The Tech of *Ourea* - Contribution Report

by

Samuel Hegner

Supervisor: John. P. Healy

School of Media

TU Dublin

A report submitted in partial fulfilment of the requirements for the degree of
Bachelor of Arts (Honours) in Game Design

May 2019/20

Declaration

I certify that this report which I now submit for examination in partial fulfilment for the award of Bachelor of Arts (Honours) in Game Design, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within this report and, where applicable, the work itself.

This report was prepared according to the general assessment regulations of TU Dublin and has not been submitted in whole or in part for another award in any other third-level institution.

Signature Samuel Hegner   Date   15/05/2020

CONTENTS

INTRODUCTION

This document has been written to provide an overview of my role as Technical Lead on the game "*Ourea*". To allow for an accurate and overview of my contribution, the document has been organised in the following way:

To start, a brief background on myself and the preproduction phase of *"Ourea"* will be outlined to help provide context on the content focused on the game's development. Previous academic modules, team formation, project goals, personal roles and preproduction research will be discussed. Next, my contributions to the game's development will be discussed in the Methodology and Approach section. The section is divided into three sub-sections to chronologically set out contributions over the three stages of development: Vertical Slice, Alpha, Beta and Gold. Examples of work will be provided and evaluated. This is followed by the Evaluation section. The Evaluation section allows for reflection on the overall project and my contributions to it. Lastly, the document will be concluded.

BACKGROUND

Previous Experience

In terms of previous experience, I have been focusing on Tech/Coding since the start of the first year of college. I have taken tech modules where possible over the last 4 years and have worked tech on almost all projects I have been a part of.

Team Formation

The team for Ourea was formed at the end of the third year, second semester and over the following summer. Every member I have worked with previously and the experiences have been excellent. We decided on five members because it allowed for Ourea's scope to be on the large side whilst being achievable in the limited time available.

Project Goals

The goal of Ourea for me personally is to create an impressive portfolio piece. The team size meant that compromises had to be made by everyone on elements of the game (genre, style, gameplay, etc.) so everyone could achieve the same goal. On my part, even though the game isn't tech-heavy, I wanted to still be able to show my abilities through it.

Preproduction

Throughout the preproduction process of Ourea, my role boiled down to researching workflows of Unity and Coding practices. Most of the research was done on Unity's new HD Render Pipeline, multi-scene editing and additive async loading. Those three elements were and still are crucial to Ourea being what it is today. In hindsight, choosing HDRP for the project was a risky decision, due to its novelty and lack of documentation. However, the research paid off as the whole team now feels comfortable within the system.

METHODOLOGY AND APPROACH

Introduction

To breakdown my contributions in a chronological way, it has been split into the three development stages.

Vertical Slice

Gameplay
I created all the gameplay necessary within the VS. This included player scripts (movement, dynamic look at, interaction script), interactable such as the torch and the boulder and implemented gameplay for the fireflies.

HDRP Configuration
I learned to configure and implemented HDRP assets, lighting settings, post-processing, volumetrics and things like 3d texture for local fog volumes.

Alpha

Scene Loading
The aspect of tech that took up the most time was researching and implementing an additive level loading solution that was expansion proof for the full game. Ourea's vertical slice implemented a multi-scene approach, however, because of the small size of the project at the time, all scenes could be loaded in at the start of the game. Again, at the end of the alpha phase, since the game was only white boxed from the vertical slice onwards, the white box area was loaded in as one scene. However, we knew that that was a temporary fix and the full scene loading system was going to be implemented for the beta phase.

Puzzle Implementation
The puzzle Implementation took a back seat this phase with only half of the second puzzle being implemented. The vertical slice player interaction implementation left a lot to be desired and needed frequent bug fixing. It was tough to expand an already broken system, quickly. However, each fix meant that the system was more future proof. The Interface approach I decided on during Vertical Slice, however, was a good solution. It meant that each interactable would have to implement certain functions (StartInteraction() and StopInteraction()) but they could implement them however they wanted.

Beta

Scene Loading

This version of the game had the full multi-scene, additive loading implementation working. The game was split into 34 separate scenes. The approach of loading in scenes at runtime was splitting the game into a manager scene, a base scene and the rest of the game into section bundles that could b easily loaded in and out.

The manager scene gets loaded in first. It then loads the rest of the game starting with the base scene and the section bundles required for where the player left off. This way of loading the game meant that I had to create debugging and playtesting tools to allow for easy testing of sections without having to play from the start. These tools came in useful in the gold phase as they could be repurposed as an easy way to implement the level save system.

- First Loaded
- Contains Managers
- Controls the game state

Manager Scene

- Loaded Second
- Contains Persistent Objects
- Player
- Main Camera
- Global Lighting
- Global Settings and Post processing

Base Scene

- Loaded In when needed
- Unloaded when not needed
- Contain all section specific objects

Section Bundle

Section Assets
- Art Assets
- Section Specific Logic
- Section Load and Unload Triggers

Section Cameras
- Cinemachine Virtual Cameras
- Cutscene Timelines
- Cutscene specific objects
- Occlusion Culling Zones

Section Lights
- Section Lighting
- Local post processing overrides

Puzzle Implementation

All puzzles within Ourea needed to be playable to a good standard for the beta. This meant that interactions like plank placing, broken growth rune, seesaw, lightning mechanics, growth rune rework and all other puzzle mechanics I created in this phase.

Gold

Performance

To increase the performance of the game for submission, a lot of research and work had to be done. I implemented Occlusion culling for the entire game, lighting was optimized and rendering draw-calls and set-passes where reduced were possible. This was done using static and instance batching.

Scene Loading

The final version of the level loading worked in the following ways. A scene bundle asset that contains references to its Assets, Camera and Lights scene gets used to async load those scenes.

Then the three scenes are merged to allow for easier unloading. The biggest change in the system in gold was to spread the loading of the three async operations out to reduce the frame drop.

```csharp
for (int i = 0; i < instance.currentBundle.scenes.Length; i++)
{
    var asyncOp = SceneManager.LoadSceneAsync(instance.currentBundle.scenes[i], LoadSceneMode.Additive);

    SceneManager.sceneLoaded += instance.MergeScenes;

    asyncOp.allowSceneActivation = false; //< Deactivate the load of gameobjects on scene load

    if (asyncOp != null)
    {
        while (!asyncOp.isDone)
        {
            // Check if the progress is less than 0.9 (if it's less, it means that we load gameobjects)
            // Else, it means that we load something else.
            if (asyncOp.progress >= 0.9f && !asyncOp.allowSceneActivation)
            {
                asyncOp.allowSceneActivation = true; //< Once everything is loaded, reactive this variable
            }
            else
            {
                // Scene is activated
                yield return null; //< We still wait until the scene load is finished
                //Then the cycle repeats for the remaining scenes in the bundle
            }
        }
    }
}
```

Puzzles and Player Mechanics

In this phase fixes to player movement and interactions were made to improve player feels. Movement on slopes was improved and interactions were sped up and made crisper. The puzzle mechanics were improved to work smoothly without hitches. An example of this was the squish rune that had issues with the player picking up the item within it.

I converted the paintings into player interactable paintings. In this phase, I also improved the procedural hand placement of Glyf which meant it could be used on items such as the growth rune and the interactable paintings.

Implementation of Assets

Over the gold process, the role of tech pivoted to heavily requiring the implementation of art, Vfx and other polish changes that required Unity implementation.

UI

I created all the Ui functionality. Most importantly the level selection, settings and the painting gallery.

Save System

I created a save system that loads the player in where they last left off. This was done through a binary encoder for security. This is integrated into the level select in the Ui. I also added saving of settings and audio levels through player prefs.


```csharp
[System.Serializable]
17 usages    Samuel Hegner    1 exposing API
public class ProgressionData
{
    //which section of the game is the player on
    public int saveSectionIndex;    Serializable

    //is it night time in the current section
    public bool nightTime;    Serializable

    Frequently called    5 usages    Samuel Hegner
    public ProgressionData(int tempSaveSectionIndex, bool tempNightTime)
    {
        saveSectionIndex = tempSaveSectionIndex;
        nightTime = tempNightTime;
    }
}
```
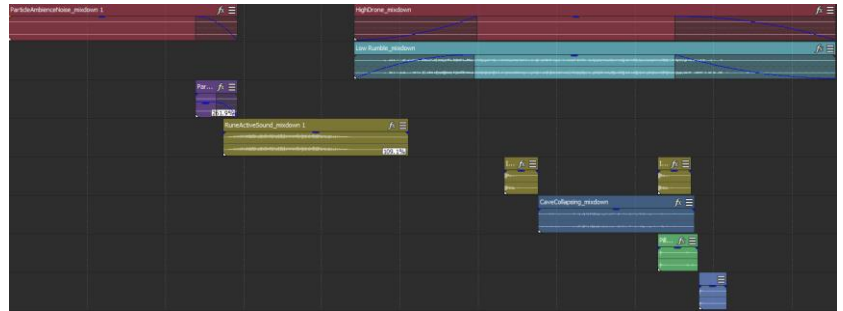
Additional Contributions

Audio

In terms of audio, I created all sounds (excluding music) in the game. The process was sourcing multiple sounds on FreeSounds.org, editing them heavily and combining multiple sounds in Audition and Audacity. This also included creating audio for the various cutscene throughout the game.

Lighting

I created the lighting for all areas of the game. Each section was reworked multiple times throughout the phases of the project. Lighting also included creating local density volumes to accentuate volumetric rays.

Marketing

Throughout the project, I created many Gifs for the game's social media channels. Theses were created using photoshop. I also created the two trailers for the game. These were made using Sony Vegas.

EVALUATION

Contribution Impact

My contributions to this project meant that the game runs from start to finish. I would like to think that I had a significant impact on the quality of the game. The code written over the past year, even though it is messy, works well. I think that the fact that I have had previous experience with tech meant that it opened doors in terms of scope and possibilities. This can be said for every single team member. In my mind, I was going to be happy with the project from the moment the team was formed.

Goals and Takeaways

In terms of the goals set out for the project, I am delighted with what we achieved collaboratively and what I learned and achieved individually. I loved working on a project with a larger scope. It introduced interesting technical challenges. I am proud of the game and the roughly 170 scripts within the project, even though in hindsight they could have been done differently. I have never learned more on a single project. It has also given me confidence in looking for work in the future. I think the most important takeaway in terms of coding is making sure all code is future proof. The number of iteration scripts went through, even scripts that I thought would not need future editing, shows that projects are unpredictable. Futureproofing scripts saves time in the long run, even if it seems tedious at the time. In terms of a general takeaway, working on Ourea with a team of 5 equally motivated and passionate people meant that even though the project scope was ambitious, it was manageable. It meant that everyone could rely on one another and communication was constant. Even one weak link would have meant Ourea would never have been where it is today.

CONCLUSION

This report briefly discussed the background of Ourea, how I approached the role of Tech and what I did and lastly a short evaluation of the project. Overall, the project was a success based on the set goals being achieved. With such a short report, it is impossible to talk about any part of my contribution in even slight detail, but hopefully, it provided a rough overview.