Liverpool John Moores University

Title:	GAMES ENGINE ARCHITECTURES
Status:	Definitive
Code:	**6105COMP** (121263)
Version Start Date:	01-08-2021
Owning School/Faculty:	Computer Science and Mathematics
Teaching School/Faculty:	Computer Science and Mathematics

Team	Leader
Chris Carter	Y
Sud Sudirman	
Abdennour El-Rhalibi	

Academic Level:	FHEQ6	**Credit Value:**	20	**Total Delivered Hours:**	55
Total Learning Hours:	200	**Private Study:**	145		

Delivery Options
Course typically offered: Semester 1

Component	Contact Hours
Workshop	55

Grading Basis: 40 %

Assessment Details

Category	Short Description	Description	Weighting (%)	Exam Duration
Artefacts	AS1	Algorithmic Game Implementation within a game engine architecture.	100	

Aims

To compare and contrast architectural approaches to building a game engine architecture.
To gain an understanding of the importance of abstraction, decoupling and encapsulation within a game engine environment.
To understand how data structures and algorithms are formally applied to
To understand how the object oriented and data oriented paradigms are applied in

an engine environment.
To understand how data-driven architectures can be used to abstract both content and behaviour in an engine environment.
To identify, formulate and apply solutions to a diverse range of advanced computer game problems across the architectural tiers.
To understand and use the structures and technologies of modern game engines.
To present advanced game programming techniques and technologies applicable to game development.
To implement key algorithms for optimising 3D scenes, organising game mechanics and structures.
To be able to integrate and use middleware libraries to solve domain-specific challenges in games (e.g. Physics, AI, Shaders, Maths, Procedural Generation).

Learning Outcomes

After completing the module the student should be able to:

1	Implement game techniques using modern game engine methodologies and data structures.
2	Apply a set of middleware and APIs to implement a particular aspect of computer games development within a game engine.
3	Apply suitable object- and data-oriented programming techniques and game technologies to solve problems specific to gaming applications.
4	Explain the core algorithms and evaluate the technologies that can be applied to a module within the tiers of a modern game engine.
5	Use and extend an existing game engine to incorporate new features at both game-play and game engine level of abstraction.

Learning Outcomes of Assessments

The assessment item list is assessed via the learning outcomes listed:

Game Implementation	1	2	3	4	5

Outline Syllabus

- Core C++ API usage, interaction event-driven software.
- Algorithms and data structures for game development.
- Containers, Iterators and their usage in Game Engines.
- Metadata, C++ Pre-Processor and Meta-Programming.
- Poly-Soup vs Hierarchical Scene Structures.
- Single vs. Multi-Threaded Engine Design.
- Monolithic vs Adaptive Module Design.
- Scene Hierarchy and Node, Graph and Tree-Based Engine Design.
- Scalar, Vector and Matrix Representation and Transform Propagation.
- Optimisation Techniques
- Platform Independence: Endianness, Fundamental Types, Memory Management, Memory Hierarchy, Templates, Generics, Cache Coherence, Consideration for Console Development.

- Visibility Determination: Potentially Visible Sets, Frustum Culling
- Spatial Data Structures: Quad Tree, BSP, Octree, Portals and Anti-Portals. Indoor vs. Outdoor Scene Design.
- Performance – Profiling, Prediction and Monitoring.
- Broad and Narrow Phase Collision Detection and Response.
- Collision Data Representation – Artist and Procedural Definition.
- Physics, Artificial Intelligence and Rendering Middleware Usage.
- Material Systems and CPU to GPU Parameterisation.
- Pre-Computed Lighting in a Game Engine – Using an Offline Renderer.
- Asset Management, Asset Conditioning and Data Import/Export.
- Game Editors, UI Tools and Command Line Interfaces. .
- Data-Driven Game Development
- Object-Oriented vs Data Oriented Programming Paradigms.
- Game Engine Architecture Design.
- Animation, Motion and Dynamics within a Game Engine.
- Inheritance vs. Component Based Models of Game Mechanics and Objects.
- Game Engine Architecture Tiers:
- 3RD Party SDKs; Core Systems; Platform Independence (Desktop vs. Console); Rendering, Physics and AI Subsystems; Game Play Foundations; Visual Effects System; Game-Specific Subsystems.
- Scripted Events vs. Procedural Dynamics.
- Soft vs. Hard Architectures: Scripting Languages
- Feedback Models: User Interfaces and Orthographic/Perspective Interface

Learning Activities

Workshop – to deliver the theoretical concepts on game engine architectures and tutor-led practical session in the computer laboratory.
Further exercises – additional exercises for students to work on in their own time.
Directed learning – provides additional reading to enable workshop work to be completed.
Learning materials can be accessed digitally via University Virtual Learning Environment (VLE).

Notes

This module will cover the software engineering principles and core algorithms which are used to implement a full-scale game engine. We will focus on the architectural layers of a modern game engine from core system and language/platform fundamental, through rendering, physics, AI and material representation towards the game play foundations of the games which are built on top the engine. We will look at various architectural design strategies and provide in-depth coverage of various core modules within a game engine and how they relate to the other domain of study computer games development students have covered. This is involved advanced programming techniques in C++ and how they are practically applied to construct a game engine. Students will take on different technical role in team setup to design and implement a game within a game engine architecture.