UFCFS4-30-3 Creative Technologies Project Proposal Document	
Student Name:	Alexander Allman
Student Number:	16015338
Project Title:	Creating a system based on a Data-Oriented paradigm that compliments an Object-Oriented workflow.

1 DESCRIPTION

Deliver a system/library which:

- Has a live custom memory allocator for storing related data in contiguious regions of memory
- A system which can schedule transmutations on specific data
    - Leveraging threading for increased performance.

Data-Oriented design has many perks when it comes to programming in comparison to Object-Oriented programming (Fabian, 2018). Unlike an Object-Oriented paradigm, classes are broken down into individual sets of data and these sets are stored in their own regions of memory. Since all the data stored in a set will be transmuted in the same way it is easy to parallelise and it is quicker due to not bloating cache lines with unrelated data (Llopis, 2009).

The downside, however, is that legibility of code is severely hindered. This is due to a single entity is made up of single entries in multiple arrays as opposed to being a single object. This makes it much harder to visualise a single entity because it is no longer just a single object but an amalgamation of loosely coupled sets of data. This results in the developer needing a shift of mindset when programming in a Data-Oriented manner (Llopis, 2009).

The aim of this project is to create a system in which a developer familiar with an Object-Oriented approach can still gain the benefits of a Data-Oriented codebase. This will be done through an interface in which an Object-Oriented developer can use to create variables and process changes on said data. This interface will manage memory allocation using a custom memory allocator which will be accompanied with a threaded system that allows transmutation of this data. The custom memory allocator will store data within the same domain using single blocks of memory, similar to the sets of data previously described. The system will also choose appropriate times to defrag memory, e.g. when the CPU has spare cycles or in between loading game scenes.

2 RESEARCH AND BACKGROUND

The primary inspiration for this project is seeing the work being done with the new Data-Oriented tech stack (D.O.T.S) that unity has implemented into its engine (Meijer, 2019).

Exploring Richard Fabian's (2018) book on Data-Oriented design explains the core concepts of what it takes to follow a Data-Oriented paradigm. It focuses on leaving the problem domain out of the codebase. Which removes the performance overhead that is involved with abstracting the data into more human legible objects (Fabian, 2018).

One of the main ways this is achieved is with Structures of Arrays (SoA) as opposed to Arrays of Structs (AoS). The difference between these two is how the data is stored. For example, AoS follows a more traditional Object-Oriented paradigm where the developer would have a single array storing individual instances of classes, such as a vector of enemies. This opposes the SoA approach where there would be a struct containing multiple arrays and, in each array, the same data being stored together. Processing transmutations on data ends up being much quicker in an SoA approach because the data is stored in contiguious memory and will also be transmuted in the same way. This results in only relevant data being pulled into cache and very fast transmutations on the data (Figure 1 & Savas, 2017). In contrast in an AoS setup if we wanted to process a single piece of data that belonged to the struct, the whole struct would be pulled into cache, filling the cache with data that isn't relevant to the operations being performed and leading to more cache misses (Figure 2 & Savas, 2017). By filling each cache line with only data that is being processed it saves significant time having to move to progressively slower caches to retrieve the data been processed.

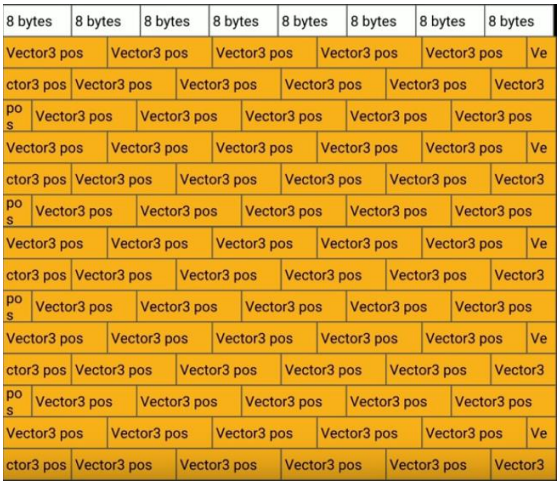


Figure 1 - A visualisation of a SoA approach where all cache lines are only filled with the data being processed

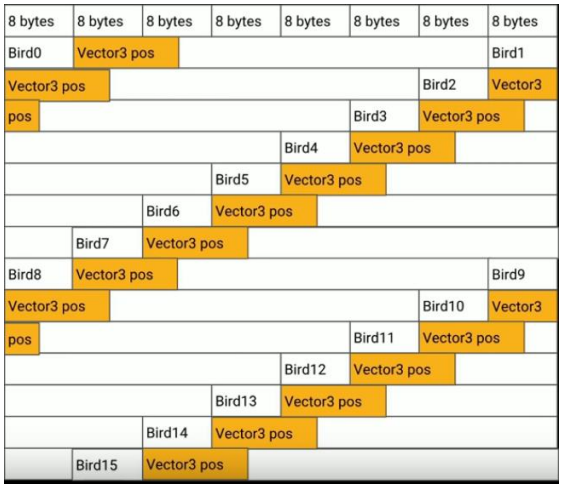


Figure 2 - A visualisation of an AoS approach where cache lines are being filled with irrelevant data to the position being processed.

Another large perk of using the SoA approach in Data-Oriented design means the data we are processing is very parallelisable because in most cases it will all be transformed in the same way (Fabian, 2018). This means you can set off a worker thread very easily that iterates over a set of data applying the same transmutation to each element of the set. If there was data that didn't follow this rule or needed more than one change per update; the idea is that those elements would be stored in another array separate to the original one and then that one would be iterated upon once the previous thread had finished updating the original one.

With this knowledge of the Data-Oriented design paradigm I will make a system in which the user will give unique types to variables in their classes which will be recognised and stored together using a custom memory allocator. This system would have the ability to schedule jobs on the data through the user specifying the data they want changed and a function pointer that contains the logic on how to change the data. This would allow the user to keep using an Object-Oriented approach to development but utilising some of the performance benefit that a Data-Oriented approach would grant you.

3 OBJECTIVES

3.1.1 Project objectives

- Create a custom memory allocator.
- Create a system to allow threaded processing on the memory stored by the allocator in a Data Oriented Manner Utilising SoA's.
- Profile differences between using the resultant system, the standard memory allocator and the more traditional AoS approach to processing the stored data.
- Test different types of memory allocators to see which would fit best with the system.
- Look into compression of data in conjunction with the memory allocator to leverage computers with powerful CPUs but less RAM.

3.1.2 Research objectives

- How to program memory allocators.
- How to use industry standard CPU profilers.
- How to thread workloads and reduce threading overheads e.g. thread pooling.

3.1.3 Learning Objectives

- Understand the use of memory allocators.
- Industry grade profiling tools and applications.
- Furthering my knowledge of parallelising tasks through threading.

4 METHODS, TECHNIQUES, TOOLS AND PROCESSES

For the creation of the project I plan to leverage the DirectX Framework. The main reason for this is having access to the full feature set of the C++ language. DirectX also has the perk of many libraries supporting it which can be leveraged to take away tasks that are not integral to the project. These could involve things such as input and math calculations for rendering or even a plugin which could be used to plot graphics using the data fed to it by the project, allowing for real time benchmarking.

Secondly, for backing up and version control of the project, GitHub will be used. I will utilise the Git Flow methodology in which new tasks are started in separate branches and are only merged into the develop branch when its been tested to work. On completing a milestone task, I shall then merge into the master branch.

Finally, for project management an Agile approach will be utilised. Larger tasks will be marked as Milestones and broken down into smaller one- or two-week sprints, depending on how much time is required. Through using GitHub Issues I can have a constant list of tasks to do and build a Gantt chart from it.

5 RISKS AND ISSUES

Risk	Mitigation	Contingency
Chosen memory allocator isn't fit for purpose.	Have a few options of different allocators that can be used if desired one isn't fit for purpose.	Use malloc for memory assignments instead and focus on the transforms of data as opposed to the allocation of it.
Failed to create a simple and user-friendly interface for Scheduling tasks on data.	Research existing systems which utilise Job systems and scheduling work on data through a threaded process.	Add a handle to the data stored so the developer can manage their own threads and process the data outside of the system.
Cannot get data visualisation API to work in DirectX.	Have a variety of usable APIs on had so that if one doesn't work more can be attempted.	Write out the data to a text file and import it to excel for visualisations.
Cannot get compression working with existing memory allocators	Research in depth compression algorithms beforehand	Stretch goal, Not vital to the core project.

6 SPECIALIST RESOURCES AND SUPPORT REQUIRED

N/A

7 SOURCES AND REFERENCES

1. Fabian, R. (2018) Data-oriented Design: Software Engineering For Limited Resources and Short Schedules. United Kingdom: Richard Fabian.

2. Llopis, N. (2009) Data-Oriented Design (Or Why You Might Be Shooting Yourself in The Foot With OOP). Game tech [blog]. 04 December. Available from: http://gamesfromwithin.com/data-oriented-design [Accessed 05 October 2019].

3. Meijer, L. (2019) On DOTS: C++ & C#. Unity Blog [blog]. 26 February. Available from: https://blogs.unity3d.com/2019/02/26/on-dots-c-c/ [Accessed 05 October 2019].

4. Savas, N. (2017) Part 4.3 — AoS vs SoA. Nomad Game Engine [blog]. 28 February. Available from: https://medium.com/@savas/nomad-game-engine-part-4-3-aos-vs-soa-storage-5bec879aa38c [Accessed 05 October 2019].

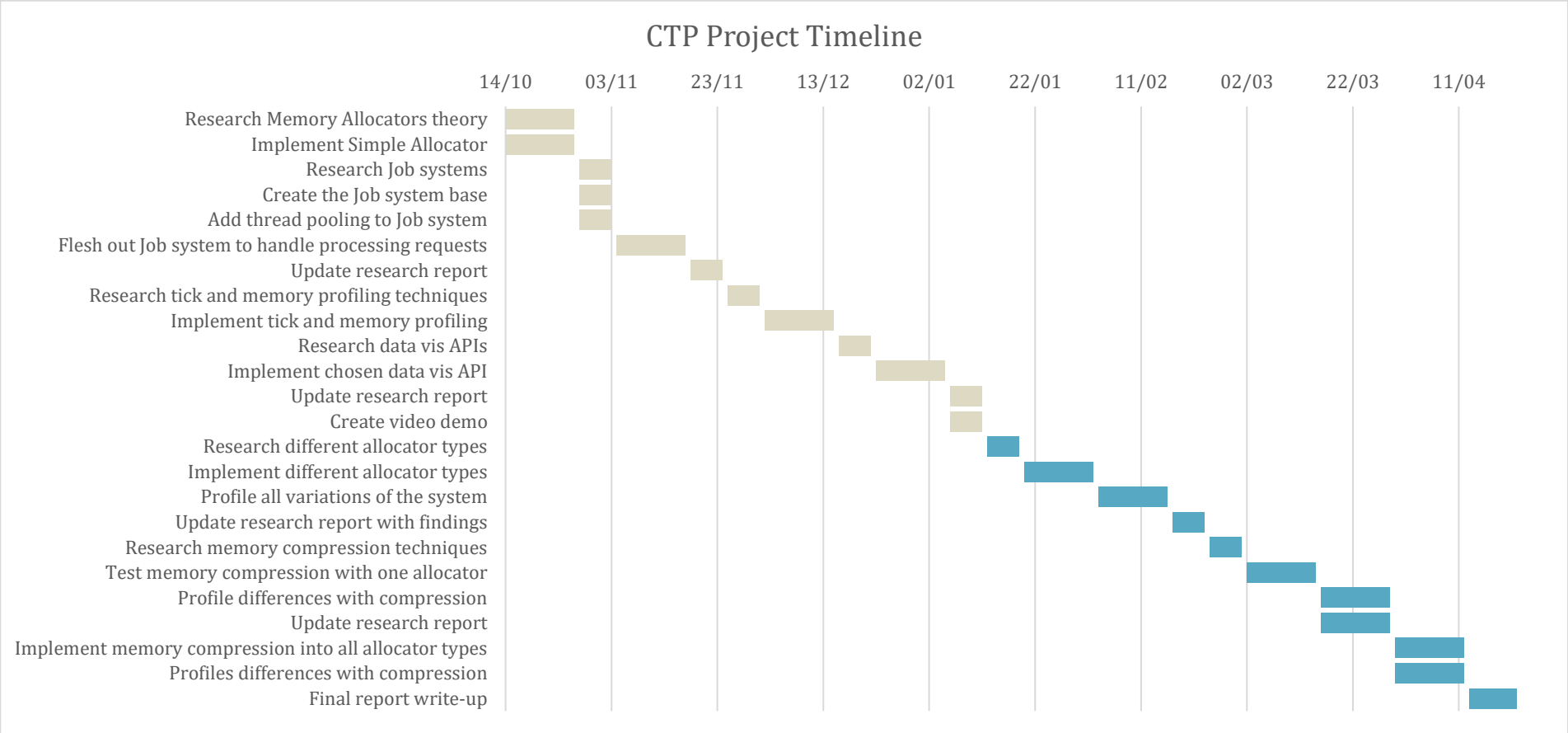
5. Trebino, M. (2019) Memory-Allocator explanations [GitHub]. 30 July. Available from: https://github.com/mtrebi/memory-allocators/blob/master/README.md [Accessed 05 October 2019].

6. Trebino, M. (2019) Thread Pooling explanation [GitHub]. 19 February. Available from: https://github.com/mtrebi/thread-pool/blob/master/README.md [Accessed 05 October 2019].

7. Figure 1 and 2 - Unity (2019). Understanding data-oriented design for entity component systems - Unity at GDC 2019. YouTube [video]. 9 April. Available from: https://youtu.be/0_Byw9UMn9g [Accessed 05 October 2019].

8 MONTHLY PROJECT PLAN

The project will be completed in either weekly or fortnightly sprints and it is split into to large milestones; the lead up to the demo in January and the lead up to the final deadline. Below is the Gantt chart for the project:



Ethical Review Checklist for Undergraduate and Postgraduate Modules

Please provide project details and complete the checklist below.

Project Details:

Module name	**Creative Technologies Project**
Module code	**UFCFS4-30-3**
Module leader	Michaela Palmer
Project Supervisor	**James Huxtable**
Proposed project title	Creating a system based on a Data-Oriented paradigm that compliments an Object-Oriented workflow.

Applicant Details:

Name of Student	Alexander Allman
Student Number	16015338
Student's email address	Alexander2.Allman@live.uwe.ac.uk

	CHECKLIST QUESTIONS	Yes/No	Explanation
1.	Does the proposed project involve **human tissue, human participants, animals, environmental damage, or the NHS.**	No	*If the answer to this is 'No' then no further checks in the list need to be considered.*
2.	Will participants be clearly asked to give consent to take part in the research and informed about how data collected in the research will be used?		
3.	If they choose, can a participant withdraw at any time (prior to a point of "no return" in the use of their data)? Are they told this?		
4.	Are measures in place to provide confidentiality for participants and ensure secure management and disposal of data collected from them?		

	CHECKLIST QUESTIONS	Yes/No	Explanation
5.	Does the study involve people who are particularly vulnerable or unable to give informed consent (eg, children or people with learning difficulties)?		
6.	Could your research cause stress, physical or psychological harm to humans or animals, or environmental damage?		
7.	Could any aspects of the research lead to unethical behaviour by participants or researchers (eg, invasion of privacy, deceit, coercion, fraud, abuse)?		
8.	Does the research involve the NHS or collection or storage of human tissue (includes anything containing human cells, such as saliva and urine)?		

Your explanations should indicate briefly for Qs 2-4 how these requirements will be met, and for Qs 5-8 what the pertinent concerns are.

- **Minimal Risk:** If **Q 1 is answered 'No'**, then no ethics approval is needed.
- **Low Risk:** If **Qs 2-4 are answered 'Yes' and Qs 5-8 are answered 'No'**, then no approval is needed from the *Faculty Research Ethics Committee* (FREC). However, your supervisor must approve (a) your information and consent forms (Qs 2 & 3) and (b) your measures for participant confidentiality and secure data management (Q4).
- **High Risk:** If **any of Qs 5-8 are answered 'Yes'**, then you must submit an application for full ethics approval *before* the project can start. This can take up to 6 weeks. Consult your supervisor about how to apply for full ethics approval.

Risk Assessment: Separate guidance on risk assessment can be found on UWE's Health and Safety forms webpage at https://go.uwe.ac.uk/RiskAssessment. If needed, you must complete a Risk Assessment form. This must also be attached to your application for full ethics approval if your project is **High Risk**.

Your supervisor must check your responses above *before* you submit this form.
Submit this completed form via the *Assignments* area in Blackboard (or elsewhere if so directed by the module leader or your supervisor).
After you have uploaded this form, your supervisor will confirm it has been correctly completed by "marking" it as *Passed*/100% via the *My Grades* link on the Blackboard.

Further research ethics guidance is available at
http://www1.uwe.ac.uk/research/researchethics