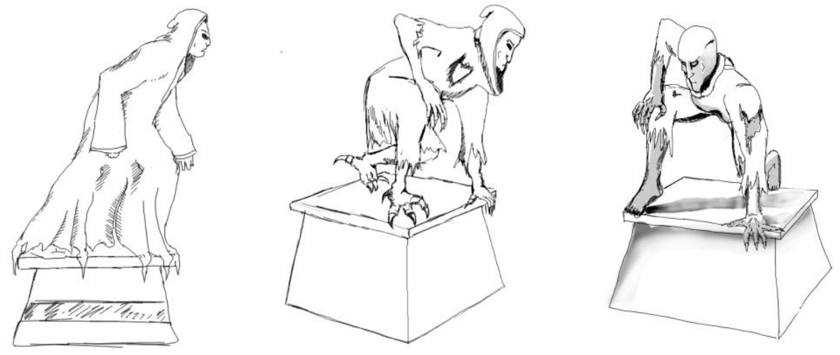


*Figure 16 – Concept Drawing – 'Lunger' Statue Concepts*

The Lunger statue gained a more dynamic pose as it developed (Figure 16), meant to imply a runner preparing for a sprint. This posing was facilitated by 3D modeling software, which was used to create a rough render of a generic model in the desired pose; these models were not used in-game, however. The statue was meant to provide a fast-moving alternative attack to Watchers who had accrued enough fear points.

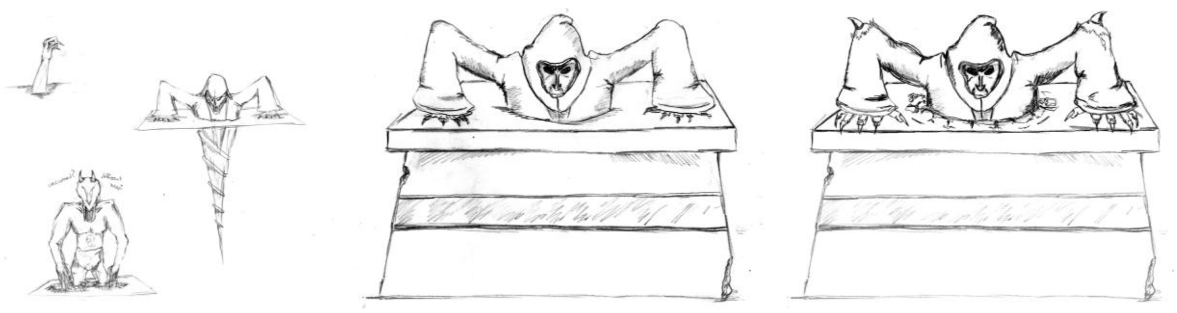


*Figure 17 – Concept Drawing – 'Digger' Statue Concepts*

The Digger statue was depicted as rising out of the ground or its pedestal (Figure 17). This statue was meant to provide a special attack allowing Watchers to dive underground and surface near hard-to-reach Mortals.

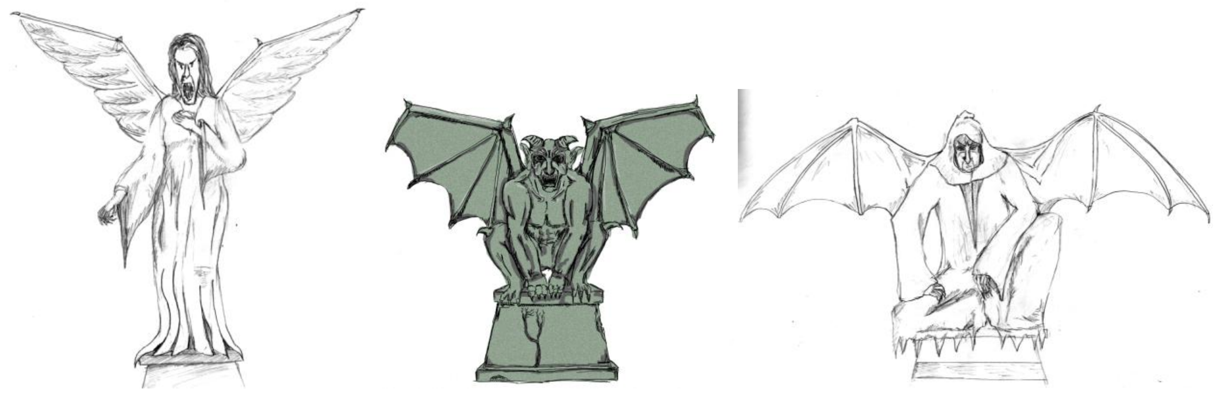


*Figure 18 – Concept Drawing – 'Angel' Statue Concepts*

As implied by the wings, the Angel statues were to grant Watchers the ability to launch a flying attack. Early designs (Figure 18) here were split between winged gargoyles and angels; the final design melded the two concepts, incorporating the bat-like wings of the gargoyles, the body of the angelic figures, and a unique crouching pose.

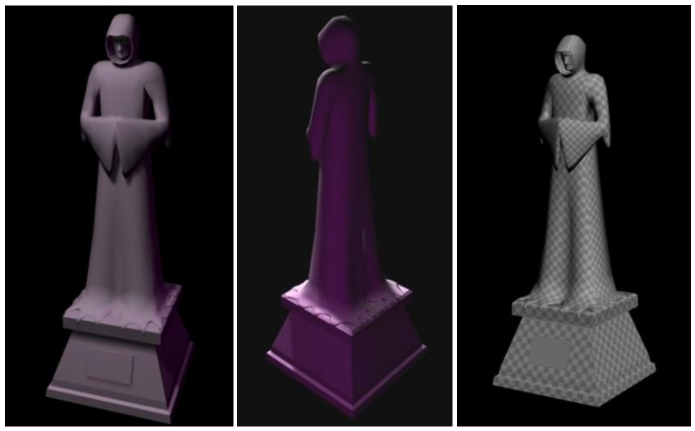


*Figure 19 – Render – Draft Statue Model*

The statue model was based on the final concept drawings: a simple standing figure, hands clasped and head bowed as though in prayer. An early version is shown in Figure 19, and the final textured version in Figure 20.

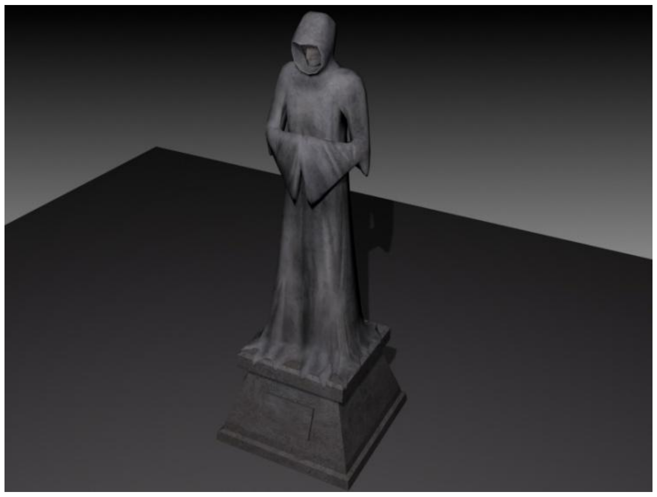


*Figure 20 – Render – Textured Watcher Statue*

The final model used for the Watcher-inhabitable statues placed throughout the map is rendered in a relatively realistic and worn style. It appears to be still yet ominous and brooding. The statues themselves do not animate, although the original design considered the possibility of having the statues subtly change pose as they neared their Mortal prey. One change to the final model – not visible here, because of the ground placed beneath the statue – is an extension of the pedestal down into the ground. This made it so that the Watcher statue never appears to be floating, supported on top of sloped terrain, but instead appears to be embedded in the ground (Figure 21).

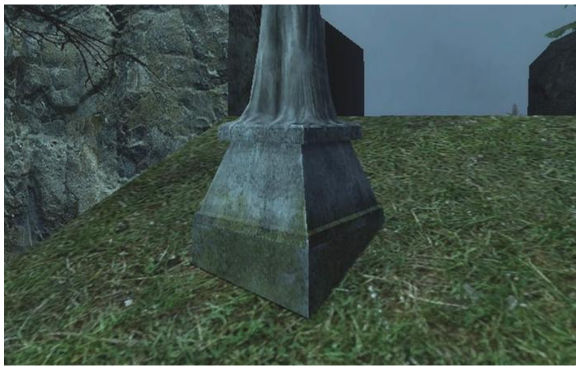


*Figure 21 – Statue Pedestal Extension*

### 3.2.2 Watcher Ghosts

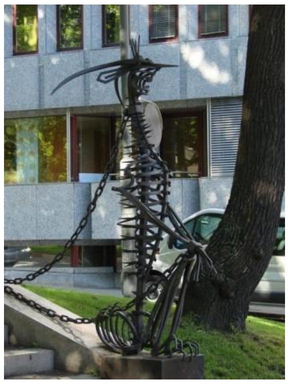


*Figure 22 – "Thin Pieces" Statue (Reference Photograph)*

While the Watcher ghosts are not based directly on any real-world photograph – they are supernatural beings, after all – elements of their design do stem from some of the statues used for general reference. The skeletal gauntness and sharpness of the "Thin Pieces" statue in Figure 22 was a direct influence on the eventual skeleton-based design used for the ghosts (Figure 23).

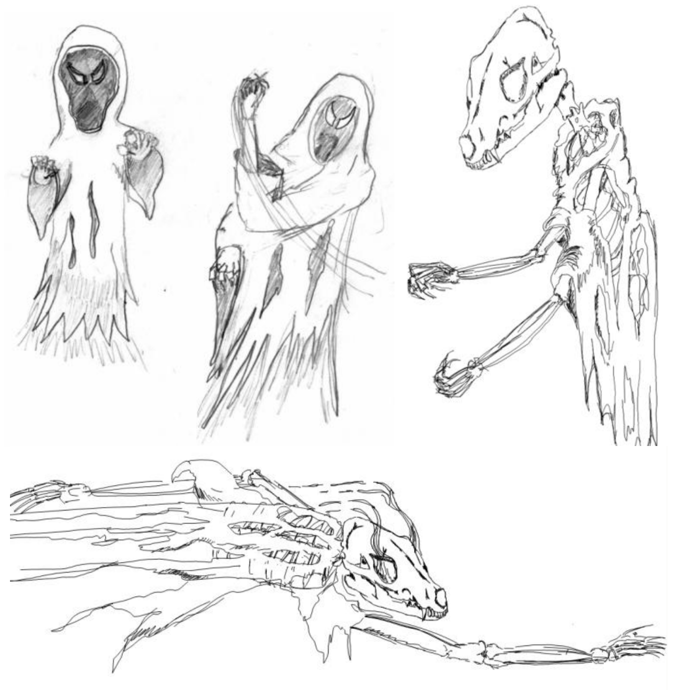


*Figure 23 – Concept Drawing – Watcher Ghost Concepts*

The earliest ghost concepts looked similar to the shrouded statues that they hid inside. This rendition was rejected in favor of a more grotesque and frightening concept based on the juxtaposition of a canine skull with a human skeleton. With the original design, players might have been confused as to whether they were controlling the statue itself or the spirit residing within the statue; the radical visual difference between the two helped make it clearer that the ghost was possessing and acting through the normally benign statues.

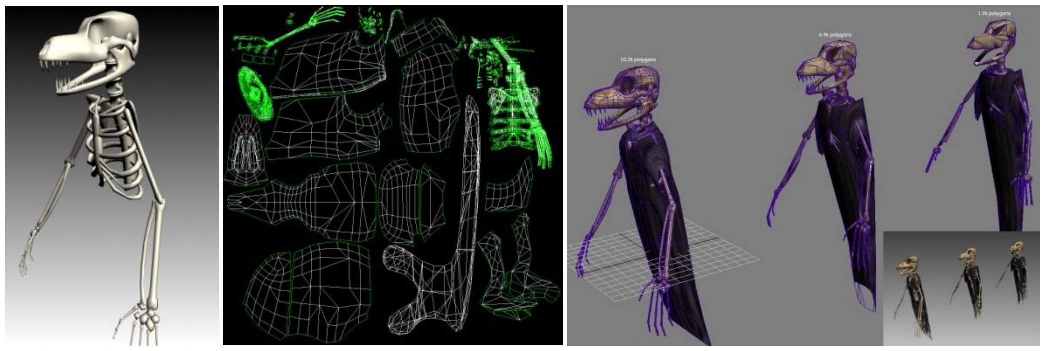


*Figure 24 – Render – Ghost Draft Model, Unwrapping, and LOD Models*

The ghost model was a large time investment, as it needed animations and a complex texture map to cover the detailed bone structure. As shown on the right side of Figure 24, the team also created model variants with fewer polygons, in an attempt to avoid graphics performance issues when rendering multiple ghosts in-game.

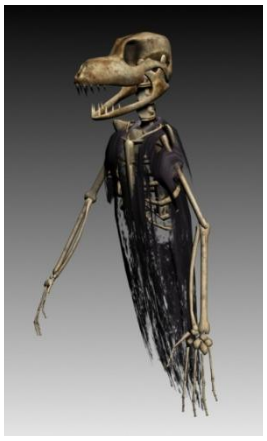


*Figure 25 – Render – Watcher Ghost Form*

In the final design (Figure 25), the conservative-looking Watcher statues contrast greatly with the ghost forms that they hide. These creatures are a composite of a dog's skull and an elongated human skeleton torso. They are draped in tattered purple rags, in keeping with an overarching color theme associating the Watcher team with the color purple. Unlike the statues, the ghosts have two separate, violent animations: one diagonal slash motion, triggered through the basic Watcher attack, and one Superman-esque flying animation, triggered during their lunge attack.

### 3.2.3 Mortals

The Mortal player assets never had any direct photographic references; they were simply modeled after college-age adults. They were visually distinguished by their flash-sticks (effectively heavy-duty flashlights) and their rugged clothing (Figure 26).

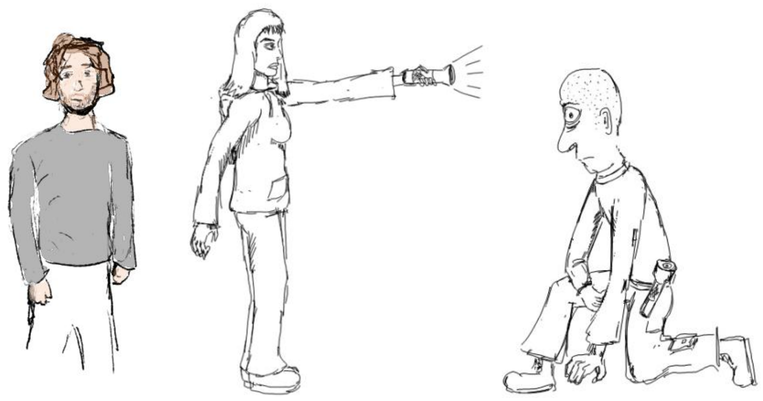


*Figure 26 – Concept Drawing – Mortal Concepts*

Unfortunately, due to technical and artistic difficulties, the planned Mortal models did not make the final release of *Petrified*. The Mortals were planned to be normal humans, who were to appear scared and vulnerable. They were pictured as roughly college-age adults, outfitted for late-fall outdoor exploration. The particular version of the Source SDK used to create *Petrified* makes use of a character model format that is not well documented and differs greatly from previous arrangements used in *Half-Life 2.* Because of this, the team was forced to retain the basic "terrorist"-styled model provided by default, so as to give the Mortal players a functioning visual representation.

## 3.3 Heads-Up Display Design

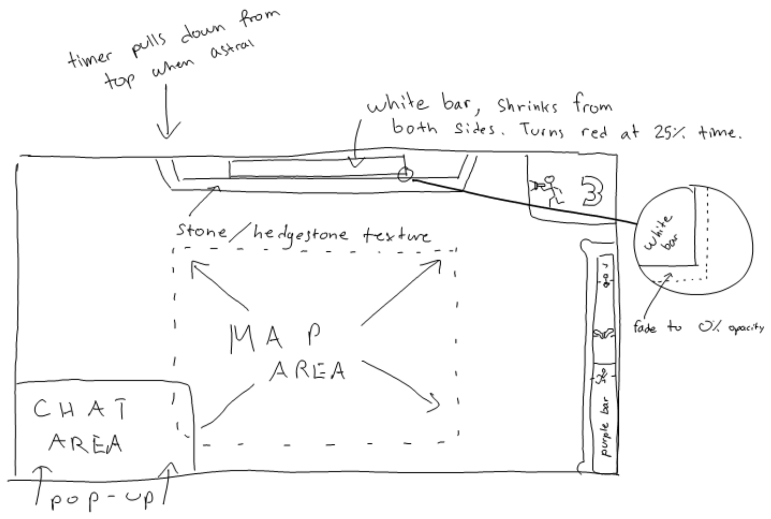

timer pulls down from top when astral
white bar, shrinks from both sides. Turns red at 25% time.
3
stone/hedgestone texture
white bar
fade to 0% opacity
MAP AREA
purple bar
CHAT AREA
pop-up

*Figure 27 – Concept Drawing – Watcher HUD Concept*

The Watcher players in *Petrified* can learn a great deal of information about the state of the game from the user interface elements that compose their heads-up display, or HUD. The original HUD concept shown in Figure 27 contains most of the informational content of the final implemented HUD, although it is less compact and lacks the timer element that was added to replace the visual effect of night turning to dawn.

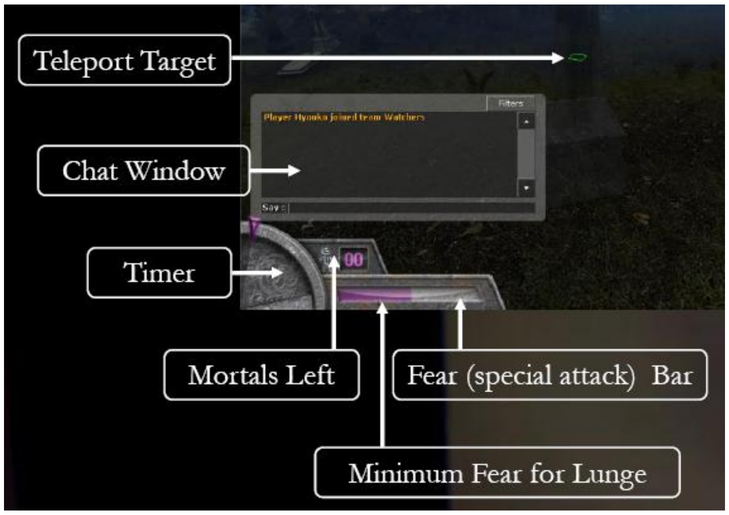


*Figure 28 – Manual Excerpt – Watcher HUD Explanation*

The final interface, shown in Figure 28, provides the number of remaining Mortal players, a timer showing the time left in the round, and a gauge tracking the player's accumulated fear points. Players can additionally bring up a chat window and a mini-map, allowing them to communicate and observe player movements, respectively. The mini-map is also used to enable selecting and swapping player control into uninhabited statues. There are two additional Watcher HUD elements that are only visible at key moments: a timer bar denoting the time remaining in the lunge attack, and a "kills" message that shows when a Watcher player successfully attacks a Mortal.

Mortals are not privy to any of this information; they have the ability to chat with each other, but do not see the number of teammates remaining or the time left until dawn (though this last option can be toggled on per the server administrator's preference). This helps increase Mortal player immersion and fear.

## 3.4 Artistic Tools

*Petrified*'s map was created using a combination of the Source SDK's *Hammer* editor and *Autodesk's 3D Studio Max* (3DSMax). Hammer was used to arrange larger segments of the level and integrate entities and objects (spawn points, etc.), while 3DSMax was used to develop the models and complex landscape segments.

The concept and design artwork was created through scanned drawings, *Adobe Photoshop*, and simple renders in 3DSMax.

# 4 Technical Design

As *Petrified* is a game-modification project, or "mod" for short, rather than a project developed from scratch, much of the architecture supporting the game was already in place. The technical challenges came from understanding and extending that architecture to realize the features called for in the gameplay and artistic designs.

## 4.1 Engine Choice Rationale

This project was based on the Source Engine from Valve, Inc.; specifically, it was developed as a total conversion mod based on *Half-Life 2: Episode 2*. The engine was chosen for the following reasons:

- The Source Engine was originally designed with a client-server architecture, and has served as the basis for first-person multiplayer games ranging from *Counterstrike Source* to *Team Fortress 2*. It supports a large number of players – the PC version of *Team Fortress 2*, for instance, supports 24 players by default, though the engine can support up to 32.
- The most recent version of the Source Engine (the one included with the popular *Orange Box* bundle) supports high-quality dynamic lighting that could be applied for most of our lighting needs.
- Source and Hammer, the associated level/map creation tool, both integrate well with common modeling tools like 3DSMax. The SDK also includes model-viewers and other helpful artistic tools.
- Source has supported a variety of particle effects since its inception (including particulate fog used for atmosphere). More recent versions also support true motion-blur effects and heat distortion effects.
- Source supports an adjustable field-of-view, which will be important in maintaining the visual-related gameplay mechanisms (such as the Mortals' ability to freeze Watchers) across configurations with different monitor/resolution settings.
- The Source engine can support large, open outdoor areas, as demonstrated in some scenes from *Half-Life 2: Episode 2,* which utilizes the most recent commercial build of the engine.

## 4.2 Specifications

These specifications are performance and feature targets; if game performance does not reach the level specified here, the visual and technical elements of the game (such as polygon counts or network packet size) will need to be reduced in complexity. The specifications themselves will only be adjusted if they are technologically infeasible or will in some way broadly compromise the gameplay or artistic design.

### 4.2.1 Hardware Requirements

According to Valve's website (steampowered.com), these are the minimum and recommended system hardware requirements for their most recent multiplayer Source Engine-based game, Team Fortress 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

*Petrified* does not utilize visual effects that are as intensive as those found in *Team Fortress 2* (for instance, a soft cartoon-shading pixel shader that is applied to all models), so it can run comfortably on configurations closer to the stated minimum requirements. The game runs smoothly on a laptop with a 2.1GHz Pentium M processor, 1GB of RAM, and a Radeon 9700 video card.

The requirements do not specify what level of Internet connection is necessary, however. Some of *Petrified*'s visual illusions (particularly the illusion that the statues remain stationary for the Mortals) are difficult to maintain over a high-latency connection. Therefore, as an additional requirement, the network connection must be a broadband connection with less than 200ms of latency to the server or a direct connection via local area network.

### 4.2.2 Software Performance Targets

*Petrified* meets the following performance targets:

- Players: 16-player maximum (Tested up to 22 players)

- Video frames per second: Stable 60FPS on recommended-level hardware, 30FPS on minimum, without significantly compromising visual detail (Achieved, with video resolutions of 800 by 600 pixels on minimum-level hardware)

- Latency: Design for less than 100ms ping time, to maintain the illusion that the statues never move. (Achieved consistent 5ms latency over local area network; average internet latency was 30-50ms).

- Voice chat supports every player talking at once, but it is expected that at most 4 will be talking at one time (Achieved through Source SDK's built in voice chat support)

## 4.3 Technical tools

The Source SDK integrates with *Microsoft's Visual Studio* line of Integrated Development Environments (IDEs). As a result, this project was developed with *Microsoft Visual C++ 2005*, as this version has the most support information in the *Valve Developer Community's* Wiki.

The project also utilized a version control system, *Subversion* (SVN), which is available through WPI's http://sourceforge.wpi.edu site. This enabled the two developers to avoid creating conflicting code and also allowed the project to revert the code base to earlier revisions when new updates proved unstable or otherwise undesirable. As *Subversion* does not integrate with *Visual Studio* by itself, the team used an additional tool, *TurtleSVN*, to commit changes and manage the code repository.

For non-code resources, such as concept art, design documents, and website materials, the team has used *Dropbox*. *Dropbox* is a file synchronizing system that allows several users to share a folder with each other over the Internet; it integrates directly with the *Windows* operating system, and it allows for version control through its web interface.

## 5 Project Workflow and Management

*Petrified's* development required all three team members to be working on separate tasks in parallel for smooth progress. With a timeline of only three terms (24 weeks, counting the extra three over the winter break), the team needed to be mindful of goals and deadlines to complete the project in full.

### 5.1 Man-hours over time

For the first two phases of the project, the *Petrified* team recorded each member's hourly work contributions to check progress against the predicted number of hours required to complete desired design and implementation work. Figure 29 shows the graph of hours worked plotted against time; the actual number of hours worked each week closely matched the predicted value.

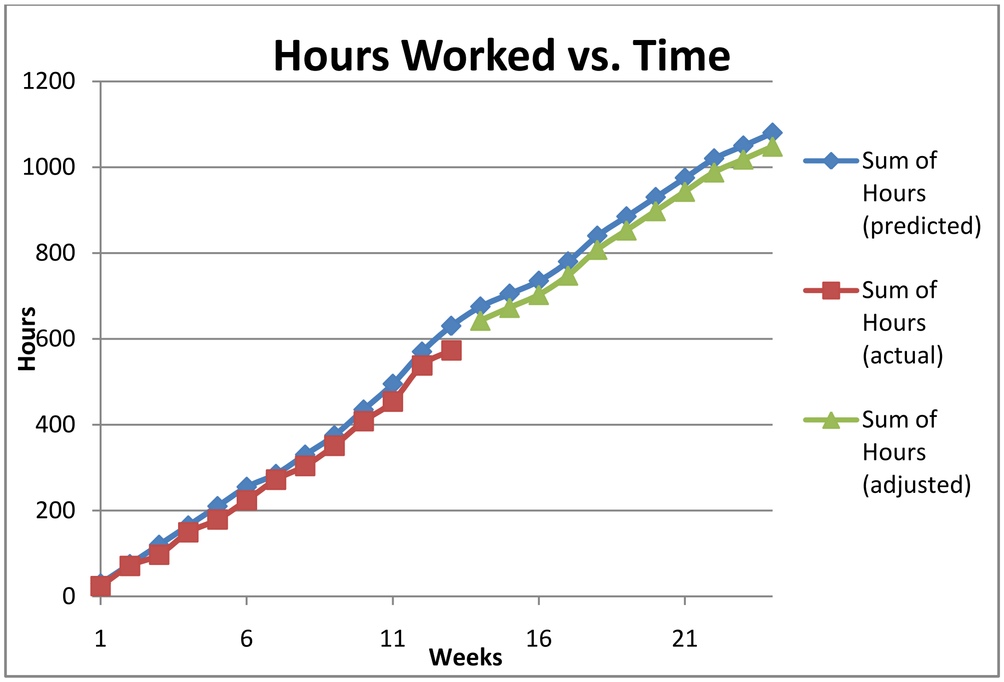


*Figure 29 – Hours Worked vs. Time*

This was an extremely useful tool during the design and implementation phases, where the team was able to predict the number of hours required for each individual task with some accuracy. Unfortunately, the third term was heavily focused on testing and bug fixing, which led to a very erratic workload; it was not possible to predict how long it would take to diagnose and fix a set of bugs, so this method was not used through the evaluation phase.

## 5.2 Task scheduling and assignment

The team used Microsoft Project to schedule tasks and assign them to specific team members. Project allows for tasks to be linked as dependant, so that at any moment in time it was easy to discern which prerequisites must be completed before a task can be started. Tasks were assigned to the team members who possessed relevant skills and were not already overcommitted on other work. The task allotment can be seen in the form of a Gantt chart in Appendix D.

## 5.3 Process

### 5.3.1 Meetings and Communication

Throughout the project, the team met at a minimum of twice a week. Each meeting lasted between one to two hours. The first meeting each week was a meeting with the project advisor. This meeting alternated format every other week between an informal review session and formal status reports. During the review sessions, the following items were covered:

- Quick updates on the development of the game from all three perspectives (technical, artistic, and gameplay)
- Discussion of problems or setbacks encountered
- Changes or clarifications to design proposed
- Unsettled questions raised for mediation/comment
- Proposed work for each member through the upcoming week

The status reports were more formal, both in terms of content and in terms of visual presentation. They were designed to simulate the type of presentations a team would need to make if they were developing in a business environment. The reports typically took the form of PowerPoint presentations, with one team member leading the presentation (chosen on a rotating basis, so that every member led the presentation multiple times). These presentations contained the following:

- Formal introduction
- Summary of changes implemented through the previous two weeks of work, with team members explaining and describing their individual efforts
- Demonstration of newly implemented features and content

- Projection or estimate of remaining work and of the team's position in relation to the original schedule
- Additional time for questions and comments

The other meetings each week were internal team meetings, held to allow the team members to coordinate their efforts and integrate/consolidate their work. Most internal decisions and task assignments were made at these meetings. The team also used this time to lay out new ideas and occasionally to do implementation work, such as "extreme programming"-style pair programming on difficult segments of code.

Between meetings, the team members communicated using a combination of instant messaging, email, and telephone.

### 5.3.2 Report Development

Much like the technical implementation of the game, the development of the project report (of which this document is an early draft) was an iterative process. During the design phase, the report went through three distinct iterations:

- **Project Proposal Review (PPR) –** The initial draft of the report, focused on the broad concepts behind the game.
- **Preliminary Design Review (PDR) –** The second draft of the report, greatly expanding and clarifying matters of gameplay and technical specifications; it also contains asset listings and a selection of early concept art.
- **Critical Design Review (CDR)** – The third draft, revised based on comments from the project advisor, Robert Lindeman, as well as comments received from student reviewers not previously familiar with the project.

Following final comments and revisions to the CDR draft, the design elements of the report (effectively, sections 2, 3, and 4) were frozen, to serve as the design document for the implementation phase of the project. The original design sections as they stood at that point can be seen in Appendix G – Original Design. Some elements were updated to reflect balancing decisions (mostly pertaining to the factors discussed in section 2.3) made during implementation and playtesting, and a few additional elements were changed or dropped to accommodate time restrictions and technical limitations. The design freeze was meant to prevent the project from succumbing to feature creep – that is, the haphazard introduction of additional elements into the design that might have unbalanced the game or delayed delivery of playable evaluation builds.

The design document was based on the original project pitch, visible in Appendix C – Original Project Pitch, which was written in an informal style. It discussed potential gameplay issues and offered multiple potential solutions for them; it was the first in-depth exploration of the gameplay mechanics, themes, and player motivations (the "fun" element). Many, though not all, of the ideas from the pitch document remained a part of the overall design, and the project was guided by the vision laid out in the sentence/paragraph descriptions.

## 5.4 Project Milestones

During the implementation and evaluation phases, the team held bi-weekly status presentations to discuss the state of the project. In addition to these presentations, progress on the final project was measured by the following milestone releases:

- **First Playable** – The first functioning prototype of the game, displaying basic functionality (movement, etc) and placeholder artwork. This release was achieved in the second week of B-Term 2008.

- **Second Playable** – The second functioning prototype, displaying more advanced functionality (Watcher abilities such as swapping and teleporting, basic flash-stick functionality). This release was complete in the fifth week of B-Term 2008.

- **Final Alpha –**The last non-public prototype; near feature complete, with most of the final artwork in place. Released in the second week of C-Term 2009.

- **Initial Beta -** The first public prototype. Feature-complete, with all artwork in place. Released in the third week of C-Term 2009.

- **Public Release Build –**The final public release developed under the auspices of this MQP; feature-complete and extensively debugged and play-tested. Released in March 2009.

These milestones were not the only builds of the game; the team created incremental builds versioned 0.4.0 through 0.9.0, with 0.0.1 increases corresponding to mid-week builds, and 0.1.0 increases at each iteration (versioning started at 0.4.0 because there were no installers compiled for the first 3 iterations). The public release build was versioned 1.0.0.

# 6 Implementation

Following the completion of *Petrified*'s design, as detailed in sections 2, 3, and 4, the team began work on the implementation of that design. The implementation phase focused on creating the artistic assets and developing the game client and server using the *Source SDK*.

## 6.1 Project Workflow and Management

From a software development standpoint, *Petrified* was built up gradually by using iterative development. The development iterations followed a weekly schedule while formal presentations were given to the project's advisor, Professor Rob Lindeman, on a bi-weekly schedule. The team met with Professor Lindeman weekly, and held an internal team meeting weekly.

The team meeting was a period of evaluation and recalculation, to make sure that the project as a whole was on track. During each team meeting, the team reviewed individual task assignments, and made adjustments to the schedule as necessary.

Each week, the team gave a status report so that Professor Lindeman would know what progress had been made, and so that the team could receive guidance. This also gave the team an opportunity to practice giving presentations and status reports, both in a formal and informal capacity.

## 6.2 Changes to Design

During the implementation phase, the team tried to follow the original design, but due to technical and time constrains some changes were necessary. Though some aspects of the game did deviate from the original design, no major new features were added.

### 6.2.1 Changes due to Technical Constraints

One aspect of the design was changed due to technical constraint – the visual effect of transition to dawn during round was replaced with a timer. The dawn transition effect was not gracefully achievable in the Source engine. While some previous *Source*-based games attempted such a transition, the resultant effect was a very short and visually unappealing switch between distinct day/night states. While it would have been possible to create a series of such transitions to make the switch more gradual, the overhead (new lightmaps and skybox textures for each stage) would have greatly increased the resource usage, reducing performance all around.

To replace the dawn effect's informative purpose, as it was the only indicator of remaining round time, the heads-up displays (HUDs) for Watchers was changed to contain timer elements, as shown in Figure 30.

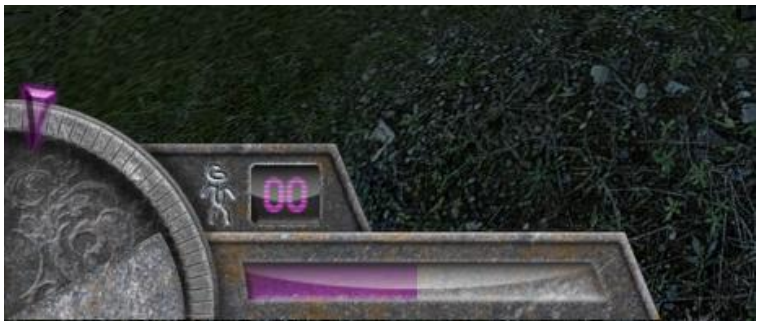


*Figure 30 – Watcher HUD w/Timer*

The timer is represented as a rotating stone dial wherein the lighter stone slowly rotates to cover the darker material beneath as the round approaches its end.

#### 6.2.1.1 Changes due to Time Constraints

Other aspects of the design were removed or reduced into more-easily implementable forms to meet the feature freeze deadline:

- Additional Watcher classes removed; "lunge" attack ability changed to a special attack of the base statue.
- Fear point accumulation is based only on time since last Mortal death.
- Mortals have no visual modifications (flashlight model, new player model, etc), and are based on the default "terrorist" model.

The additional Watcher classes were cut because they represented a significant time investment on both the technical and artistic sides; each new class would require a large amount of programming and testing, as well as new models and poses to visually differentiate them from the base statues. The design still required their functionality as a stalemate-breaking tool, however; in light of this, the team decided to change the long-distance "lunge" attack into a special attack, accessible from any statue given an appropriate balance of fear points.

Fear point accumulation was implemented as simply as possible to begin with. While it would not have taken a great deal of time to implement the other accumulation techniques, early testing showed that the simple implementation was sufficient, and other features were of higher priority.

Finally, the team decided to prioritize level design and statue/Watcher modeling over Mortal improvements. The Watcher design is particularly crucial to the game's atmosphere, and could not be neglected. Similarly, the level design has more direct impact on player experience and exposure to gameplay mechanics.

## 6.3 Artistic Implementation

The implementation of the artistic requirements proved difficult because of the error-ridden pipeline used to get content from *3D Studio Max* or *Photoshop* imported into the *Source* engine. The most problematic part of getting assets in game was getting the collision model to export correctly.

The collision model is an invisible shell that goes over an object. When a player steps on the object, he or she is really colliding with the collision model and not the object itself. All collision models must be convex. For example, if one made a hollow tube as a collision model, *Source* would only see it as a solid cylinder. To get around this, the team tried to make concave collision models out of multiple convex parts. However, the engine still filled in the collision hull and treated it as one solid piece. Since the only props that needed such a collision model were broken iron gate segments, those props were simply not used.

When using an animated collision model, such as that of the ghost, collisions were even more problematic. When exported, the collision model was rotated, translated and did not animate correctly with the rest of the model. These problems with the collision hull were posted on the appropriate forums, but even the author of the *3D Studio Max 2009* SMD converter[2], who refers to himself as 'Wunderboy', was unable to figure out what was wrong. Instead, a simple collision box was used to detect collisions between a ghost with a Mortal, or with the Mortal's flashlight.

Another challenging area in the artistic design of *Petrified* was the creation of convincing rock/cliff structures using the *Source* level editor, *Hammer*. *Hammer* was originally created for designing interior spaces. By looking at how Valve level designers created cliffs in levels from *Half Life 2*, the team developed a method for creating cliffs. Cliffs could be built by approximating the shape of the cliff using cubes, then applying a displacement and subdivision to the cubes, which created more vertices to work with and helped to smooth out the shape. The shape was then tuned using the "Paint Geometry" tool, as this gave more flexibility than manipulating each vertex one at a time.

[2] SMD is the initial file type used in the Source engine's model compiling pipeline.

## 6.4 Technical Implementation

The *Petrified* team evaluated a number of potential technological platforms for the project, as discussed in section 4.1; ultimately, the team settled on *Valve Corporation's Source SDK*. Working within the *Source SDK*, the team was presented with an additional choice between four code bases:

- Half-Life 2 Singleplayer (HL2 SP)
- Half-Life 2 Multiplayer (HL2 MP)
- From Scratch
- Skeleton

The team chose to use the "from scratch" base code, which provided assets and working multiplayer functionality, but was not as complex as the HL2 MP base. The Skeleton base is designed for developers who are very experienced with the *Source* engine, as it provides almost no functionality or artistic assets. Thus, the "from scratch" base was a good compromise between complexity and pre-defined functionality.

### 6.4.1 Source SDK Architecture Overview

The *Visual Studio* solution for the *Source SDK* is divided into Client and Server projects, as seen in Figure 31.

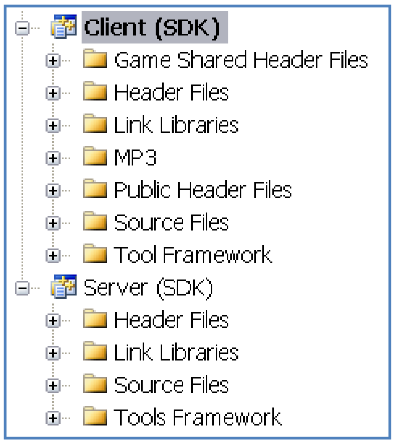


*Figure 31 – Source SDK Solution*

Most classes are represented within both projects, with both client- and server-side implementations. The *Source* engine uses a thin-client architecture, and as such the client primarily deals with rendering the game world, managing the user interface, capturing input to send to the server, and predicting the gamestate. It predicts the local gamestate from the server's most recent updates and from the captured input. The server code handles all of the actual game logic and runs the definitive physics simulation. The physics code is duplicated on both client and server so that the client can do accurate prediction of the effects of the input, and of the movement of other entities.

Some entities and bits of game logic exist only on the client side. For instance, there are some temporary entities, such as the individual sprites that compose a particle effect, that are not managed by the server (since it would require a great deal of bandwidth to do so). Other entities and effects are managed client-side because it would not make sense for all players to see or know about them. In *Petrified*, the Watcher's teleport destination indicator is client-side-only, since it would give away Watcher locations if the indicator was visible to other players.

*Source SDK* modifications do not directly alter the *Source* engine in any way. The Client and Server projects compile to DLL files that are then loaded by the engine executable. Valve has exposed a great deal of the engine's functionality to the *Source SDK*, but is not possible to alter the underlying engine behavior directly, such as the rendering system.

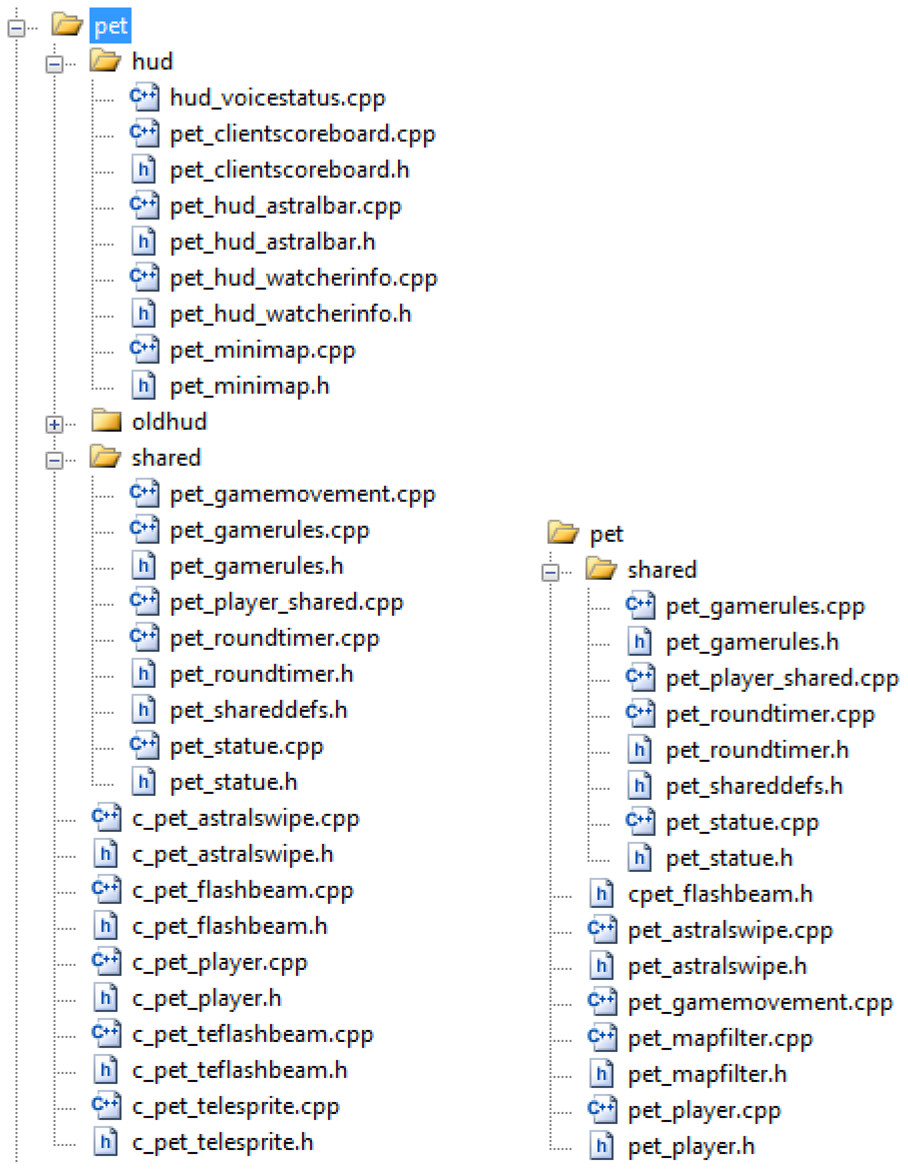
Unfortunately, the *Source* codebase is not particularly organized beyond the Client/Server distinction; almost every file for both projects is lumped directly into the "Source Files" folder. Though there is no universally consistent naming scheme, it is possible to identify files belonging to certain modules by their prefixes:

- `vgui_` files relate to *Valve's* user interface system (vgui)
- `hud_` heads-up display entities, most are based on the vgui system
- `c_` files typically denote a client-side entity
- `c` (without an underscore) files denote a server-side entity, often both a client and server-side version of this entity exist.
- `te_` files are all temporary entities, typically used for special effects or rubble that do not need to be modeled directly.

A few of the files are clearly commented, while most are not. *Valve's* online *Source SDK* documentation contains extra explanations and details for some of the classes and functions available through the *SDK*. Most importantly, the *SDK* Docs provide tutorials that point out the various entry points in the code, such as the game rules entities, which will eventually control the game and differentiate it from the codebase's default behavior.

### 6.4.2 *Petrified* Extensions

While programming *Petrified*, the team tried to follow the encapsulation principle of the Object-Oriented paradigm, and in doing so separated *Petrified's* code, seen in Figure 32, from the *Source SDK* code wherever possible.



```
pet
  hud
    hud_voicestatus.cpp
    pet_clientscoreboard.cpp
    pet_clientscoreboard.h
    pet_hud_astralbar.cpp
    pet_hud_astralbar.h
    pet_hud_watcherinfo.cpp
    pet_hud_watcherinfo.h
    pet_minimap.cpp
    pet_minimap.h
  oldhud
  shared
    pet_gamemovement.cpp
    pet_gamerules.cpp
    pet_gamerules.h
    pet_player_shared.cpp
    pet_roundtimer.cpp
    pet_roundtimer.h
    pet_shareddefs.h
    pet_statue.cpp
    pet_statue.h
  c_pet_astralswipe.cpp
  c_pet_astralswipe.h
  c_pet_flashbeam.cpp
  c_pet_flashbeam.h
  c_pet_player.cpp
  c_pet_player.h
  c_pet_teflashbeam.cpp
  c_pet_teflashbeam.h
  c_pet_telesprite.cpp
  c_pet_telesprite.h
```

```
pet
  shared
    pet_gamerules.cpp
    pet_gamerules.h
    pet_player_shared.cpp
    pet_roundtimer.cpp
    pet_roundtimer.h
    pet_shareddefs.h
    pet_statue.cpp
    pet_statue.h
  cpet_flashbeam.h
  pet_astralswipe.cpp
  pet_astralswipe.h
  pet_gamemovement.cpp
  pet_mapfilter.cpp
  pet_mapfilter.h
  pet_player.cpp
  pet_player.h
```

*Figure 32 – Petrified Code*

The *Source SDK* codebase is enormous, and it would have become impractical for the team's programmers to keep track of changes amongst the SDK's hundred of classes and components. This more modular approach keeps all of the major additions and extensions to the basic "from-scratch" *Source SDK* mod in one convenient and organized location.

Almost all of the *Petrified* classes extend and override existing *Source SDK* classes. There are two major components, however:

- The game rules' entity, shared on client and server as `pet_gamerules`, manages all of the game logic; it checks for win-conditions, manages rounds, intermissions, newly joined players, and generally perform clean-up duties to make sure the game continues to function.
- The player entities – defined across `c_pet_player` client-side, `pet_player` server-side, and `pet_player_shared` for components needed on both – handle player controls, appearances, and behavior for both teams.

Of the other classes, only the `pet_statue` class is a persistent, visible entity. It defines the statue entities that Watchers can possess and control. The remainder is divided between HUD components, such as `pet_minimap` and `pet_hud_watcherinfo`, and temporary entities like `pet_astralswipe` that model attacks and special effects in-game.

Not all of the changes to the *Source SDK* codebase were made through these extended classes. Some, such as various user control additions and key mappings, were worked into existing files like the `in_main` component used to process all key input. These changes were marked with starred comments, such as "`//*chris- Launches a swipe attack`," so that they could easily be identified with a simple search through the code. This method sufficed for the present scope of the *Petrified* project, since the number of these non-modular additions was kept to a minimum.

# 7 Evaluation

To evaluate *Petrified*, it was first necessary to define what constitutes a success. The team laid out several broad goals for *Petrified* from the start:

- **Balance** – Both teams should win on an approximately even basis; there should be no major exploitable behaviors that lead to a guaranteed win for one side or the other.
- **Fun** – Playing *Petrified* should be an enjoyable and compelling experience.
- **Immersion** – *Petrified* should communicate a feeling of paranoia and fear to Mortal players, while simultaneously empowering the Watcher players.

Mathematical approximation[3] and quality assurance testing[4] can be used to tune game balance. Fun and immersion, however, are both very subjective factors. By exposing *Petrified* to an external, unbiased audience the team could receive useful feedback, in the form of personal anecdotes and experiences. In the aggregate, these experiences yield a picture of the game's successes and failings in achieving its goals.

A further, more objective point of evaluation is simply the amount of exposure and popularity *Petrified* receives.

## 7.1 Methodology

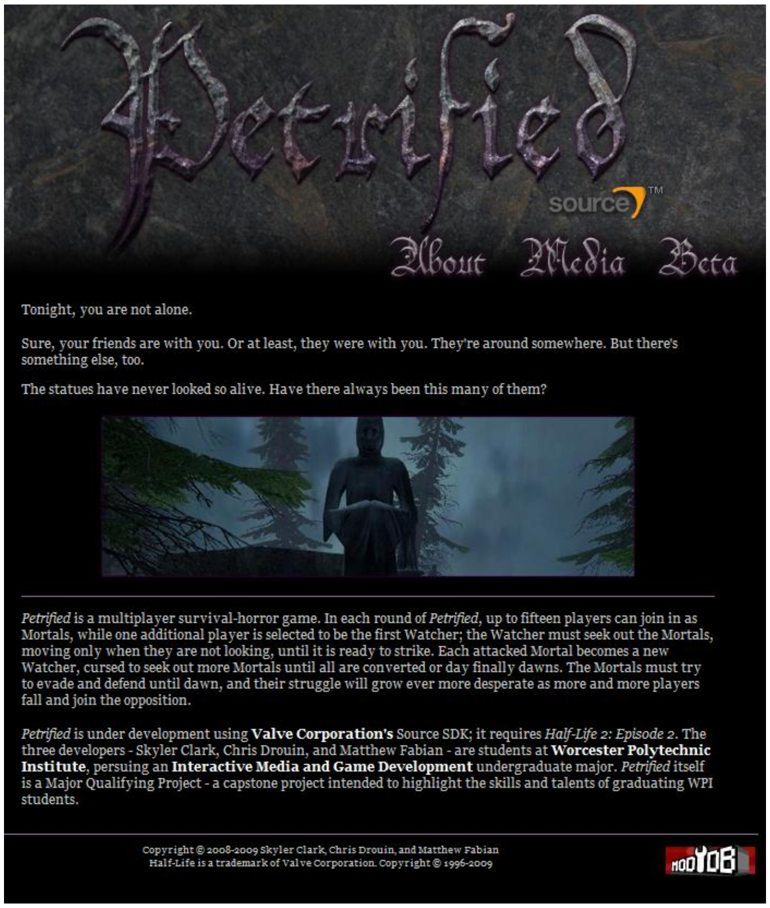
Given the need for a varied audience, the *Petrified* team set out to promote *Petrified* and make it available to play, while still maintaining the control over the game necessary to monitor its use and gather feedback.

### 7.1.1 Promotion

The team used a variety of channels to promote *Petrified* to potential testers. Much of this promotion was focused on increasing the online visibility of the game.

[3] For instance, determining an attack's optimal length based on point-of-intersection with a moving opponent.

[4] Quality assurance testing is the process of manually checking game mechanics under an exhaustive set of preconditions to find bugs or exploitable behavior.



Tonight, you are not alone.

Sure, your friends are with you. Or at least, they were with you. They're around somewhere. But there's something else, too.

The statues have never looked so alive. Have there always been this many of them?

*Petrified* is a multiplayer survival-horror game. In each round of *Petrified*, up to fifteen players can join in as Mortals, while one additional player is selected to be the first Watcher; the Watcher must seek out the Mortals, moving only when they are not looking, until it is ready to strike. Each attacked Mortal becomes a new Watcher, cursed to seek out more Mortals until all are converted or day finally dawns. The Mortals must try to evade and defend until dawn, and their struggle will grow ever more desperate as more and more players fall and join the opposition.

*Petrified* is under development using **Valve Corporation's** Source SDK; it requires *Half-Life 2: Episode 2*. The three developers - Skyler Clark, Chris Drouin, and Matthew Fabian - are students at **Worcester Polytechnic Institute**, persuing an **Interactive Media and Game Development** undergraduate major. *Petrified* itself is a Major Qualifying Project - a capstone project intended to highlight the skills and talents of graduating WPI students.

Copyright © 2008-2009 Skyler Clark, Chris Drouin, and Matthew Fabian
Half-Life is a trademark of Valve Corporation. Copyright © 1996-2009

*Figure 33 – Petrified Website (About)*

*Petrified* was directly promoted through two sites: a dedicated website hosted at www.petrifiedmod.com , shown in Figure 33, as well as a webpage hosted through ModDB, a popular website for mod-enthusiasts.  The ModDB-provided site was more frequently updated and referenced, because it had numerous useful tools and services available for free.  It was also linked to a strong community of gaming enthusiasts, enabling them to explore and learn about our game prior to its release.

To date, *Petrified* on ModDB has 63 "watchers," which in this case are members of the ModDB site interested in hearing about the future development.  The site has also received approximately 5,000 views; some of the individual videos available through the site have been viewed over 900 times, which suggests that many users are taking more than just a cursory look at the content provided by the team.

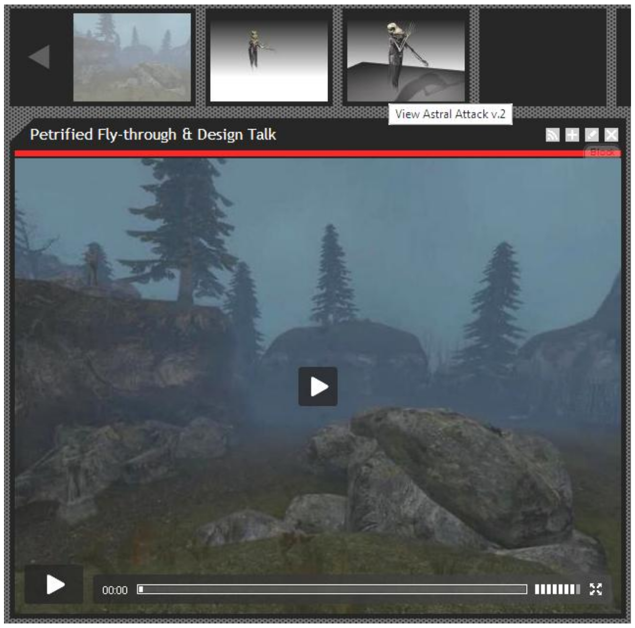


*Figure 34 – ModDB Multimedia Gallery*

ModDB allowed the team to create multimedia galleries, shown in Figure 34, of game images (screenshots, renders, and even pages from the manual) and videos. The team used the image gallery to visually document development progress. Towards the later stages of development, the team also created several videos to promote the game. Two of these showcased animation sequences that could not be represented through still pictures, and one was an in-engine video that explored the playtest map.

The team also reached out directly to gaming communities that might have an interest through their forums. Playtesters were solicited from the following forums:

- Unknown Worlds / Natural Selection – Offtopic (unknownworlds.com)
- Press Start (platformers.net)
- King of Development: Eyes of Burning Indie (selectbutton.net)

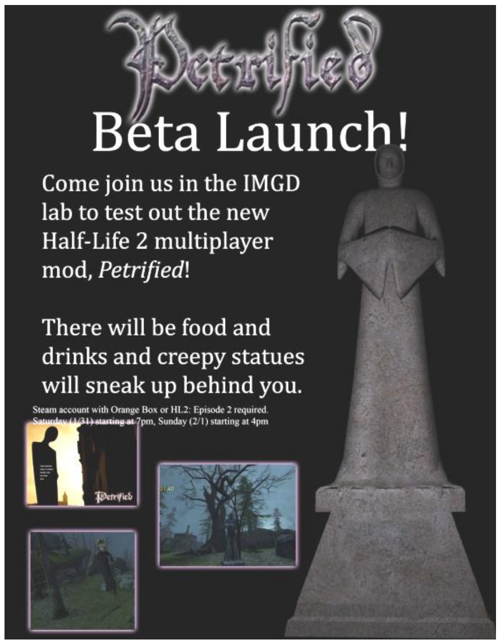

Petrified
Beta Launch!
Come join us in the IMGD lab to test out the new Half-Life 2 multiplayer mod, Petrified!
There will be food and drinks and creepy statues will sneak up behind you.
Steam account with Orange Box or HL2: Episode 2 required.
Saturday (1/31) starting at 7pm, Sunday (2/1) starting at 4pm

*Figure 35 – Petrified Beta Launch Poster*

Finally, the team searched for on-campus testers by inviting other WPI students via posters (Figure 35), mass emails, and word-of-mouth.

### 7.1.2 Internal Testing

The *Petrified* internal tests ran on two separate nights in 2-hour blocks (60 or more rounds of *Petrified* each). These tests were open to anybody on the WPI campus at the time; in practice, though, the majority of the participants were friends of the development team or WPI IMGD students who joined in soon after.

Participants also needed access to a *Steam* account with a copy of *Half-Life 2: Episode 2*, though the team had four appropriately outfitted accounts available for those without *Steam* accounts of their own. Most rounds involved between four and six players.

During internal testing, two maps were available, as shown in Figure 36.

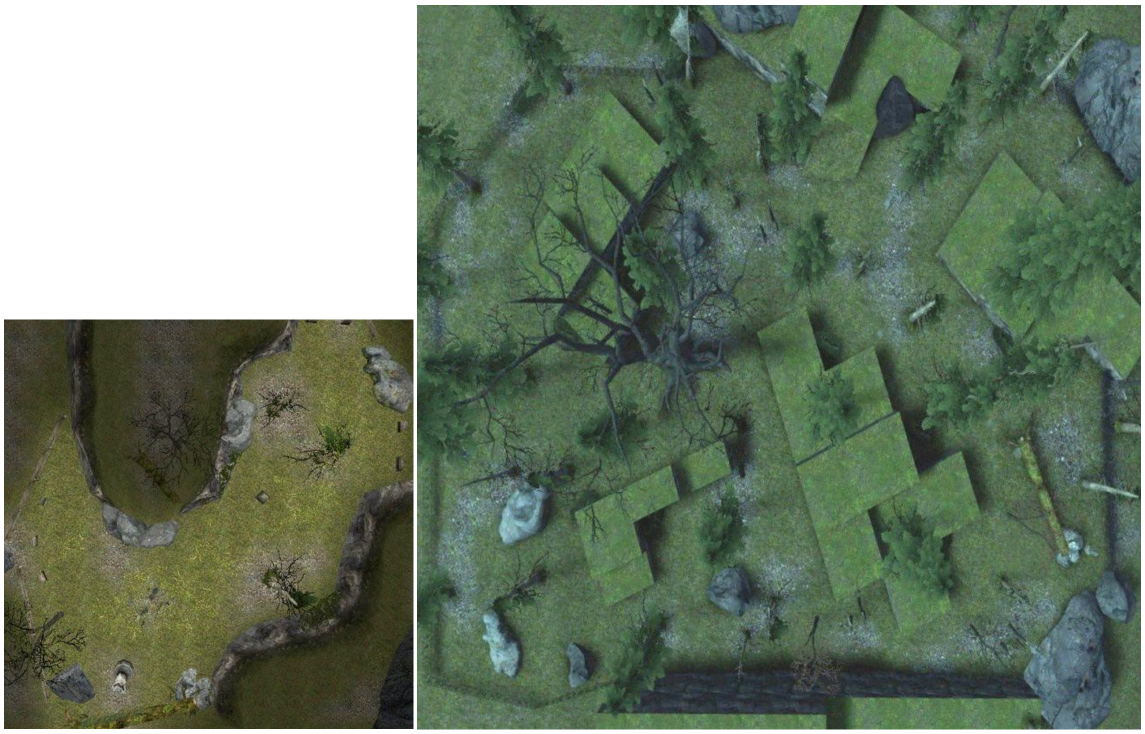


*Figure 36 – Overhead Views of 'pet_test' (left) and 'pet_graveyard' (right) Maps*

The smaller map, `pet_test`, was created as a proof-of-concept. It was not fully enclosed; players could escape the boundaries and hide in 'broken' segments of the map. The larger map, `pet_graveyard`, was centered on a large tree as a landmark. It was also fully enclosed. The terrain, however, was still in a relatively early 'blocked out' state, making it less visually appealing and somewhat more difficult to navigate.

At the end of the playtests, testers were encouraged to give feedback via paper feedback forms.

### 7.1.3 External Testing

The external playtests for *Petrified* were scheduled several days in advance, so that the team could accrue a large enough set of participants for each test. There were four external playtests in all, scheduled near the weekend at varying hours so that players from all over the world could participate comfortably. The first two tests were run for roughly an hour and a half each, while the latter tests both lasted an hour.

To participate in the external playtests, interested players had to request membership in an exclusive *Steam* group, the "Petrified Playtesters." These players were generally gaming enthusiasts, drawn from communities dedicated to discussing and playing games. By the final external playtest, the ”Petrified Playtesters” group contained 50 volunteer members, averaging 26 hours of gaming a week through *Steam*-platform games. This group was very important in coordinating the testing events themselves, since it allowed the team to

- create pop-up schedule items visible to group members
- control distribution of the game client
- communicate with testers *en masse* prior to and during the tests

*Petrified* continued to evolve throughout the external playtests. The team implemented numerous bug and balancing fixes, and developed several iterations of a new map, which was utilized during the playtests. Figure 37 shows an overhead perspective of this map in its final form.

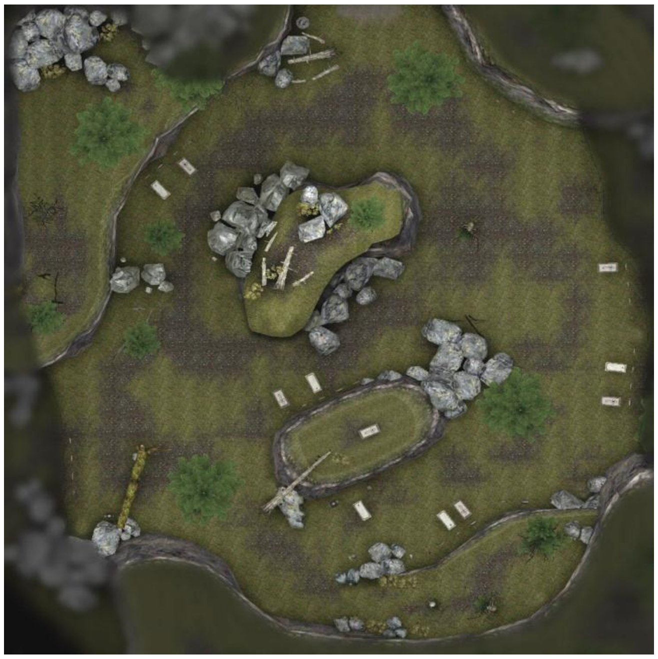


*Figure 37 – Overhead View of 'pet_medium' Map*

This map, designated `pet_medium`, was designed in part from feedback gathered during the internal playtests. The map itself evolved throughout the course of the external tests; these changes will be discussed in section 7.2.2. Since some simple game variables could be tweaked directly through console variables, the team conducting some testing of changes during the ongoing playtests. These tests included changes to fear point accumulation rate, round length, and teleport range.

During the actual playtest sessions, there were usually between five and eight testers; though the server was limited to eight players at once. Players were encouraged to communicate via headset and microphone, in an attempt to mimic the open environment of the laboratory during the internal tests.

Following each test, testers were encouraged to respond to an online survey about their experience provided through surveymonkey.com.

## 7.2 Results

The overall response to *Petrified* was very positive. Players reported being immersed in the game world, and it was compelling enough that many testers in all sessions stayed for dozens of rounds; some even came to multiple playtest sessions.

In general, the regular gamers who tested *Petrified* were more critical of the experience. They carried over some knowledge and strategies from similar titles, giving them a marginal advantage, and they knew how to look for glitches and exploits. Inexperienced gamers also provided invaluable feedback, particularly regarding the process of learning the game controls and rules.

### 7.2.1 Internal Testing

The internal tests were the very first tests run, so the game was still lacking some of the polish of the later builds. Two maps (Figure 36) were available at this time: the large `pet_graveyard`, designed for higher numbers of players, and the small `pet_test` map, which was primarily created as a proof-of-concept. Despite this, the `pet_test` map proved more popular with playtesters. Its slightly claustrophobic environment worked well for fast, energetic engagements. It also became apparent that the statue placement in the `pet_graveyard` map was undesirable for both teams – the statues physically hindered Mortal movement through the map, but they were also placed such that most could be seen by all Mortals at all times, paralyzing the Watcher team.

*Figure 38 – Testers teaching each other about Petrified*

The open lab environment that the tests were conducted in encouraged mentoring and communication. As can be seen in Figure 38, testers took the time to explain the game's controls and mechanics to each other. Throughout the playtests, the testers could be heard cheering in victory and yelping with fright and surprise. One Watcher player claimed that his favorite moment was "teleporting right behind a Mortal and hearing them go 'Oh, God!'," but another tester stated how much he enjoyed "being a mortal and trying to stay alive."

At the end of the playtests, the team gathered results via feedback/comment sheets, seen in Appendix E – Internal Playtest Feedback Questions. Feedback was universally positive. Most players felt that the game was biased in favor of the Watchers, though Mortals still managed to win many times; many also commented on their initial difficulty in grasping the Watcher controls, which prompted the team to create a user manual, which can be seen in Appendix F – Game Manual.

### 7.2.2 External Testing

External testing was almost exclusively carried out by experienced gaming enthusiasts. These enthusiasts were already intimately familiar with the control scheme used for the Mortals, and learned the Watcher controls quickly through experimentation and questions. Because the testers started at an advanced level of play, they quickly began to discover more complicated and effective play strategies. Mortal players began to move in formations and seek sheltered hiding spots, while

Watcher players moved statues to control chokepoints in the map and bided their time to accumulate fear points for successive attacks.

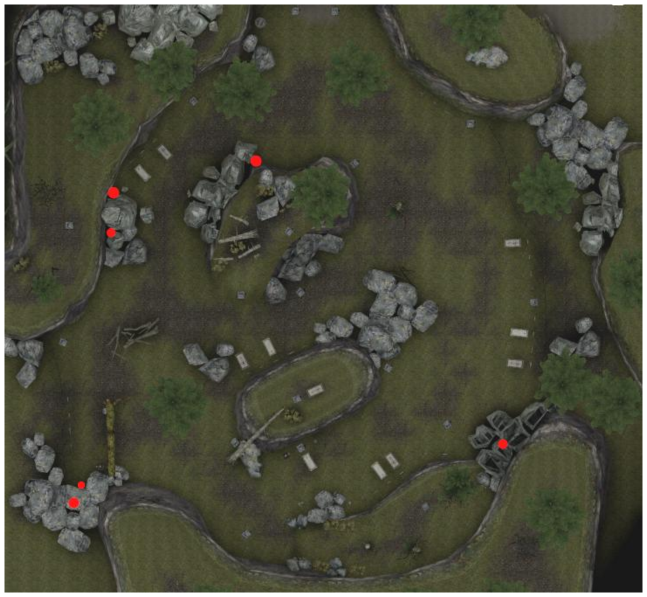


*Figure 39 – Hiding Spots Identified in 'pet_medium'*

During the early external tests, most testers felt that the Mortals had a mild advantage over the Watchers, in the sense that they won somewhat more frequently. They faulted certain Mortal-related exploits, such as the ability to boost velocity by repeatedly jumping across the level, and the map's surfeit of hiding or "camping" spots (Figure 39) for creating this imbalance. The team modified the Mortal movement code to counter the jumping exploit. The hiding-spots problem was more complicated; the team used a combination of changes to the map geometry (exposing some of the hiding spots) and initial statue placement to discourage this behavior.

Many players requested a further objective or activity for Mortals – for instance, collectible items, or objectives to complete on a linear map. This was unfortunately beyond the scope of the project, but this feedback was noted for potential future extensions to the *Petrified* game.

Players appeared to enjoy playing on both teams. In an early test, one tester responded that "Beeing [*sic*] the last mortal left and running away from seven watchers is pretty tense," while another tester felt that killing two Mortals with a single lunge attack was "fairly epic."

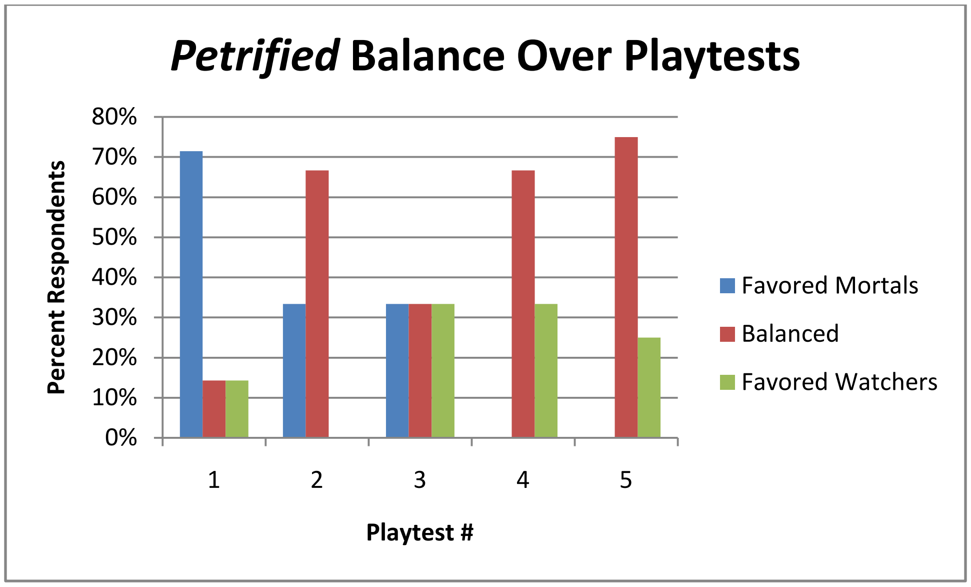


*Figure 40 – Chart – Petrified Balance by Playtest*

The team was also able to collect quantitative data, some of which is shown in Figure 40, on testers' opinions regarding game balance through the voluntary feedback surveys. Most of these surveys garnered between five and eight responses, though the third only attracted three responses. As the playtests progressed, more and more respondents said that they felt the game was balanced. The fixes may have given the Watchers a slight advantage, however. A few respondents in the later testing sessions claimed that they felt that experienced Watcher players had an advantage; one felt that Watchers "had the bigger edge" due to their ability to "teleport, [make] ranged attack[s], instant switch, build blockades, [and engage in] teamwork with others later on."

# 8 Conclusion

The *Petrified* project team successfully completed the three planned development phases – design, implementation, and evaluation – and delivered an end product in line with the original concept.

During the design phase, the team laid out a full specification for the game, detailing its gameplay mechanics, artistic style, and technical architecture. The design went through several revisions, as the team sought to remove ambiguities, clarify the vision, and address concerns over design issues. At the end of the design phase, the design was effectively frozen, to prevent new features from being introduced haphazardly and skewing the existing extensively reviewed design. The only changes permitted were those due to technical and time constraints.

The team developed the design into a working prototype during the implementation phase. Through this phase the team utilized iterative development to create successively more functional and complete builds of the game. The artistic and technical work ran parallel through this phase, with Matt Fabian developing art assets and designing levels while Skyler Clark and Chris Drouin worked with the *Source* SDK to implement the game client and server.

*Petrified* was rigorously tested through both internal, local-area tests and globally accessible external tests. Based on tester feedback during this evaluation phase, *Petrified* is fun to play, relatively balanced, and immersive. Testers reported enjoying their time with the game. They found themselves immersed in the fear and unease of the Mortal play experience, and they felt empowered when playing as Watchers. The testers also discovered several bugs and examples of exploitable behavior, which the team corrected.

While *Petrified* is now playable and has been evaluated as fun, its development is not concluded. Some aspects of the original design remain unimplemented, and the Mortal player experience is not as polished and compelling as the Watcher play experience. The team intends to continue work on *Petrified* beyond the scope of this project.

## Appendix A – Reference Material

These works contain information that may prove useful in developing and implementing the *Petrified* design:

Guilfoyle, Erik. *Half-Life 2 Mods for Dummies.* Wiley Publishing: Indianapolis, 2007.

"HL2World Knowledge Base." HL2World. URL:
http://www.hl2world.com/wiki/index.php/Main_Page

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

"Tutorials." Mod DB. URL:
http://www.moddb.com/tutorials?filter=t&kw=&subtype=def&meta=def&game=61

## Appendix B – Asset Listing

*Table B1 – Texture asset listing*

| **Texture Assets** | | | |
|---|---|---|---|
| *Name* | *Description* | *Sources* | *Completion %* |
| Dirt (variation 1) | 512x512, for ground | HL2, photography | 100% (HL2) |
| Dirt (variation 2) | 512x512, for ground | HL2, photography | 100% (HL2) |
| Grass (variation 1) | 512x512, for ground | HL2, photography | 100%(HL2) |
| Grass (variation 2) | 512x512, for ground | HL2, photography | 100%(HL2) |
| Rock (variation 1) | 512x512, for ground/debris | HL2, photography | 100% (HL2) |
| Weed (variation 1) | 256x256, for foliage | HL2, Photography | 100% (HL2) |
| Weed (variation 2) | 256x256, for foliage | HL2, Photography | 100% (HL2) |
| Weed (variation 3) | 256x256, for foliage | HL2, Photography | 100% (HL2) |
| Iron | For iron gate/fence | Original creation | 100% (Original) |
| Mortal skin | Clothing, skin, etc. laid over Mortal model | Original creation | 0% |
| Watcher ghost skin | Bone, rags laid over ghost model | Original creation | 100% (Original) |
| Base statue skin | Weathered/moss-covered stone over statue | Original creation | 100% (Original) |

*Table B2 – Model asset listing*

| **Model Assets** | | | |
|---|---|---|---|
| *Name* | *Description* | *Sources* | *Completion %* |
| Base Mortal | College-age human, geared for chilly outdoor exploration | Original creation, HL2 (walking, running) | 0% (used terrorist model) |
| Base Statue | Weathered statue styled after a hooded religious figure in prayer | Original creation | 100% (original) |
| Ghost | Skeletal ghost figure draped in rags, w/dog's skull head | Original creation | 100% (original) |

| | | | |
|---|---|---|---|
| Iron fence | Wrought iron fencing w/variations, gate | Original creation | 100% (original) |

*Table B3 – Sound asset listing*

| **Sound Assets** | | | |
|---|---|---|---|
| *Name* | *Description* | *Sources* | *Completion %* |
| Footstep (grass/dirt) (x3 variations) | WAV clip, event (mortal walking on soft material) | HL2 | 100% |
| Footstep (stone) (x3) | WAV clip, event (mortal walking on hard material) | HL2 | 100% |
| Flash-stick strike | WAV clip, event (mortal hitting ghost) | Online, recording | 100% (original) |
| Statue teleport | WAV clip, event (watcher teleporting) | Online, recording | 100% (original) |
| Statue swap | WAV clip, event (watcher swapping statues) | Online, recording | 100% (original) |
| Normal ghost attack (x3) | WAV clip, event (ghost watcher attacking Mortal) | Online, recording | 100% (original) |
| Lunging ghost attack | WAV clip, event (ghost watcher lunging at Mortal) | Online, recording | 100% (original) |
| Mortal death (x2) | WAV clip, event (Mortal dying) | Online, recording | 100% (original) |
| Ambient wind | WAV clip, ambient (occurs in stone passage) | HL2 | 100% |

## Appendix C – Original Project Pitch

This is the final version of the original project pitch, sans table of contents, as it was presented in late April of 2008. It explores the aspects of *Petrified* that make it entertaining for players on both teams, and it discusses several potential issues with the basic design (as well as several potential solutions to those issues).

## One Sentence Description

Don't turn your back, or you may find your heart turned to stone – and in Petrified, if you want to survive until dawn, you'll need to keep a very close eye on the waiting statues, lest they add your soul to their ranks.

## One Paragraph Description

Can you last until dawn? In Petrified, a source powered multiplayer game, a band of humans is pitted against an array of statues and gargoyles. Most of those statues are harmless – at least to begin with – but a few are flitting closer, just out of the corner of the eye, looking to steal human souls and swell their own numbers. As a statue you can enjoy the thrill of the hunt waiting for the right time to strike. As a human you'll be running around a chilling landscape that never moves yet constantly changes, and when you have finally realized which statues are real and which are fake, it may be too late!

## Why is it Fun?

### Team Meat

The main reasons it will be fun to play as a human in Petrified will be for the level of fear it presents, the challenge, and the teamwork and camaraderie.

Many people watch horror movies because they like to be scared. This is one reason people find survival games fun. The issue with the standard zombie survival formula is that there is no fear; players have been desensitized to zombies of all sorts, and it is simply neither scary nor challenging to deal with anything less than a large pack of zombies. In a standard zombie survival game you need not fear a lone zombie, but in Petrified there is no such thing as a lone statue. Even with just the one starting opponent there are many statues that could potentially spring to life and attack.

This will help to instill fear in the human players, as will the environment. The game is set at night in a graveyard style setting with their only hope to be to survive until dawn.

Other than the challenge of survival (against stacked odds), there are no real challenges in a zombie survival game; the human players just pump bullets into the zombies. There is not much speed to the action, which decreases the capacity of zombie game to instill fear. People do find the challenge of survival a compelling enough reason to play a zombie survival game, but Petrified will provide that as well as a more action-packed gameplay experience. By increasing the speed of the threats, the challenge to overcome them will also be increased, as will a player's adrenaline. A faster pace of gameplay will be accomplished by creating a constant need for humans to be on the move, as well as to have an active environment (e.g. lightning, thunder, fog, the sun rising).

Some players do not enjoy the challenge to survive or to be scared. Petrified will offer something for them as well. In Petrified, the use of teamwork can help the humans to survive. The humans will be provided with methods to fight back against the statues, and as with any weapon, they will be far more useful when in greater numbers. This will provide some incentive to stick with their team. Being able to try to save their other teammates by yelling to them or by simply freezing their assailants will also help to create a feeling of camaraderie. This will make the game more fun as it will make the player feel as if they are among friends.

Of course, all players have different tastes, and many may enjoy our game for a combination of the reasons above, while others may only desire to become the starting statue. In that case, another element may come in to play- strategy and betrayal. The surviving humans at the end of each round form the candidates for the starting statue in the next round; reducing the number of survivors through treachery improves individual odds, but decreases the chance that the human team will ultimately survive. While there will be no friendly fire to prevent team killing, there will be nothing to stop a human from "looking the other way" as a statue attacks their teammates.

The role of treachery in a player's enjoyment of the game is also highly relevant when they are on the statue team, as it will be their main motivation to attack their former teammates. One cannot guarantee a quick shift in allegiance, of course, but revenge is a good motivator.

## Team Stone

From the statues' perspective the game changes completely. The game becomes one of strategic placement – arranging statues in such a way that efficiently switching between them can provide an angle of attack in any situation. Skilled and creative players will be able to stalk and hunt human players, adding to the tension and atmosphere of the game.

The biggest draw to playing on the statue team will be “the hunt”. People find it fun to overwhelm those that are weaker or vulnerable – which is more or less why griefers exist, after all. This will offer a relaxing break from the high-challenge and fear induced stress of playing as a human.

Many games have a reward system to encourage players to play the game, which help to make a game more fun and less repetitive. By capturing humans, statue players will be rewarded with enhanced abilities. This will make it fun to be a statue even if you are not the starting statue.

The starting statue will likely be the most fun. The actual details to the starting statue will be ironed out in the planning stage of the MQP. We currently have two concepts for this role. The first would be that the initial statue is given a commander-like role and can decide where statues are initially placed, the types of statues, and during the game, when they are actually activated. The second idea would be for the starting player to be given a statue with special abilities to help capture humans when they are alone on the statue team.

Despite the weakness of the human players, strategy will still be required. The units move in a method that has not been explored in previous games, and this type of movement requires timing and patience. The gameplay produced by this movement mode can be likened to “real time chess”. The second movement mode of statues is more familiar, and offers a highrisk/high-reward/high-action style of play – the “astral forms” of the statues can move freely, but are visible and vulnerable to human attack.

## Gameplay Issues & Possible Solutions

### Corners for the Cowards – Turtling Players

#### Issue

If a player or many players hide in a corner, there will be no way for a statue to get behind and turn them to stone.

#### Potential Solutions

- Panic/Anxiety System

  - Human characters will get nervous / scared from staying in one location for too long, and will start to behave erratically. They may start hearing sounds or seeing things that are not there and the player's view may move on its own accord to look at these cues.
- Level Design
  - If levels are designed so that walls are not safe, for example if they have windows or if the walls are fences, they will be vulnerable to attack from behind.
- Statue Secondary Movement Mode
  - All statues will be able to go through objects such as fences, and in addition to this some statue types will be able to fly, while others may be able to go underground. Both of these will help to dissuade turtling.

## It's Not Nice to Stare – Formations, etc

### Issue

If a human player stares at a statue, the statue is unable to move. This problem is compounded with multiple players staring or using formations.

### Potential Solution

- Anxiety System
  - This would be the same as described above. This solution may not be as effective because players could be on the move or could take turns look towards the statues to keep anxiety low.
- Lightning Strikes
  - Lightning will randomly flash, and when it does it could temporarily blind players. The likelihood of lightning strikes may be increased when a human has been staring for a while or in one place for too long.
- Statue Secondary Movement Mode

  - In this case, the flying or burrowing units would be highly effective in breaking up formations.

- Statue Area-of-Effect Abilities
  - It could be possible that a statue teleporting from out of player sight into player sight could set off a lightning strike flash bang. This would mean the statue is now stuck, but would allow other statues to move in at this time. This may be better than just doing flashes randomly as it would require teamwork, but this solution would definitely require testing to see how it affects the balance and gameplay.

## Where'd You Come From?

### Issue

To maintain the illusion that the statues themselves never move, players can't see them move into their line of sight, either.

### Potential Solution

- Lightning Strikes
  - Cover the statue's arrival with a dramatic flash of lightning. The statue's movement cooldown will prevent it from being able to pounce immediately after, but this may raise balancing issues.
- Blocked Movement
  - Simply don't allow movement into player line of sight. This still gives statues something of an advantage, though, since blocked movement indicates an approaching human. With this option, it might make sense to visually represent the "visibility cones" of the human players for the statues.

## Humans Have No Chance to Survive

### Issue

If it is impossible for the humans to win or if the odds are stacked too high it may not be fun to be a human.

### Potential Solution

- Statue Vulnerability
  - The solution here lies in the statue’s vulnerable form that they must assume to attack a player. When in this form, the humans could use a gun that would be able to destroy the statues while they are in astral form. Mainly just balancing the speed of the astral forms, the vulnerability of them to the human’s weapon, the level design, and focus on teamwork should ensure that it is fun.
- Timed Survival Goal
  - The player must survive until dawn. This provides a timed goal without destroying the immersion by having an actual countdown timer. We could do this just with the environment, skybox, and lighting for a very atmospheric effect. The exact time for this goal will come down to play-testing, as presumably shorter survival times will be easier and long times harder.

## Repetition, Repetition, Repetition…

### Issue

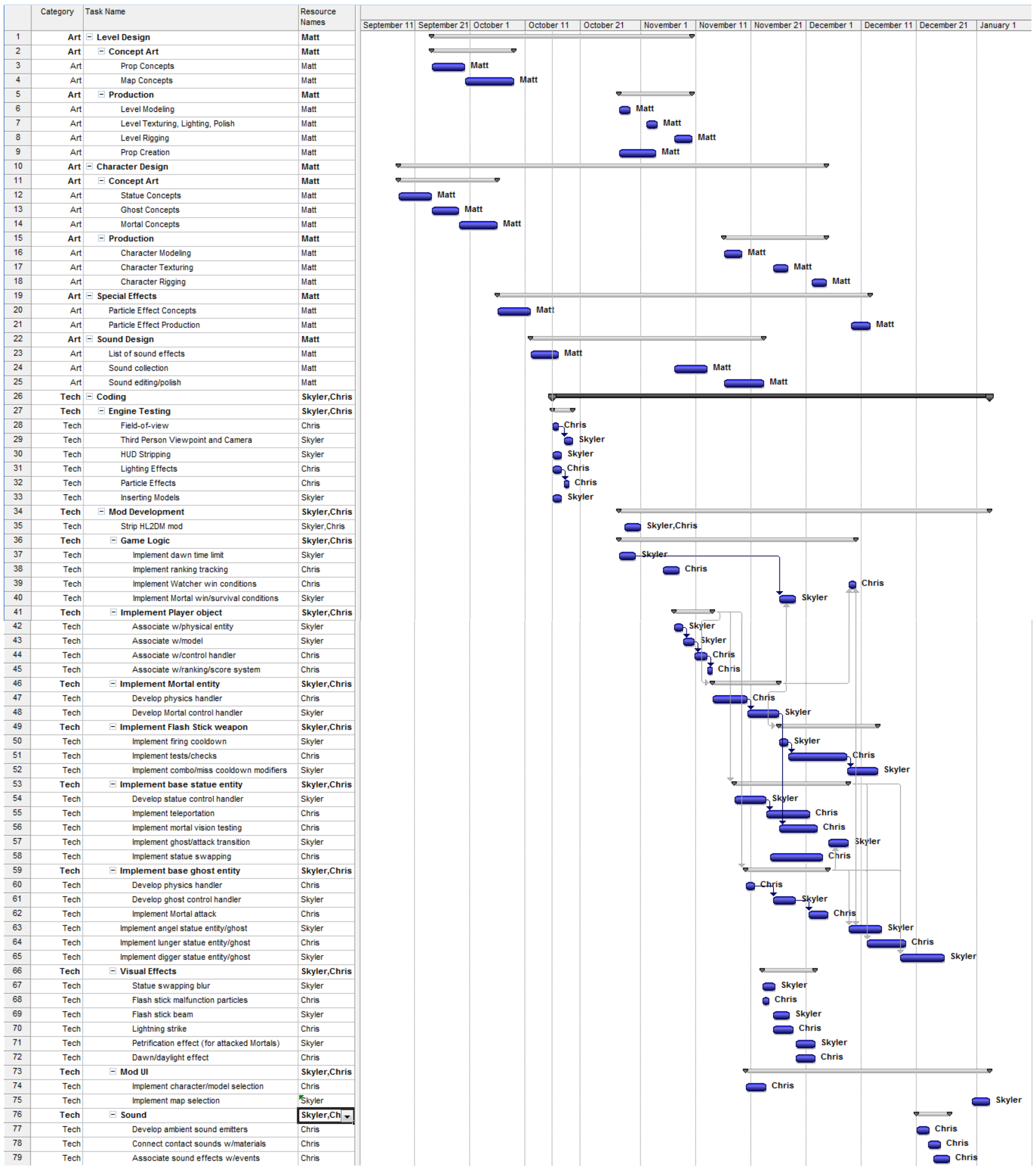
Since the player will not have any loyalty to their team since they are always changing, the game will become boring if it is too repetitive. All games become repetitive; we need to prevent this from occurring at a faster rate due to the lack of persistent teams.

### Potential Solution

- Randomized Environment
  - Different starting positions for humans and statues will help to prevent an identical gameplay experience each round.
  - Further environmental randomization (rearranging map elements), largely
- Varying abilities and units
  - Different unit types and abilities will ensure that each game does not play the same, and if one strategy gets boring a player may try another.

# Appendix D – Gantt Charts

This chart, developed in Microsoft Project, lays out the implementation tasks that the team must complete during the second phase of the project. The subtasks and task assignments are tentative.



| | Category | Task Name | Resource Names |
|---|---|---|---|
| 1 | **Art** | **Level Design** | **Matt** |
| 2 | **Art** | **Concept Art** | **Matt** |
| 3 | Art | Prop Concepts | Matt |
| 4 | Art | Map Concepts | Matt |
| 5 | **Art** | **Production** | **Matt** |
| 6 | Art | Level Modeling | Matt |
| 7 | Art | Level Texturing, Lighting, Polish | Matt |
| 8 | Art | Level Rigging | Matt |
| 9 | Art | Prop Creation | Matt |
| 10 | **Art** | **Character Design** | **Matt** |
| 11 | **Art** | **Concept Art** | **Matt** |
| 12 | Art | Statue Concepts | Matt |
| 13 | Art | Ghost Concepts | Matt |
| 14 | Art | Mortal Concepts | Matt |
| 15 | **Art** | **Production** | **Matt** |
| 16 | Art | Character Modeling | Matt |
| 17 | Art | Character Texturing | Matt |
| 18 | Art | Character Rigging | Matt |
| 19 | **Art** | **Special Effects** | **Matt** |
| 20 | Art | Particle Effect Concepts | Matt |
| 21 | Art | Particle Effect Production | Matt |
| 22 | **Art** | **Sound Design** | **Matt** |
| 23 | Art | List of sound effects | Matt |
| 24 | Art | Sound collection | Matt |
| 25 | Art | Sound editing/polish | Matt |
| 26 | **Tech** | **Coding** | **Skyler,Chris** |
| 27 | **Tech** | **Engine Testing** | **Skyler,Chris** |
| 28 | Tech | Field-of-view | Chris |
| 29 | Tech | Third Person Viewpoint and Camera | Skyler |
| 30 | Tech | HUD Stripping | Skyler |
| 31 | Tech | Lighting Effects | Chris |
| 32 | Tech | Particle Effects | Chris |
| 33 | Tech | Inserting Models | Skyler |
| 34 | **Tech** | **Mod Development** | **Skyler,Chris** |
| 35 | Tech | Strip HL2DM mod | Skyler,Chris |
| 36 | **Tech** | **Game Logic** | **Skyler,Chris** |
| 37 | Tech | Implement dawn time limit | Skyler |
| 38 | Tech | Implement ranking tracking | Chris |
| 39 | Tech | Implement Watcher win conditions | Chris |
| 40 | Tech | Implement Mortal win/survival conditions | Skyler |
| 41 | **Tech** | **Implement Player object** | **Skyler,Chris** |
| 42 | Tech | Associate w/physical entity | Skyler |
| 43 | Tech | Associate w/model | Skyler |
| 44 | Tech | Associate w/control handler | Chris |
| 45 | Tech | Associate w/ranking/score system | Chris |
| 46 | **Tech** | **Implement Mortal entity** | **Skyler,Chris** |
| 47 | Tech | Develop physics handler | Chris |
| 48 | Tech | Develop Mortal control handler | Skyler |
| 49 | **Tech** | **Implement Flash Stick weapon** | **Skyler,Chris** |
| 50 | Tech | Implement firing cooldown | Skyler |
| 51 | Tech | Implement tests/checks | Chris |
| 52 | Tech | Implement combo/miss cooldown modifiers | Skyler |
| 53 | **Tech** | **Implement base statue entity** | **Skyler,Chris** |
| 54 | Tech | Develop statue control handler | Skyler |
| 55 | Tech | Implement teleportation | Chris |
| 56 | Tech | Implement mortal vision testing | Chris |
| 57 | Tech | Implement ghost/attack transition | Skyler |
| 58 | Tech | Implement statue swapping | Chris |
| 59 | **Tech** | **Implement base ghost entity** | **Skyler,Chris** |
| 60 | Tech | Develop physics handler | Chris |
| 61 | Tech | Develop ghost control handler | Skyler |
| 62 | Tech | Implement Mortal attack | Chris |
| 63 | Tech | Implement angel statue entity/ghost | Skyler |
| 64 | Tech | Implement lunger statue entity/ghost | Chris |
| 65 | Tech | Implement digger statue entity/ghost | Skyler |
| 66 | **Tech** | **Visual Effects** | **Skyler,Chris** |
| 67 | Tech | Statue swapping blur | Skyler |
| 68 | Tech | Flash stick malfunction particles | Chris |
| 69 | Tech | Flash stick beam | Skyler |
| 70 | Tech | Lightning strike | Chris |
| 71 | Tech | Petrification effect (for attacked Mortals) | Skyler |
| 72 | Tech | Dawn/daylight effect | Chris |
| 73 | **Tech** | **Mod UI** | **Skyler,Chris** |
| 74 | Tech | Implement character/model selection | Chris |
| 75 | Tech | Implement map selection | Skyler |
| 76 | **Tech** | **Sound** | **Skyler,Ch** |
| 77 | Tech | Develop ambient sound emitters | Chris |
| 78 | Tech | Connect contact sounds w/materials | Chris |
| 79 | Tech | Associate sound effects w/events | Chris |

## Appendix E – Internal Playtest Feedback Questions

These questions were given to internal playtest participants in the form of a printed sheet. Testers did not have to respond to any or all of the questions. The questions were cleared via the IRB prior to the internal playtests.

---

Your participation in this evaluation is voluntary – you can quit at any time, and you do not have to respond to any or all of the questions. If you choose to divulge your email address, we will use it solely to contact you once when the open beta for *Petrified* begins. Your personal information will never be divulged in any form.

**Demographic Info**

| | |
|---|---|
| **Name** | **Email Address** |
| **Experienced Gamer?** **Y N** | **Major/Year** |

**Mortal Team**

- Please describe in brief your experience and feelings playing as a Mortal.
- Did you find the Mortal controls and abilities intuitive?

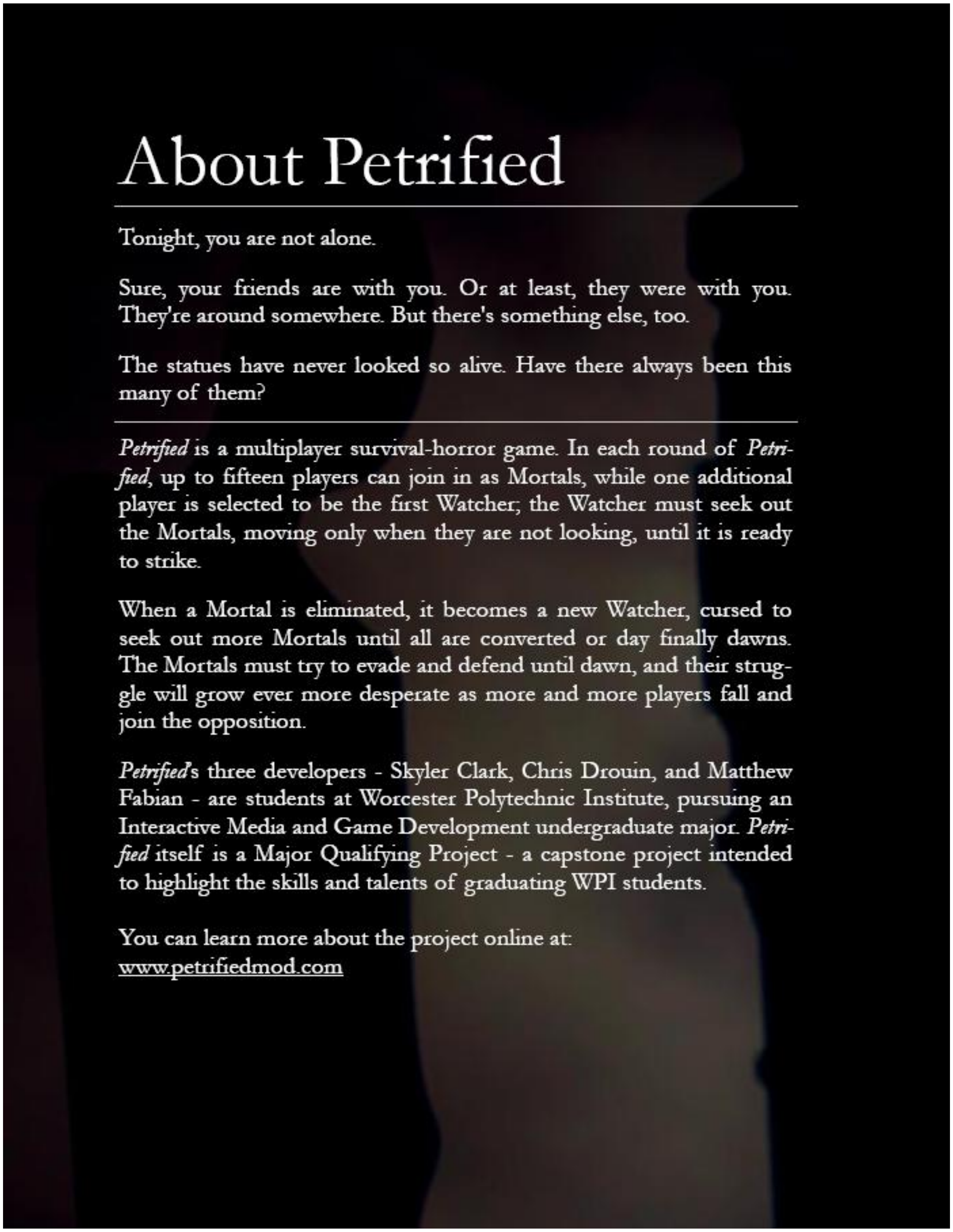
**Watcher Team**

- Please describe in brief your experience and feelings playing as a Watcher.
- Did you find the Watcher controls and abilities intuitive?

**Overall**

- What was your favorite experience or best moment from playing *Petrified*?
- What did you find most frustrating or unintuitive during your time playing *Petrified*?
- Did you find that you were able to function effectively as a team in game? Why or why not?
- Did you prefer to play as a Watcher or as a Mortal? Why?
- Please note any bugs or unexpected behavior you experienced during play (include circumstances – team, # of players in match, map, etc.)
- Would you be interested in playing *Petrified* on its own (i.e. outside of the official beta)?
- Any other comments for the *Petrified* team?

## Appendix F – Game Manual



# About Petrified

Tonight, you are not alone.

Sure, your friends are with you. Or at least, they were with you. They're around somewhere. But there's something else, too.

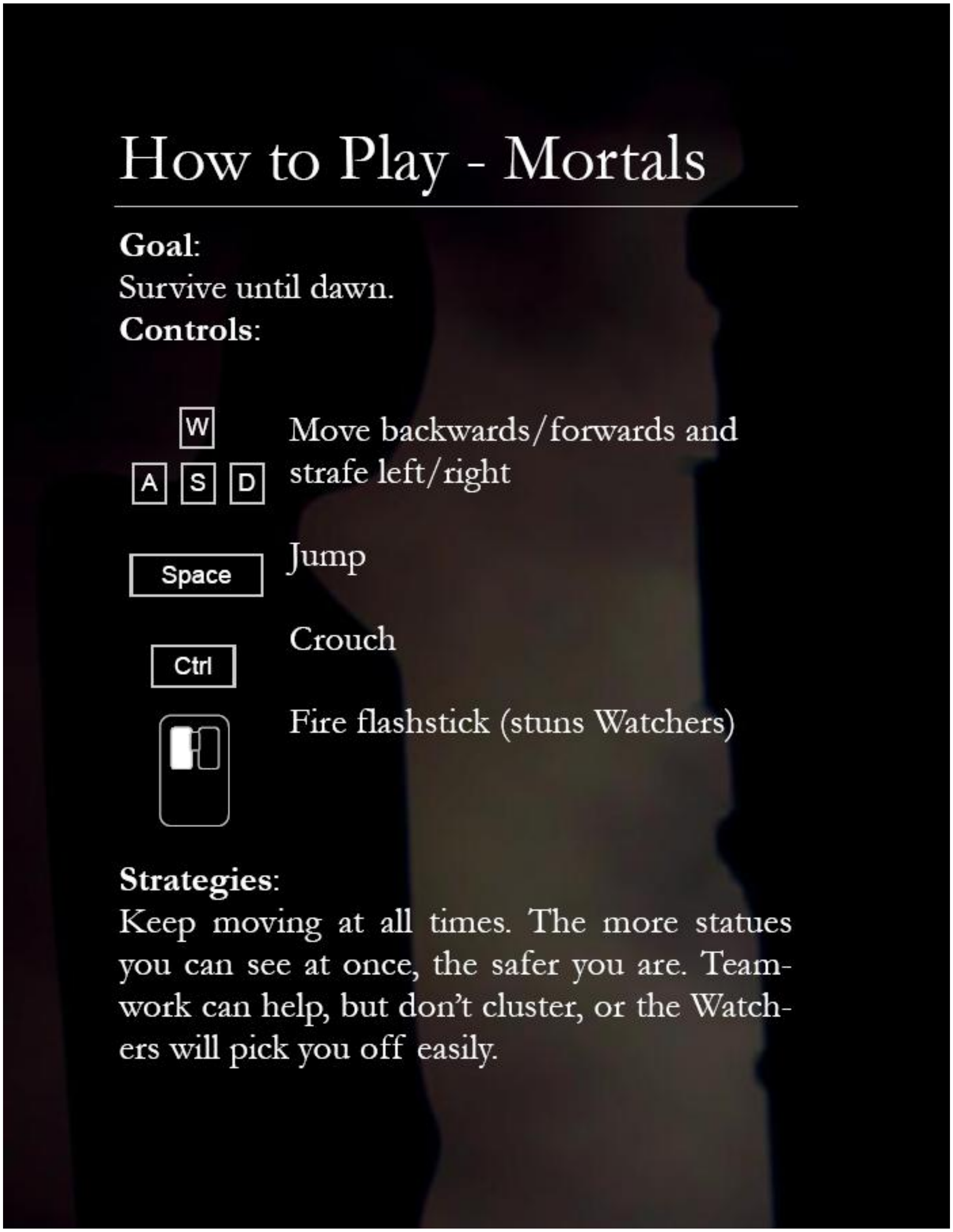
The statues have never looked so alive. Have there always been this many of them?

*Petrified* is a multiplayer survival-horror game. In each round of *Petrified*, up to fifteen players can join in as Mortals, while one additional player is selected to be the first Watcher; the Watcher must seek out the Mortals, moving only when they are not looking, until it is ready to strike.

When a Mortal is eliminated, it becomes a new Watcher, cursed to seek out more Mortals until all are converted or day finally dawns. The Mortals must try to evade and defend until dawn, and their struggle will grow ever more desperate as more and more players fall and join the opposition.

*Petrified*'s three developers - Skyler Clark, Chris Drouin, and Matthew Fabian - are students at Worcester Polytechnic Institute, pursuing an Interactive Media and Game Development undergraduate major. *Petrified* itself is a Major Qualifying Project - a capstone project intended to highlight the skills and talents of graduating WPI students.

You can learn more about the project online at:
www.petrifiedmod.com



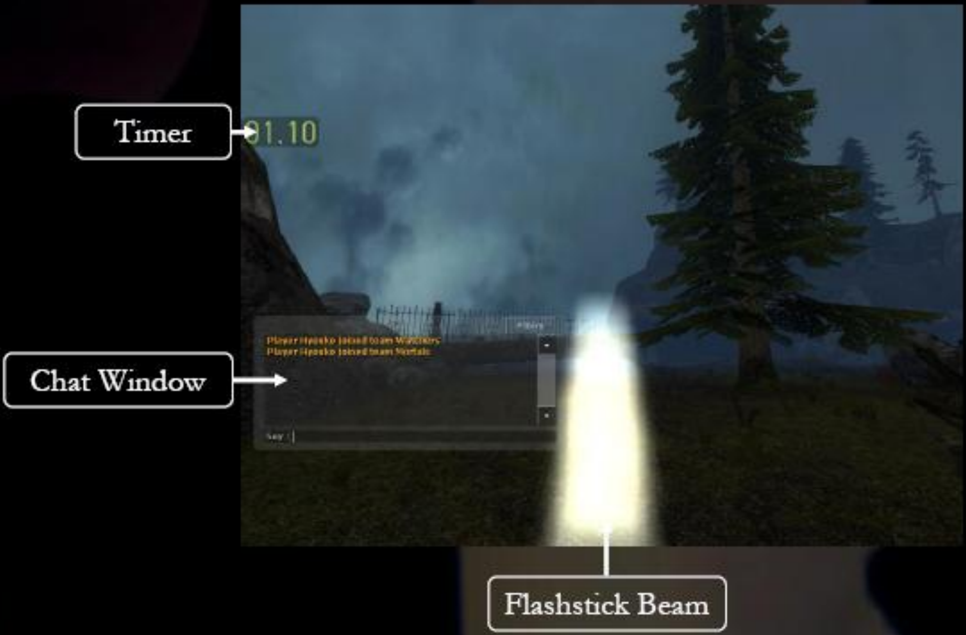
## How to Play - Mortals

**Goal**:
Survive until dawn.
**Controls**:

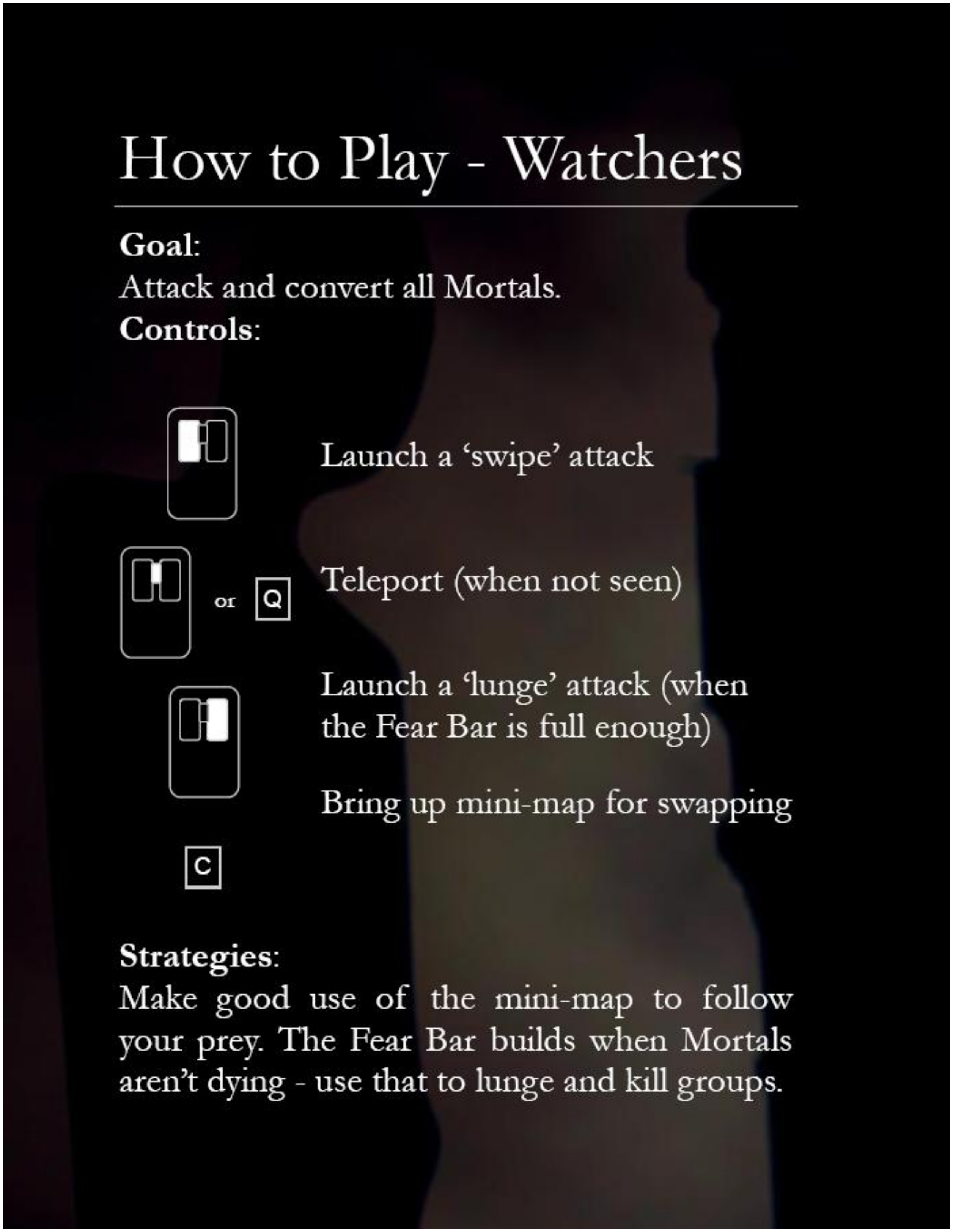
| | |
|---|---|
| W A S D | Move backwards/forwards and strafe left/right |
| Space | Jump |
| Ctrl | Crouch |
| 🖱 | Fire flashstick (stuns Watchers) |

**Strategies**:
Keep moving at all times. The more statues you can see at once, the safer you are. Teamwork can help, but don't cluster, or the Watchers will pick you off easily.

## UI - Mortals



Player Hyosko joined team Watchers
Player Hyosko joined team Mortals



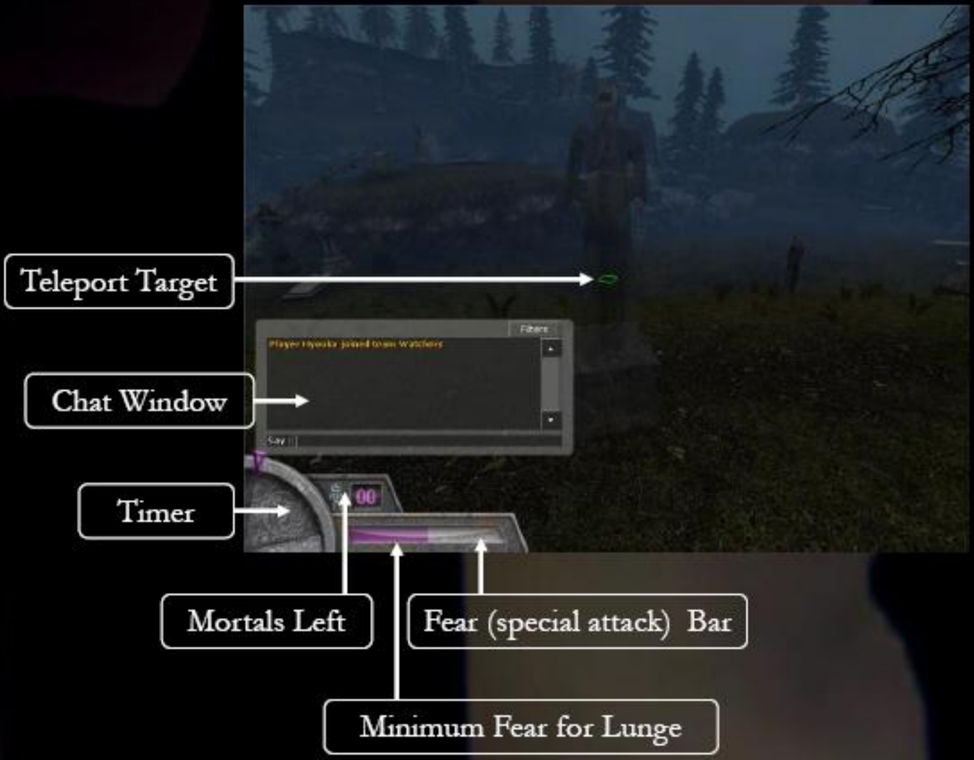
## How to Play - Watchers

**Goal**:
Attack and convert all Mortals.
**Controls**:

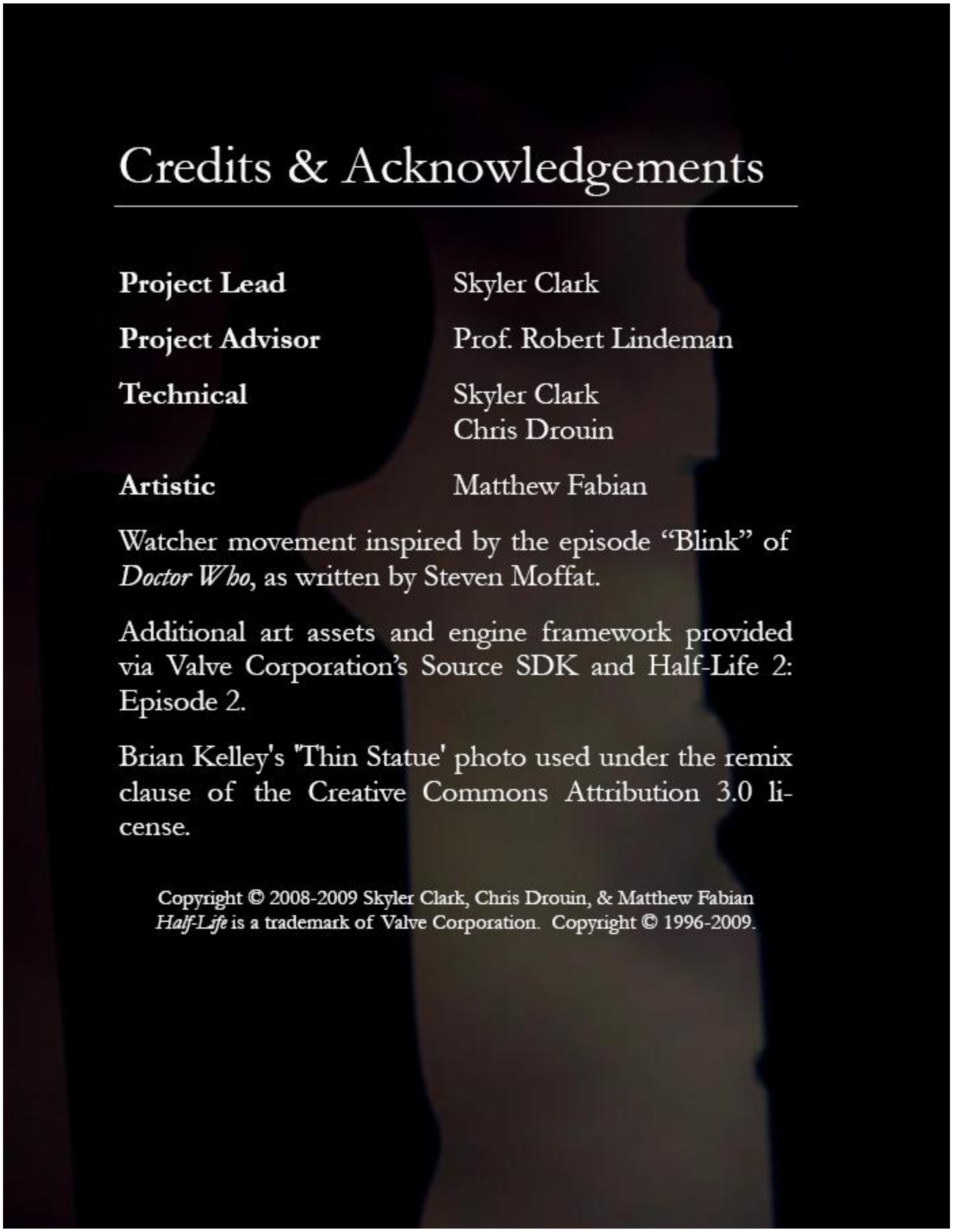
| | |
|---|---|
| 🖱 | Launch a 'swipe' attack |
| 🖱 or Q | Teleport (when not seen) |
| 🖱 | Launch a 'lunge' attack (when the Fear Bar is full enough) |
| C | Bring up mini-map for swapping |

**Strategies**:
Make good use of the mini-map to follow your prey. The Fear Bar builds when Mortals aren't dying - use that to lunge and kill groups.

## UI - Watchers



Player Hyouka joined team Watchers



## Credits & Acknowledgements

| | |
|---|---|
| **Project Lead** | Skyler Clark |
| **Project Advisor** | Prof. Robert Lindeman |
| **Technical** | Skyler Clark<br>Chris Drouin |
| **Artistic** | Matthew Fabian |

Watcher movement inspired by the episode "Blink" of *Doctor Who*, as written by Steven Moffat.

Additional art assets and engine framework provided via Valve Corporation's Source SDK and Half-Life 2: Episode 2.

Brian Kelley's 'Thin Statue' photo used under the remix clause of the Creative Commons Attribution 3.0 license.

# Appendix G – Original Design

*Petrified*'s design has changed in numerous ways since the original design phase. This section contains the original design section of the report, as it stood at the end of A-term 2008. The major differences revolve around the additional three Watcher classes and the extended fear points system that was to support them.

## 2 Gameplay Design

In *Petrified*, Mortals try to survive until dawn, while Watchers, spectral beings that inhabit statues, wait to sneak up on them and turn them to stone. Players control both sides, but the selection of which player lands on which team is determined in part by player performance in-game.

When a round begins, there will only be one player on the Watcher team. Throughout each round, Watcher players will attempt to attack the Mortals. When these attacks are successful, the Mortals' avatars turn to stone, and the affected players are transferred to the Watcher side, in control of basic statues.

The end of the round is signified by dawn, or by the elimination of the Mortals. If there are any Mortals remaining when the sun rises, the Mortals win the round, otherwise, the Watchers win.

### 2.1 Mortals

Mortal players begin each round near a group of teammates, not in the immediate presence of any Watcher-possessed statues. As they begin to move and play, the world will appear stationary, but the location of the statues will keep shifting when they are not looking, and perhaps they will hear the faint sounds accompanying Watcher teleportation and statue swapping. Alert players must correlate this information to keep track of the locations of nearby Watchers and escape them.

From the Mortal player's perspective, the game will shift between periods of uneasy calm, in which the Watchers approach but do not reveal themselves or attack, and frantic moments of fight-or-flight as the Watchers attack and multiply. This second period will be punctuated by additional environmental effects, such as lightning strikes, to emphasize the tension and danger of the situation.

#### 2.1.1 Objectives

The primary objective of each Mortal player is to evade Watcher attack until dawn, which is represented by the lightening sky. If they cannot, Mortal players are encouraged to survive as long as possible.

This goal does not encourage teamwork, but in keeping one's teammates alive a player will be able to survive longer, because he or she will have other players to watch his or her back. So while survival is a selfish goal, this aspect of survival will encourage the use of teamwork.

### 2.1.2 Abilities

The Mortal player's primary tool in managing and thwarting the progress of the Watcher-possessed statues is simply his or her field of view; any Watcher-possessed statues visible to the Mortal become unable to teleport (though they can still swap to other statues, described later).

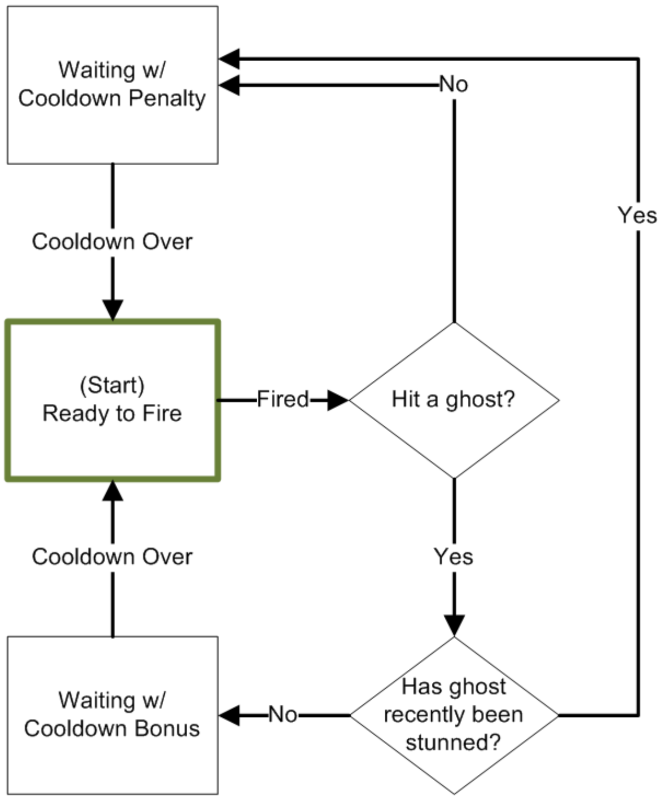

Waiting w/ Cooldown Penalty
No
Yes
Cooldown Over
(Start) Ready to Fire
Fired
Hit a ghost?
Cooldown Over
Yes
Waiting w/ Cooldown Bonus
No
Has ghost recently been stunned?

*Figure 1 – Flash-stick Mechanics*

Mortals have another defensive ability: they carry flash-sticks, which let out short bursts of light. These bursts can stun the Watchers' astral form, which is used for attack[5], providing time for the Mortals to escape or re-position. The flash-sticks are not wholly reliable, however; there is an internal system of logic that determines whether or not they work, which is shown in Figure 1.

If the flash-stick is within its cool-down phase and a Mortal player attempts to use it again, the stick will malfunction, spewing out purple sparks. As shown in Figure 1, however, successive hits on different ghosts will temporarily decrease the cool-down phase, enabling skilled players to take on multiple attacking ghosts if they can time their bursts appropriately. Also, if a Mortal player attempts to shoot a ghost that has already been stunned, their flash-stick will fail, so that the affected Watcher player is not stuck in a frustrating loop of stun attacks.

Mortal players additionally have the option to use voice communication that will address all members of their team. This ability has a penalty, however – Watchers lurking near a speaking player will be able to overhear that player, potentially compromising the Mortals' strategy.

### 2.1.3 Projected Player Strategies

The most natural initial strategy for the Mortal players will be to bunch together, seeking safety in numbers and increased vision coverage. In this phase, Mortal players may seek to move in coordinated formations that cover wide spaces with their combined field of view, effectively freezing any nearby Watchers. Such behavior, however, will ultimately contribute to the Watcher team's arsenal of abilities, as stalemated Watcher players gain powers through the Fear Point system (see 2.2.3).

Many Mortal players may also seek shelter in physical features of the environment in order to limit the ability of Watchers to approach. They may position themselves against walls or in corners. While the Watcher statues themselves cannot travel through physical obstacles, their ghosts can. Ghosts are able to travel through physical obstacles like walls, trees, and tombstones; and, in the case of the Digger statues, even through the ground. Experienced Mortal players will probably not rely on physical shelter, but they will need to know the map layout to efficiently escape from attacking statues.

[5] Hereafter the "astral form" of a Watcher will be referred to as a "ghost".

Another potential – and likely more successful – strategy is for Mortal players to head to regions with few statues. This may tend to separate the Mortals, however, making them easier to pick off. Small, coordinated groups of Mortals may be able to strike the best balance between vision coverage, mobility, and Fear Point marginalization.

Some players initially assigned as Mortals may prefer the more predatory style of the Watcher team, and so they may try to get attacked and converted by the Watchers early on. This is unfortunate, as they will not contribute to the camaraderie of the Mortal team.

### 2.1.3 Controls

Mortals are controlled through relatively standard first person shooter (FPS) controls by default. The standard configuration will be familiar to anyone who has played a Source-based game, such as *Half-Life 2*, *Counterstrike: Source*, or *Team Fortress 2.* The controls are shown in Table 1.

*Table 1 – Default Mortal Control Scheme*

| **Input (key or action)** | **Behavior** |
|---|---|
| Move mouse | Perspective follows mouse (enables turning, looking up/down) |
| Click left mouse button | Attempt to use flash-stick (see Figure 1 for behavior) |
| Press/hold [W] | Walk forward |
| Press/hold [S] | Walk backward |
| Press/hold [A] | Strafe (walk sideways) left |
| Press/hold [D] | Strafe (walk sideways) right |
| Hold [Shift] | Run (boost speed in all directions) |
| Press [Space] | Jump |

## 2.2 Watchers

Watchers are supernatural beings that typically inhabit or possess statues; they derive some of their capabilities from the statues that they inhabit. They are guardians, seeking to protect the area entrusted to them. They can leave their statue shells to attack trespassing Mortals, though emerging as ghostly astral projections makes them visible and vulnerable to attack.

The Watcher team begins with only a single player, and inhabitable statues randomly placed around the map. The last surviving Mortal of the previous round will be given the option to be this initial Watcher, but if he or she declines, or if there are no previous rounds, the role will be randomly assigned. In a case of multiple Mortals surviving in the previous round, randomly one of them will be given the option, and if they decline another surviving player will randomly be given the option. This process will repeat until there are no more surviving players to give preference to. At this point, it will just randomly choose. In all cases, the remaining players will be assigned to the Mortal team.

The player who is selected to be the first Watcher will start in a statue located some distance from the starting cluster of Mortals. As the round progresses, the Watcher team will grow in size as Watcher players attack and petrify Mortals, converting them to the Watcher team. The tide of the game turns much as it does in a "zombie" style game, with an ever-increasing number of "infected" players turned against their former comrades.

Unlike the Mortals, who view the game entirely from a first person perspective, Watchers switch between a third-person perspective, when they are inhabiting a statue, to a first-person perspective when they leave that state to attack Mortals. Also, unlike the Mortal view, which is entirely free of interface elements, the Watcher view contains cues related to knowledge that Watchers should logically have, such as teleport destinations, inhabitable statues, and fear points.

### 2.2.1 Objectives

The Watchers players' goal is to capture all of the Mortals – that is, successfully attack them so they are switched to the Watcher team. It is only possible to lose as a Watcher if the team allows Mortals to survive until dawn.

As the Watcher team grows, strategy and teamwork will become important to efficiently capture the remaining Mortals, who will doubtless rank amongst the more-skilled and experienced players. On a more abstract level, the Watchers also serve as the puppet-masters or directors of the game; as the antagonists, they deliver the most direct frights to the Mortal players, and their actions prior to attack (teleportation and swapping, see 2.2.2) serve to develop the atmosphere.

### 2.2.2 Baseline Abilities

Watchers all possess three basic abilities, regardless of the statues that they currently inhabit. Two of these abilities remain largely the same regardless of statue: the ability to teleport, and the ability to swap control to another uninhabited statue.

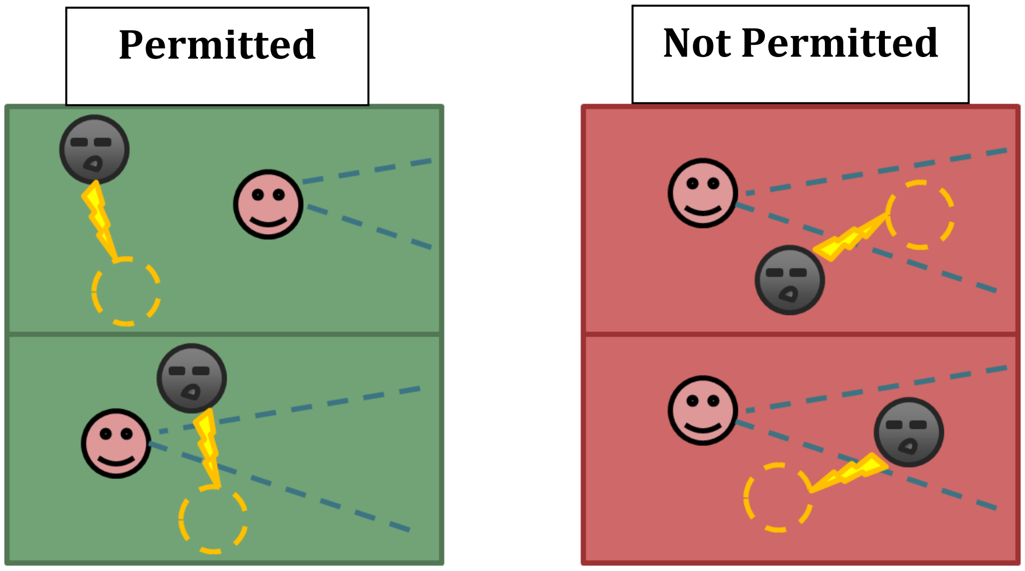


*Figure 2- Watcher Teleportation Mechanics*

Teleportation enables Watchers to quickly close the distance between themselves and the Mortals. Watchers can only teleport within a limited range, and they cannot teleport when they are visible to a Mortal. Furthermore, they cannot teleport into a region that is visible to a Mortal (Figure 2). To teleport, Watcher players will simply select the region on the ground that they wish to teleport to with the mouse; the third-person camera tracking the statue will pan left and right to follow the mouse as it moves across the terrain.

In the event of blocked teleportation, however, the game will allow Watcher players to “lock in” a teleportation destination. As soon as the destination and the origin are out of sight, the teleport will take place. Teleportation is also subject to a cool-down time that is associated with the statue – Watchers cannot teleport without pausing between moves. Teleporting also causes a small noise, which wary Mortals may notice.

So long as they inhabit a statue, Watcher players also have the ability to swap control into another uninhabited statue anywhere in the game. They can do this even if they are currently being watched by a Mortal player. This swap ability will be performed by selecting statues on a mini-map. Not all statues are immediately available, however; some of the more powerful varieties (described in 2.2.4) can only be swapped into by Watchers who have amassed enough fear points (see 2.2.3) to unlock them.

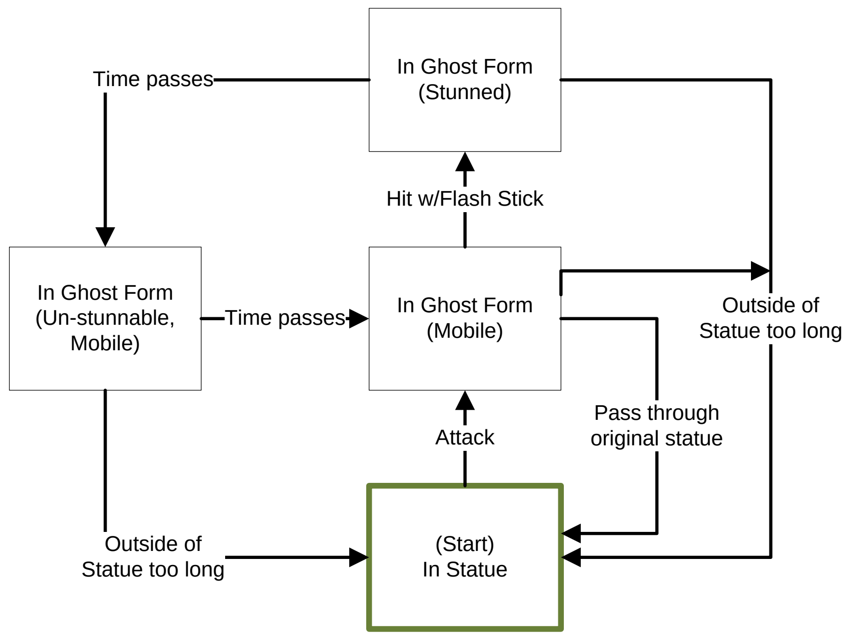

In Ghost Form (Stunned)
Time passes
Hit w/Flash Stick
In Ghost Form (Un-stunnable, Mobile)
Time passes
In Ghost Form (Mobile)
Outside of Statue too long
Pass through original statue
Attack
Outside of Statue too long
(Start) In Statue

*Figure 3 – Watcher Attack States*

Finally, Watchers are able to attack Mortals by projecting their ghost form; when the ghost passes through a Mortal, they are petrified and converted to the Watcher team. Ghost-form Watchers are vulnerable to flash-stick attacks, and can only last for a limited time (and, by extension, range) outside of their statue shells without getting sucked back in. They can move into the field-of-vision of Mortal players, but in doing so risk being stunned by a Mortal's flash-stick. Watchers can attack when they are in view of Mortals, but they run a high risk of being stunned if they do so. After being stunned, a Watcher player will have a "grace period" in which they won't be able to be stunned. This will help to prevent frustration caused by repeated stuns from Mortal players. Each type of statue has a different ghost form attack. These will be described in greater detail in section 2.2.4.

Like the Mortals, the Watchers have a dedicated voice chat channel to enable them to coordinate and strategize. While they can overhear Mortals that are talking nearby, the Mortals cannot hear talking Watchers.

### 2.2.3 Fear-Point System

Mortal players holding in formations can effectively ward off basic statues, and in general stalemated rounds are not fun for either side. The fear-point system helps mitigate this problem. Each Watcher has a separate fear-point counter; it is both a reward for creepy behavior and a means to break stalemates. Gaining the appropriate number of fear points allows Watcher players to inhabit three other statue forms, each with unique abilities that can, if used wisely, break a stalemate situation. It is important to note that while this gives a Watcher player the ability to inhabit specialized statues, the player can only control one at a time. Fear points can only be accumulated until they reach a maximum value, however – it is not possible to gain a permanent advantage by waiting for the counter to reach a very high value.

Fear points can be gained in several ways:

- All Watchers begin to gain fear points slowly if there have not been any Mortal conversions for a moderate length of time (for instance, a fifth of the total time until dawn).
- A Watcher will accumulate fear points if it inhabits a statue that is pinned down by a Mortal's stare. This does not stack – the amount of points accumulated does not depend on the number of Mortal's satisfying the condition.
- A Watcher will accumulate fear points if it can see multiple Mortals within its own field of view. The points will accumulate proportionately with the number of Mortals visible.

Just as Watchers can gain fear points, they can also lose them in multiple ways:

- Using abilities (particularly the attack ability) from specialist statues will deplete fear points, proportionally to the length of time spent using those abilities.
- Failing to return to the origin statue in time when in ghost form will incur a heavy fear point loss.

When a Watcher inhabiting a specialist statue has fully depleted its store of fear points that Watcher will be unable to do anything except swap into a basic statue.

### 2.2.4 Statue Types

There are three additional types of specialist statues, in addition to the base statue, which become available in successive order as Watchers gain fear points. These each have a different visual look (as described in section 3), and they each have unique attack abilities. That said, these statues will not be distributed as prolifically about the map, and the fear point system limits their usage.

**Base Statue**

The basic Watcher's attack will be a short-ranged attack with an approximate range of 10 feet. After being fired this attack will not be not be user-controlled – the ghost simply lunges out in the direction chosen by the player – and so it is very important for Watcher players using the base statue to maneuver close to Mortals with their teleport ability to attack effectively. Visually, it will look like the ghost leans out of the statue and slashes out at the Mortal player.

**Lunger**

The Lunger's ghost form moves much more quickly than the standard ghost form, and it is constantly propelled forward; it will be similar to a fly-by-wire missile in other games. Upon hitting a Mortal target, the Lunger ghost form rebounds back to its statue automatically. It does not, however, last much longer than a standard ghost, so it must be used with care.

This statue will unlock with the lowest number of fear points of the three.

**Angel**

This statue's ghost form has major advantages when compared to the standard statue's ghost form: it can fly, it is player-controllable, and it has a more lenient return deadline. Because of this, the ghost can attack from many new angles and dodge Mortal attacks more readily.

This statue unlocks between the Lunger and the Digger in fear points.

**Digger**

The Digger's ghost form has the strongest ability of all – that of traveling underground. When using this attack, a Watcher player can pass straight through obstacles and loop underneath Mortals to neutralize their defensive formations. It can stay in ghost form for somewhat longer than the base statue, as well. The Digger statue also has a unique visual perspective – when in the first-person Digger perspective, Mortal players will be highlighted (as if through infrared vision), while terrain elements will be obscured.

This statue requires the greatest number of fear points to unlock (near the maximum), and as such will usually only come in to play during heavy stalemate situations.

### 2.2.5 Projected Player Strategies

Most Watcher players will initially do what comes naturally: move in close to Mortal players and attack when their backs are turned. This basic maneuvering will always work to a degree, depending on player skill, but it may not ultimately be the most effective strategy. Still, careful teleportation will be the key to Watcher strategy while players are limited to basic statues; the short-range basic attack makes it necessary to be right behind Mortal players to successfully attack them.

More advanced players will make great use of the swapping mechanism, trying to position themselves within statues that will have better opportunities to strike. Waiting patiently is the key to this strategy. If a Watcher can lull the Mortals into thinking that there is nothing in the area, it will have an easier time attacking them. The major issue here is that passively waiting for opportunities wastes time and any time wasted brings the night closer to dawn and to a Mortal victory.

The statues positioned throughout the map can also block a Mortal player's movement. Fast-acting Watcher players, especially those with good teamwork, will be able to block off potential escape routes by lining up extra statues between existing physical barriers.

Many Watcher players may also try to boost their fear points and use the abilities of the additional classes. This is a viable strategy, but as most methods of intentionally boosting fear points involve staring down Mortals, it may once again be too slow.

### 2.2.6 Controls

Watchers inhabiting statues control very differently from their Mortal counterparts on the opposing team. The swapping and teleportation controls are identical for all forms of statue, but the ghost-form controls do vary between Watcher classes. With variations included, the default controls for all varieties of statue are shown in Table 2.

*Table 2 – Default Watcher Control Scheme*

| **Input (key or action)** | **Behavior** |
|---|---|
| **Teleport** | |
| Move mouse | Move teleport destination target within field of view, pan camera left/right |
| Left-click | Teleport if not in sight of Mortal; otherwise, lock on to the destination (or override the existing lock) |
| **Swap** | |
| Hold Right-Click | Opens the map while right-click is held. |
| Left-click over statue on map | Swap Watcher into selected statue |

| Basic Attack | |
|---|---|
| Press [Space] as statue | Initiate ghost-form attack if possible; switches to first-person view. This applies to all statue classes. The attack movement is automatic for the basic attack. |
| Move mouse | Perspective follows mouse (enables turning, looking up/down). This applies to all statue classes. |
| Pass through Mortal | Petrify the Mortal, converting him or her. This applies to all statue classes. |
| Pass through original statue | Return to statue; returns to third-person view. This applies to all classes. The return is automatic for the basic attack. |
| **Angel Attack** | |
| Press/hold [W][A][S][D] | Move forward, left, back, and right (respectively) directly relative to present perspective. |
| Press/hold [Space] | Move along the up-normal to the player's perspective. (Flap wings) |
| **Lunger Attack** | |
| Press/hold [A][D] | Shunt to the left or right, respectively. |
| Pass through Mortal | In addition to petrifying the Mortal, this also initiates the "homing" phase, automatically returning the Watcher to its origin statue. |
| Right-click mouse | Initiate homing statue return phase early. |
| **Digger Attack** | |
| Press/hold [W][A][S][D] | Move forward, left, back, and right (respectively) directly relative to present perspective. Not blocked by physical obstacles or terrain. |
| Press/hold [Space] | Move along the up-normal to the player's perspective, so long as the player is in contact with a solid obstacle. |

## 2.3 Balancing Issues

There are several behaviors and mechanics in this design that will be difficult to quantify until they can be seen against the backdrop of a functioning prototype of these game; these are largely issues of timing and quantity that will affect the game's balance. Here are some of the major unknowns that will need to be decided once the game reaches a playable state:

- **Distribution of statues –** This will be a major factor in the movement/tactics flexibility afforded the Watcher team. Too many statues, and Mortal players will be helplessly outgunned, completely unable to pick out the Watchers in their midst; too few, and the game will lack the proper atmosphere and limit the strategies available to the Watchers.
- **Distribution of specialist statues** – Similar to the placement of normal statues - it may be prudent to place specialist statues in particular locations (for instance, Angel statues in high vantage points). The number of specialist statues in the map will partially dictate how frequently they can be used (if there is only one Digger, it will be hard to get whole-map coverage).
- **Time until dawn –** This directly affects how difficult it is for the Mortals to survive. Too short, and some Mortals will survive because there simply will not be time for the Watchers to track them all down; too long, and the Watchers will be able to brute-force the remaining Mortals into submission.
- **Flash-stick cool-down –** This will affect the usefulness of the Mortals' flash-sticks; if the sticks are too frequently unresponsive, the Mortals may practically forget that they exist.
- **Fear point gain/loss rates –** This will impact the frequency and length of specialist statue appearances.
- **Statue return deadline –** This time value will yield an effective maximum range for attacking Watchers, and it will also decide the urgency of the potential fear point loss.
- **Movement speeds –** Watcher teleportation rates and Mortal walking/running speeds will determine the pace of the action; relative to each other, they will also determine how easy it is for Mortals to evade Watcher attacks.

## 3 Artistic Design and Vision

*[Omitted – this section is largely similar to the final version.]*

## 4 Technical Design

As *Petrified* is a game modification project, rather than a project developed from scratch, much of the architecture supporting the game is already in place. The technical challenge will come from understanding and extending that architecture to realize the features called for in the gameplay and artistic designs.

### 4.1 Engine Choice Rationale

This project will be based on the Source Engine from Valve, Inc.; specifically, it will be developed as a total conversion mod based on *Half Life 2 Deathmatch*. The engine has been chosen for the following reasons:

- The Source Engine was originally designed with a client-server architecture, and has served as the basis for first-person multiplayer games ranging from *Counterstrike Source* to *Team Fortress 2*. It supports a large number of players – the PC version of *Team Fortress 2*, for instance, supports 24 players by default, though the engine can support up to 32.
- The most recent version of the Source Engine (the one included with the popular Orange Box bundle) supports high-quality dynamic lighting that could be applied for most of our lighting needs.
- Source and Hammer, the associated level/map creation tool, both integrate well with common modeling tools like 3DSMax. The SDK also includes model-viewers and other helpful artistic tools.
- Source has supported a variety of particle effects since its inception (including particulate fog used for atmosphere). More recent versions also support true motion-blur effects and heat distortion effects.
- Source supports an adjustable field-of-view, which will be important in maintaining the visual-related gameplay mechanisms (such as the Mortal's ability to freeze Watchers) across configurations with different monitor/resolution settings.

- The Source engine can support large, open outdoor areas, as demonstrated in some scenes from *Half-Life 2: Episode 2,* which utilizes the most recent commercial build of the engine.

## 4.2 Specifications

These specifications are performance and feature targets; if game performance does not reach the level specified here, the visual and technical elements of the game (such as polygon counts or network packet size) will need to be reduced in complexity. The specifications themselves will only be adjusted if they are technologically infeasible or will in some way broadly compromise the gameplay or artistic design.

### 4.2.1 Hardware Requirements

According to Valve's website (steampowered.com), these are the minimum and recommended system hardware requirements for their most recent multiplayer Source Engine-based game, Team Fortress 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

*Petrified* will not utilize visual effects that are as intensive as those found in *Team Fortress 2* (for instance, a soft cartoon-shading pixel shader that is applied to all models), so it may be able to run comfortably on configurations closer to the stated minimum requirements. The requirements do not specify what level of Internet connection is necessary, however. Some of *Petrified*'s visual illusions (particularly the illusion that the statues remain stationary for the Mortals) will be difficult to maintain over a high-latency connection. Therefore, as an additional requirement, the network connection must be a low-latency broadband connection or a direct connection via local area network.

### 4.2.2 Software Performance Targets

*Petrified* will meet the following performance targets:

- Players: 16-player maximum

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

- Latency: Design for less than 100ms ping time, to maintain the illusion that the statues never move.

- Voice chat supports every player talking at once, but it is expected that at most 4 will be talking at one time (2 players simultaneous/team).

## 4.3 Software Architecture

The players and game mechanics of *Petrified* can be formally specified in the form of UML objects; each in-game entity (such as a Mortal player or a Watcher ghost) can be described in terms of its attributes and the methods that it uses to interact with the game world. These objects can be abstracted into interfaces to show the similarities and connections between them; in the implementation phase, these objects will translate directly into C++ classes that slot into the Source Engine.

The Source engine already has a complex architecture with which the team is not intimately familiar, so some of the behaviors and attributes specified in the UML diagrams may overlap with existing Source classes. Still, many of the behaviors are virtually guaranteed to be specific to the needs of the *Petrified* design. The UML diagrams cover *Petrified's* overarching rules, specific player types and behaviors for both teams, additional objects associated with the players (such as the mortals' flash-sticks), and objects that interface the players and rules with the physics, graphics, and sound modules of the Source engine. The team will attempt to implement the UML as written, integrating the resultant objects with the Source classes and extending the Source classes where possible to avoid duplicating functionality.

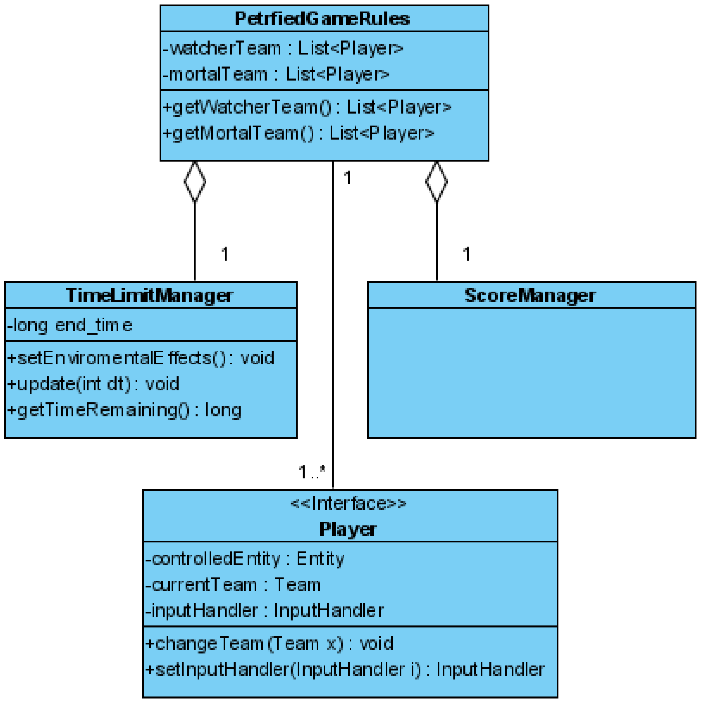
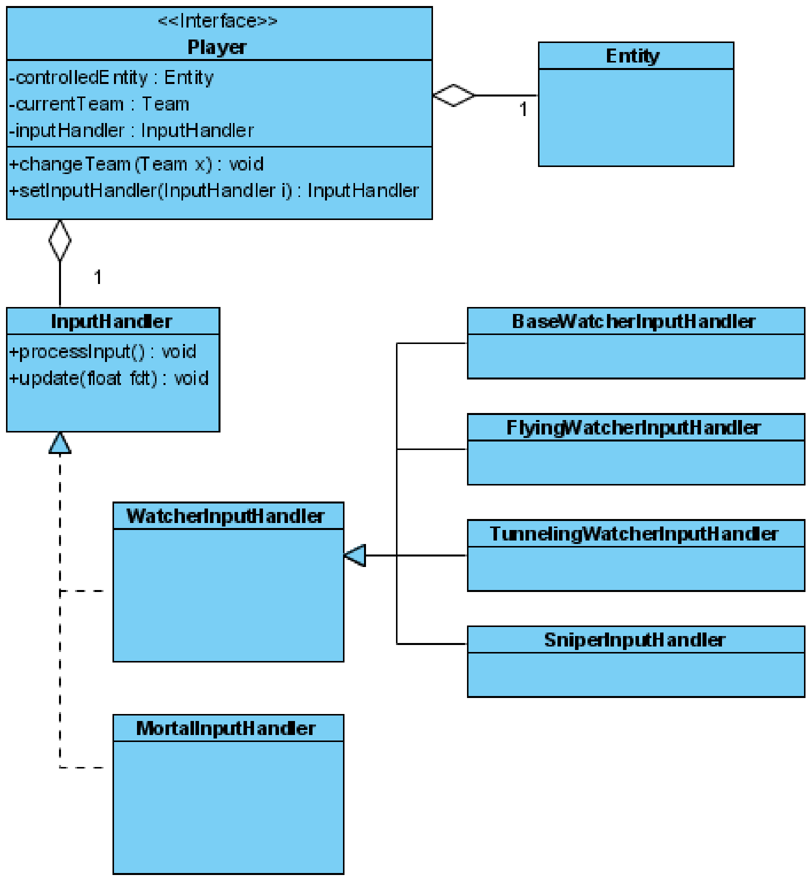

PetrfiedGameRules
-watcherTeam : List<Player>
-mortalTeam : List<Player>
+getWatcherTeam() : List<Player>
+getMortalTeam() : List<Player>
1
1
1
TimeLimitManager
-long end_time
+setEnviromentalEffects() : void
+update(int dt) : void
+getTimeRemaining() : long
ScoreManager
1..*
<<Interface>>
Player
-controlledEntity : Entity
-currentTeam : Team
-inputHandler : InputHandler
+changeTeam(Team x) : void
+setInputHandler(InputHandler i) : InputHandler

*Figure 4 – GameRules UML*

The overarching object that will handle the gameplay mechanics of *Petrified* will be the **PetrifiedGameRules** class (see Figure 4). This class will keep track of all the Players and which team they are on. It will also have a reference to a **TimeLimitManager** which will keep track of the time remaining before dawn and handle the environmental cues. When the time limit is reached, it will notify the game rules object which will then take the appropriate action.



<<Interface>>
Player
-controlledEntity : Entity
-currentTeam : Team
-inputHandler : InputHandler
+changeTeam(Team x) : void
+setInputHandler(InputHandler i) : InputHandler
Entity
1
1
InputHandler
+processInput() : void
+update(float fdt) : void
BaseWatcherInputHandler
FlyingWatcherInputHandler
WatcherInputHandler
TunnelingWatcherInputHandler
SniperInputHandler
MortalInputHandler

*Figure 5 – Player UML*

The **Player** class, as seen in Figure 5, will be the class that handles all the aspects of a player, either by doing it itself or delegating. Each player will have an **Entity** which is the visual representation of this player in the world. The entity the player is controlling will be able to changed at run-time, for example when the player switches teams or goes into ghost form the entity would need to change to reflect that. Each player will also have an **InputHandler** which can also be switched at run-time. This will make it far easier to implement the various differences in control schemes for the Watcher classes, as well as the differences between those and a Mortal's.

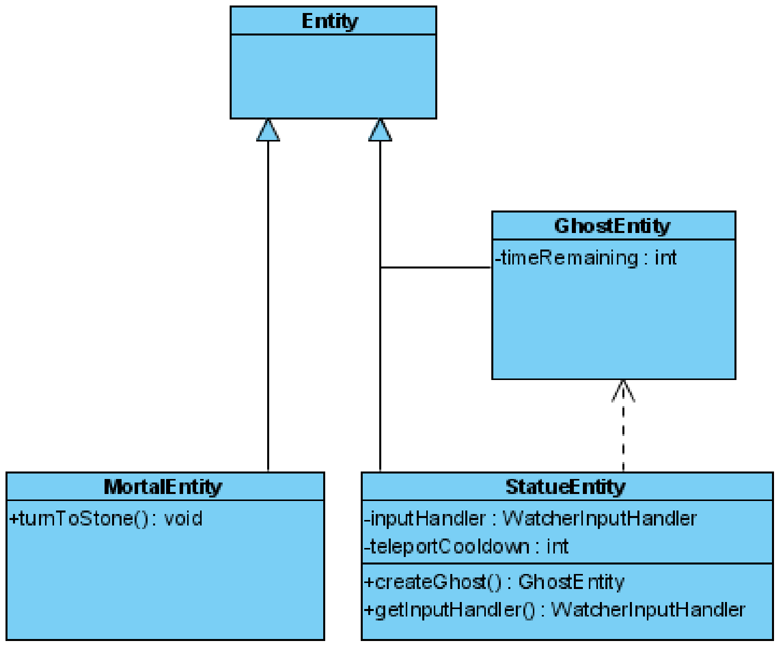


*Figure 6 – Entity UML*

The **Entity** class is a class that already exists in the *Source* engine's framework. The team will be extending it into a **MortalEntity**, **StatueEntity**, and **GhostEntity** (see Figure 6). This will allow for some easier implementation of the features we have planned. For example, by storing an **InputHandler** and a **GhostEntity** in a **StatueEntity** we ensure that ever statue has their own ghost, and provides a very easy way for the **Player** class to get the appropriate **InputHandler** when a Watcher player swaps into a different statue.

## 4.4 Technical tools

The Source SDK integrates with *Microsoft's Visual Studio* line of Integrated Development Environments (IDEs). As a result, this project will be developed with *Microsoft Visual C++ 2005*, as this version has the most support information in the *Valve Developer Community's* Wiki.

The project will also utilize a version control system, such as *Subversion* (SVN), which is available through WPI's http://sourceforge.wpi.edu site. This will enable the two developers to avoid creating conflicting code and it will also allow the project to revert the code base to earlier revisions if new updates prove unstable or otherwise undesirable. As *Subversion* does not integrate with *Visual Studio* by itself, the team will need to use an additional tool, such as *TurtleSVN*, to commit changes and manage the code repository.

For non-code resources, such as concept art, design documents, and website materials, the team has decided to, and has used *Dropbox*.  *Dropbox* is a file synchronizing system that allows several users to share a folder with each other over the internet; it integrates directly with the *Windows* operating system, and it allows for version control through its web interface.