Introduction to Game Engines
(SENG 463 - Game Programming)

Dr.Çağatay ÜNDEĞER
Research and Innovation Director
SimBT Inc.

e-mail : cagatay.undeger@simbt.com.tr
cagatay@undeger.com

Game Engine Reference
Roger Mailler, Ph.D.,
Associate Professor of Computer Science, University of Tulsa
https://www.gameenginebook.com/coursemat.html

Outline

- What is a game?
- Game Types
- Game Development Team
- Game Engines

What is a Game?

- Interactive experience
  - That provides the player with an increasingly challenging sequence of patterns
  - Which he or she learns and eventually masters
- The core idea is that the “fun” is experience during the play

First-person Shooters

- Games like Unreal, Half-life, Call of Duty
- Focus on
  - Efficient rendering of large 3d worlds
  - Responsive camera control
  - High-fidelity animations
  - Detailed weapons
  - Good physics model
- Rendering technology focuses on optimization for the environment

Third-person games

- Games like Ratchet and Clank, Gears of War, Ghost Of Tsushime, Mount & Blade
- Focus on
  - Action environments
  - Puzzle like elements
  - Moving environmental objects
  - Third person follow camera
  - Complex camera collision system

Fighting games

- Games like Tekken, Street fighter, Fight Night, and Soul Calibur
- Technology focus on
  - Fighting animations
  - Hit detection
  - User input system
  - Crowds
  - Awesome character animations and shaders
  - Physics based cloth and hair

Racing games

- Games like Need for Speed, Grand Turismo, Mariokart, Hydro Thunder
- Technology tricks include
  - Using simple cards for background objects
  - Track is broken down into sectors
  - Third-person and first person cameras
  - Camera collision

Real-time strategy

- Games like Warcraft, Starcraft, Age of Empires
- Technology involves
  - Low resolution characters
  - Height map based terrain
  - Complex goal trees
  - User interaction can take many forms,
    - but reactivity is really important

Massive Multiplayer Online Games (MMOG)

- Games like World of Warcraft, Star Wars Galaxies, EverQuest
- Extra technology over 3rd person include
  - Server side artifacts for
    - Sign in/out
    - State management
    - Billing
  - Client side rendering and state management
  - Network layer for state consistency and cheat detection

Simulation Games

- Games like Microsoft Flight Simulator
- Air to simulate physical activities such as driving an aircraft, a board, a train, etc.

Serious Games

- Aimed toward problem-solving rather than entertainment
- Used in various areas
  - Education, Healthcare,
  - Marketing, Defence

VR & AR & MR Games

- Technologically similar in many respects to first-person shooter engines
- Many FPS-capable engines such as Unity and Unreal Engine support VR & AR & MR .

VR & AR & MR Games

- Differ from FPS games in a number of significant ways:
  - Stereoscopic rendering
    - A VR game needs to render the scene twice, once for each eye
  - Very high frame rate.
    - Studies have shown that VR running at below 90 frames per second is likely to induce disorientation, nausea, and other negative user effect
  - Navigation (position & orientation tracking)

Games with Player-Authored Content

- Allowing the player to build content as part of the game
  - Different from Mods
- Good example include
  - Little Big Planet series
  - Minecraft
- Fun is in sharing with others
- Simplicity is key

- Lots of members, many jobs
  - Managers
  - Engineers
  - Artists
  - Game & Level Designers
  - Producers
  - Publisher
  - Other Staff

Game Development Team

Managers

- Performs high level management of game development team
- Not need to be a technical people

Engineers

- Build software that makes the game and tools work
- Lead by a senior engineer (technical leader)
- Game programmers
- Tool programmers

Artists

- Content is the king
- Lead by the art director
- Game gome into life with many artists
  - Concept Artists
  - 3D modelers
  - Texture artists
  - Lighting artists
  - Animators
  - Motion Capture
  - Sound Design
  - Voice Actors

Game Designers

- Responsible for game play
  - Story line
  - Details of scenario
    - Friends
    - Threats
    - Puzzles
    - Levels
    - Weapons
- Employ writers and sometimes old engineers

Producers & Publishers

- Producers:
  - Work with designers, lead artist and lead programmer to coordinate and lead development of game in mostly artistic manner
- Publisher
  - A company that publishes and sells the game

Game Engines

In the old days, programmers would design games from scratch.

Given the complexities of modern games, game programmers rely on software systems called game engines to provide underlying support (rendering, animation, physics).

This frees the game programmer from many of the more advance technical tasks in order to focus on the essential elements of the game play.

Some common examples of game engines include Unity 3D, Unreal Engine and Cry Engine

What is a game engine

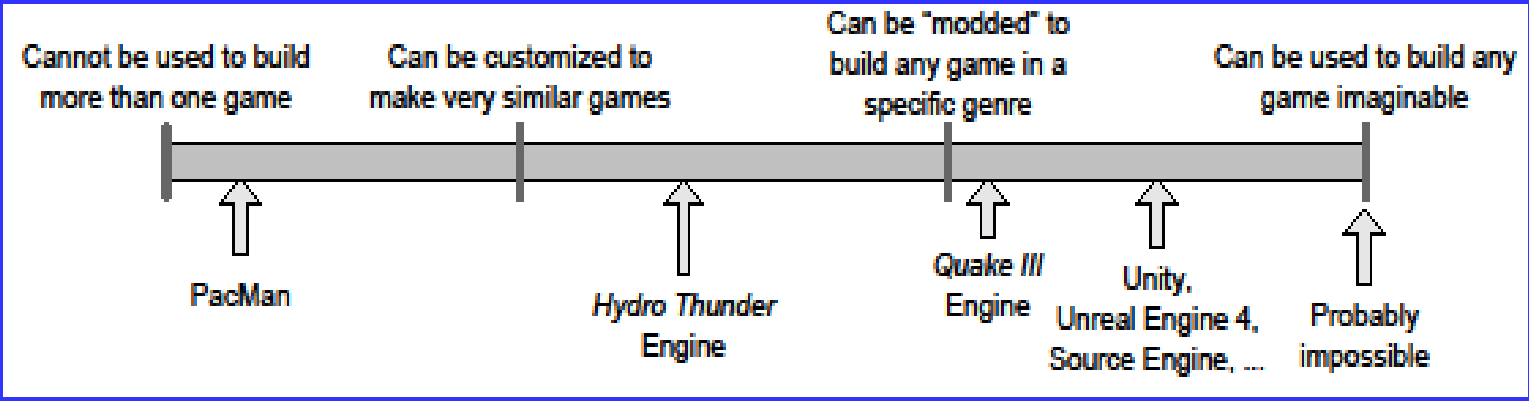
A game engine is a software system designed for the creation and development of video games. (Wikipedia).

The core functionality typically provided by a game engine may include

- a rendering engine ("renderer") for 2D or 3D graphics,
- a physics engine or collision detection (and collision response),
- sound,
- scripting,
- animation,
- artificial intelligence,
- networking,
- streaming,
- memory management,
- threading,
- localization support,
- scene graph,
- video support for cinematics.
  - Greatest cinematics ever:
    - https://youtu.be/j7WbkzvTuU8

What is a game engine

- The term game engine arose in the 1990s
- Doom by id was at the center
  - The core components were separated from the game content
- Quake III and Unreal were designed with the separation in mind
  - Sold licenses to their engine and tools
  - So of you may have done modes using these tools
- Game engine are data-driven architectures that are reusable and therefore do not contain game content – mostly true



Some current engines

Quake family

- Used to create many games
- Has links that extends to modern games like *Medal of Honor*
- *Quake* and *Quake II* engines source code are freely available

Unreal Engine

- Very rich tool set
- Large developers network
- Good licensing model – good for small developers
- Uses C++ or blueprint visual scripting language

Some current engines

Unity

- Very feature rich
- Uses Javascript or C# for scripting
- Large community support
- Great for cross-platform development

Source Engine

- Games like *Half-life 2* and its sequels, *Team Fortress 2,* and *Portal*
- Very powerful with good graphics capabilities and a good toolset

DICE's Frostbite

- Used to create games like *Battlefield 4*
- FrostEd – asset creation tool

Some current engines

Cry Engine

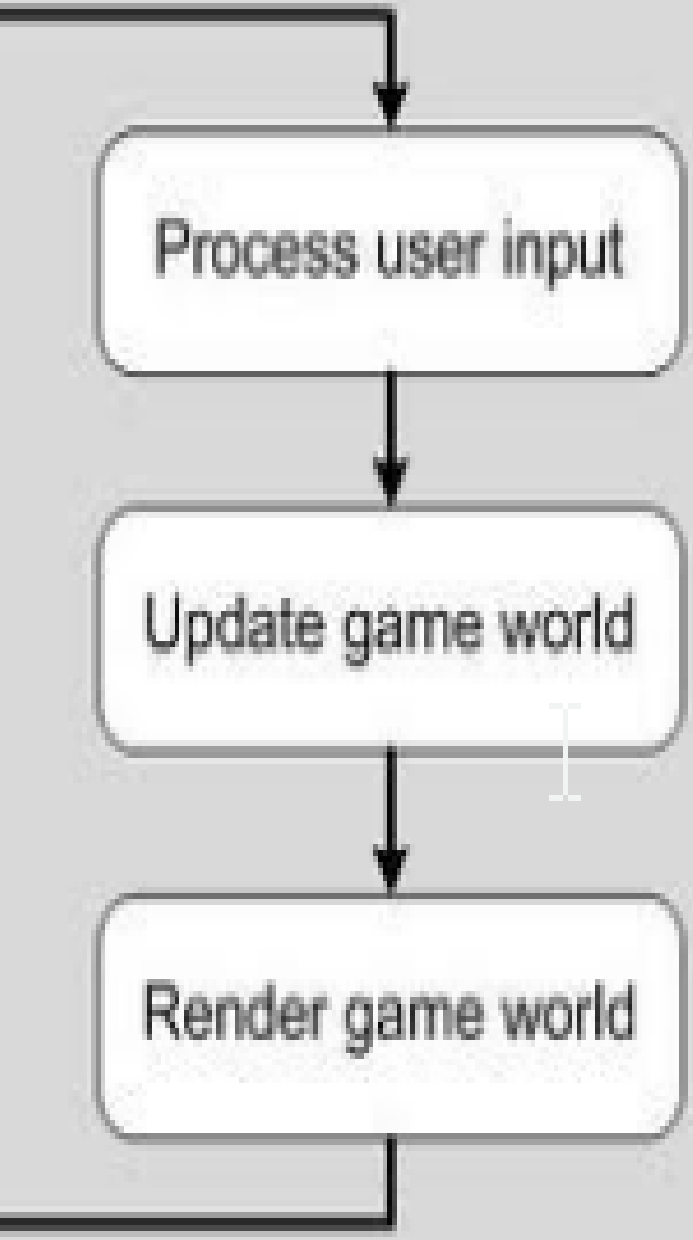
- Originally developed as a demo for Nvidia
- Used to develop numerous games – starting with *Far Cry*

Sony PhyreEngine

- Uses to create games for the Sony platforms
- Numerous titles have been written with this engine

Microsoft XNA and MonoGame

- Based on C# - easy to use
- Used for Xbox and PC games
- Not longer supported – replaced by MonoGame



Game Engines

- Since the flow of the program (game) depends on the actions of the player, most of the game engines are event driven or mix:
  - Updates.
  - Mouse movements.
  - Key presses.
  - Avatar collision.
  - Avatar near a bot.
  - Network activity.
  - The contents of a environment definition file.
- Programming a game engine is therefore a matter of programming what happens when an event occurs.

Game Engines

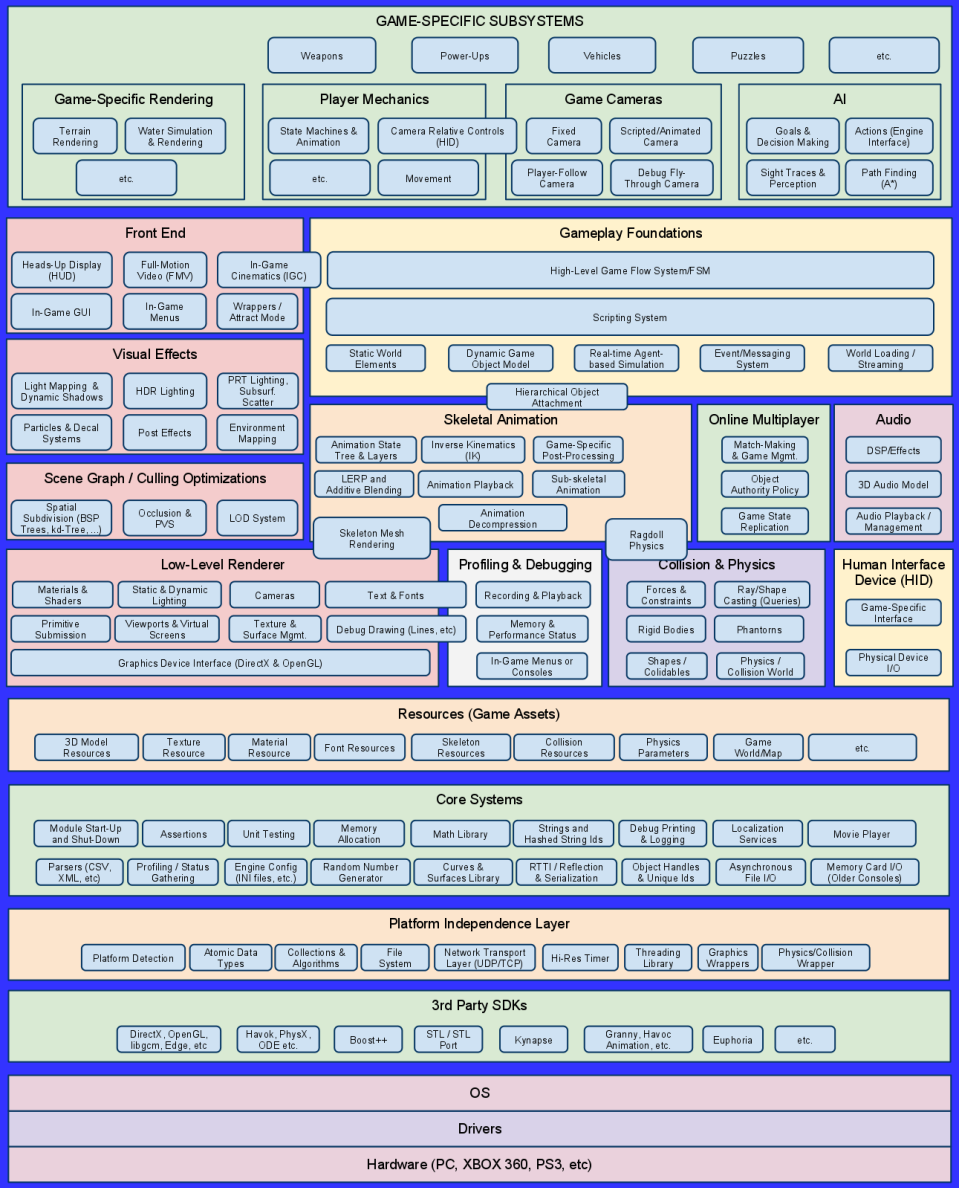
	Unity	Unreal	Torque	HTML5	Flash	Source Engine	Ogre	Second Life	GameMaker	XNA
Level editor	■	■	■		■	■		■	■	
Scripting	■	■	■	■	■			■	■	
C++		■	■			■	■			
Networking	■	■	■	■	■	■		■	■	■
3D Graphics	■	■	■			■	■	■		■
Shader effects	■	■	■			■	■		■	■
Dynamic shadows	■	■	■			■	■			■
Physics	■	■	■			■		■	■	■
Artificial Intelligence	■	■	■			■			■	
Free non-commercial	■	■	■	■		■	■	■		■
Free for commercial				■			■			■
Mobile Devices	■	■		■	■		■		■	■
Web player	■			■	■				■	■

- Cowan, Brent and Bill Kapralos.
- “A Survey of Frameworks and Game Engines for Serious Game Development.”
- *2014 IEEE 14th International Conference on Advanced Learning Technologies* (2014): 662-664.

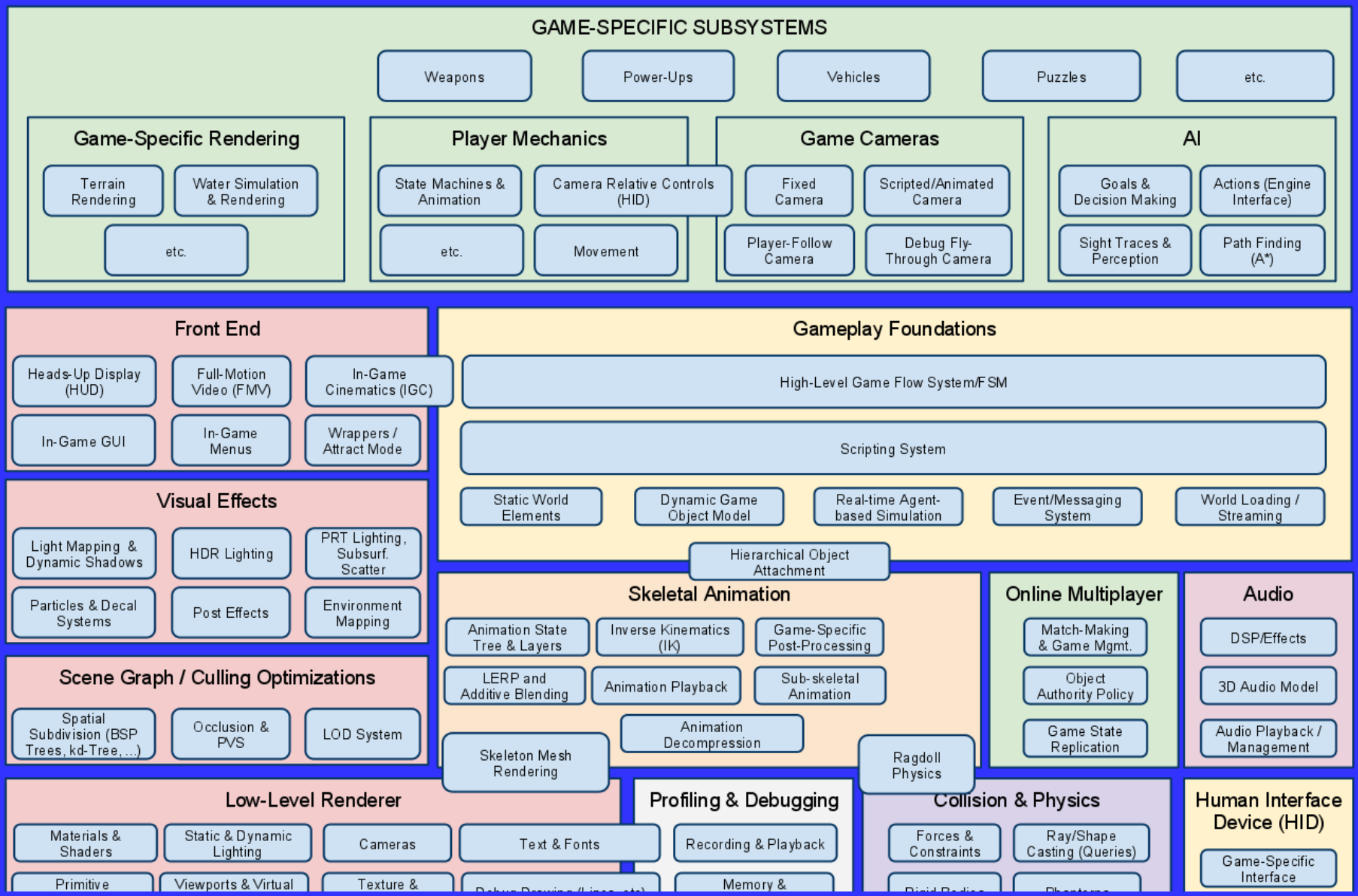
Game Engine Architecture

- Consists of the tools/editors and runtime game components
- Large Scale
  - Spans hardware to high-level application
- Designed in layers
  - Avoids circular dependencies to maximize reuse and testability

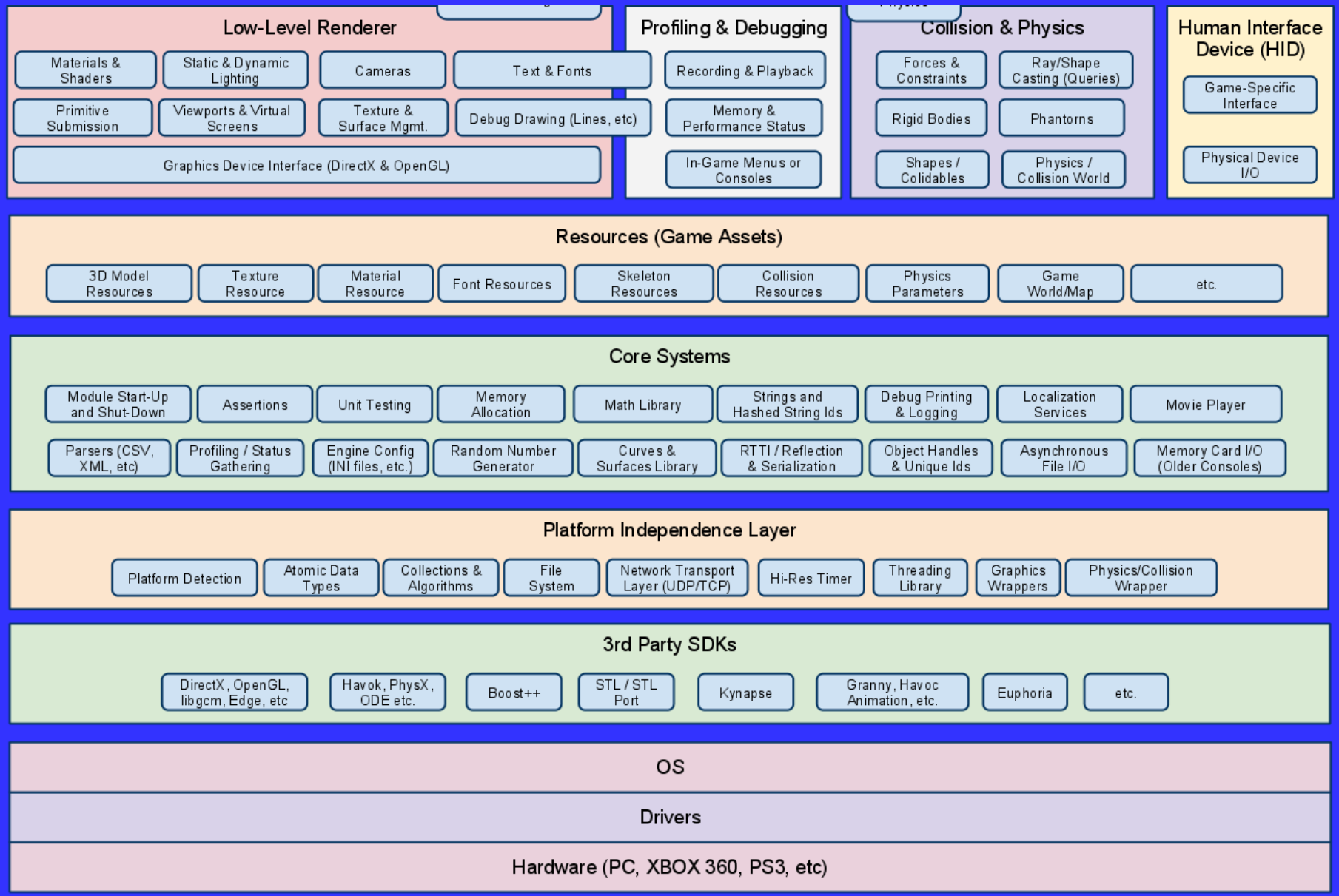
A Sample Game Engine Architecture



A Sample Game Engine Architecture



A Sample Game Engine Architecture

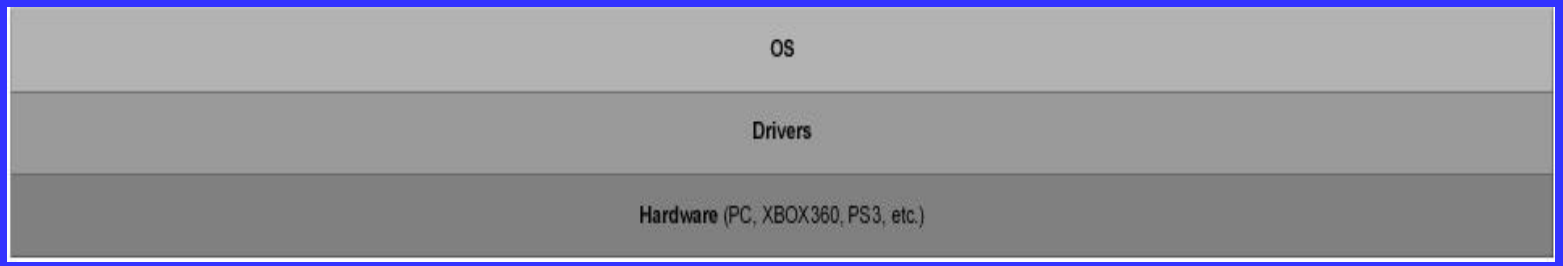


Game Engine Architecture

- Low level components
- 3rd Party SDKs
- Platform independence layer
- Core systems
- Resources manager
- Rendering engine
- Profiling/Debugging
- Collisions and Physics
- Animation
- Human Interface Devices
- Audio
- Gameplay foundation system

Low level components

- Hardware
  - This is the system that the game is to run on
- Device Drivers
  - Shield the OS and upper layers from low level device communications details
- Operating System
  - Handles the execution and interruption of multiple programs on a single machine
  - Very thin on a console



3rd Party SDKs

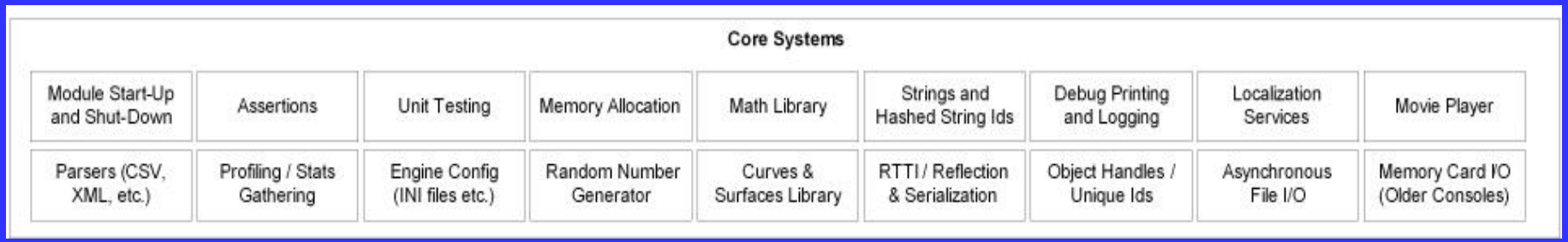
- These are libraries and sofware development toolkits (SDKs), usually provided from a third party.
- Data Structure and Algorithms
  - STL (Standard Template Library) – C++ data structures, strings, stream-based I/O
  - Boost – powerful data structures and algorithms
- Graphics
  - OpenGL, Vulkan and DirectX
- Collisions and Physics
  - PhysX, Havok, ODE, Bullet
- Character Animation
- Artificial Intelligence – Kynapse (Autodesk)

Platform independence layer

- Allows the engine to be developed without the concern of the underlying platform
- Provides wrappers to common target specific operations
- Include things like primitive types, network, file systems, etc.
- Idea is to have generic system access libraries not specific to an OS

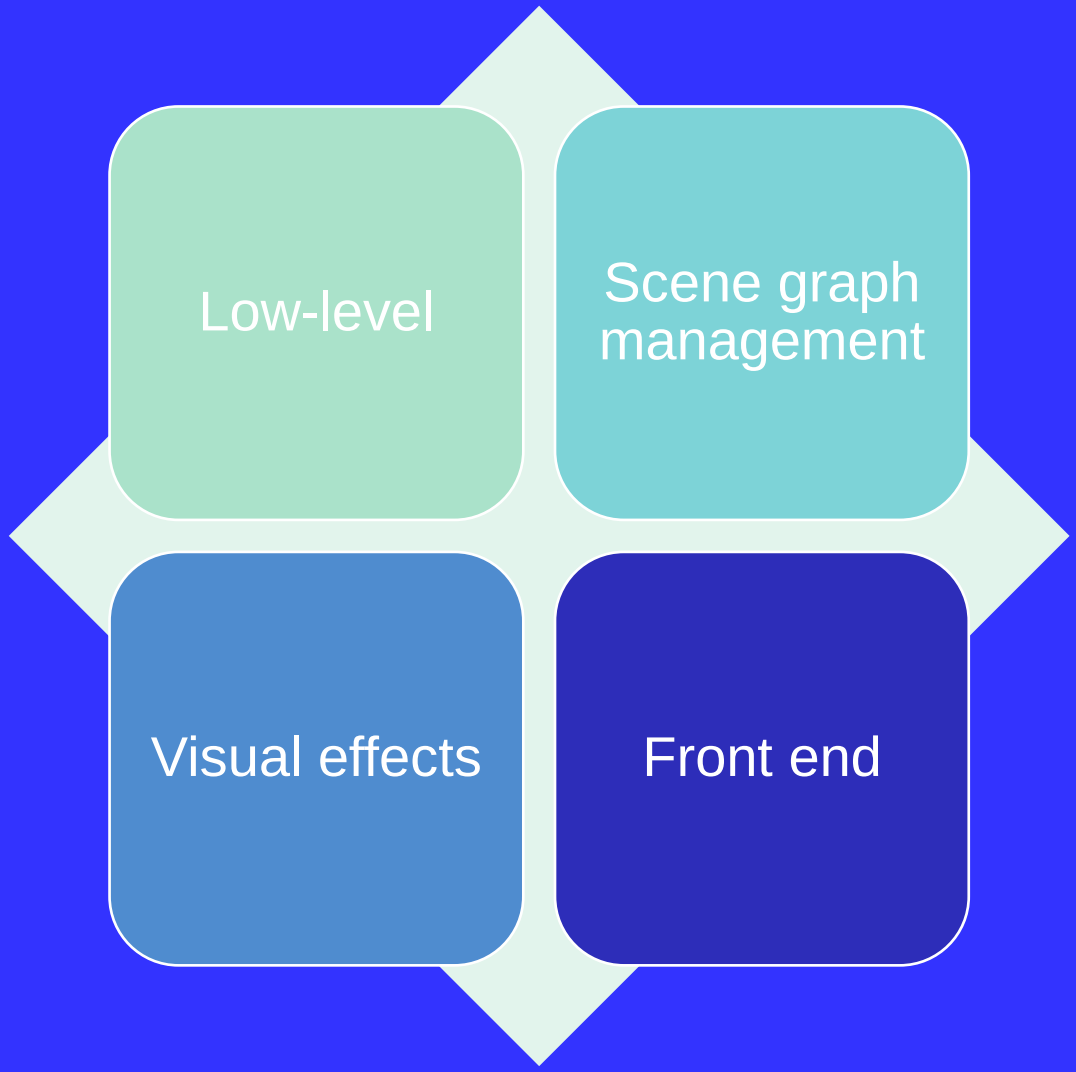
Core systems

- Assertions – error checking code
- Memory Management
- Math library – vector and matrix math, numeric integrators
- Custom data structures



Resource manager

- Provides a unified interface for accessing assets
- The level of complexity is dictated by need
  - Often the game programmers must do resource loading directly
  - Engines like Unreal do unpackaging and complex manipulation of assets in the engine



Rendering Engine

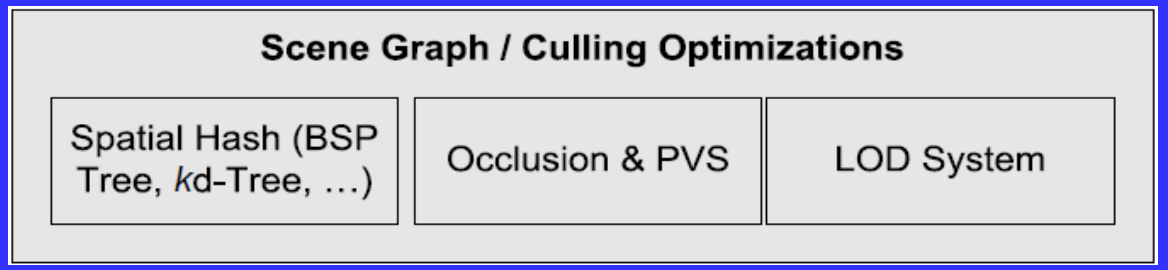
This is one of the largest and most complex components of any real-time 3D Game.

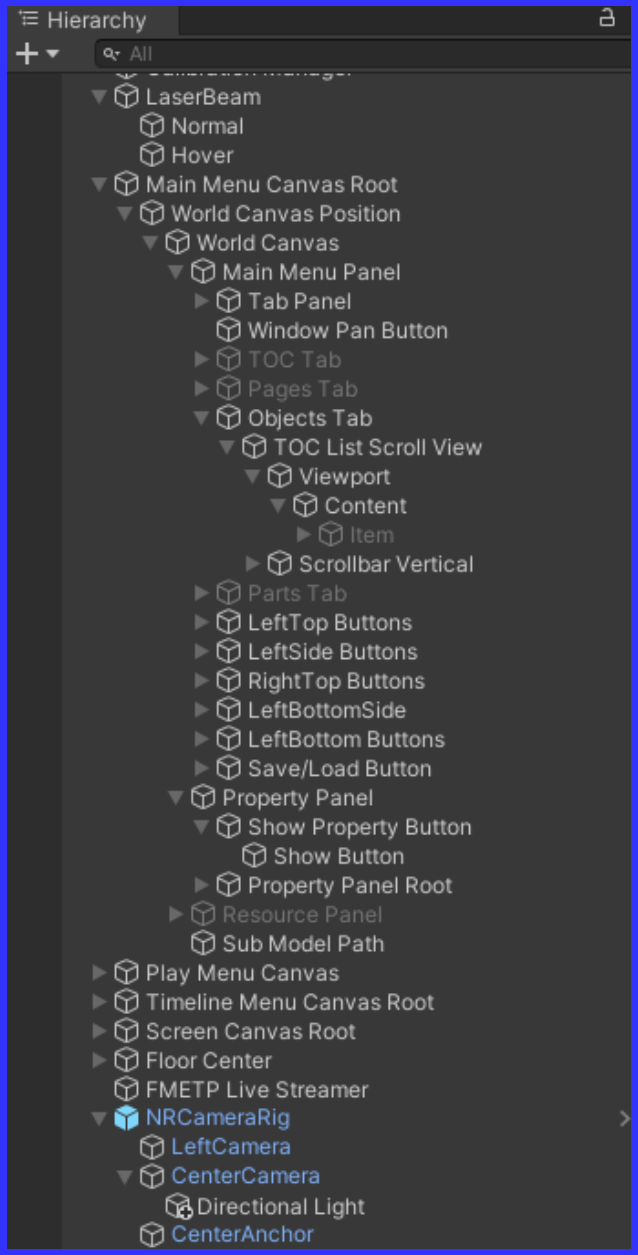
This involves all aspects of drawing, and may involve close interaction with the graphics hardware, or graphics processing unit (GPU),

for the sake of enhanced efficiency

Scene graph

- A **scene graph** is a general data structure commonly used by modern computer games, which arranges the logical and often spatial representation of a graphical scene mostly.
- It is a collection of nodes in a graph or tree structure.
- Simplifies scene design and optimizes performance
- Limits the number of primitives submitted for rendering
- Uses frustum culling to remove things in invisible area
- Perform spatial subdivision using advance algorithms
  - BSP, quadtree, octree, kd-tree





Scene Graph

- A sample Unity scene graph

Visual effects

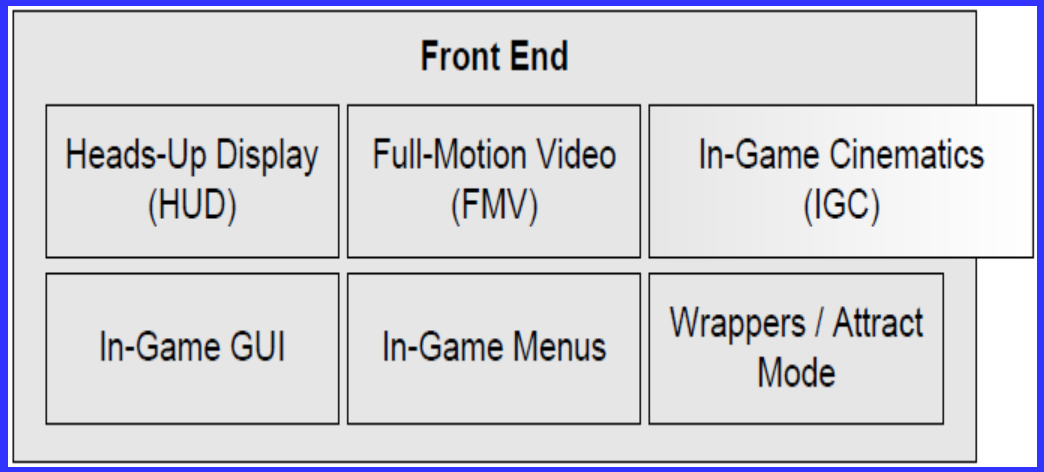
- Particle systems
- Decal systems
- Light mapping
- Dynamic shadows
- Full screen post process effects

Decal

Visual Effects		
Light Mapping & Dynamic Shadows	HDR Lighting	PRT Lighting, Subsurf. Scatter
Particle & Decal Systems	Post Effects	Environment Mapping

Front end

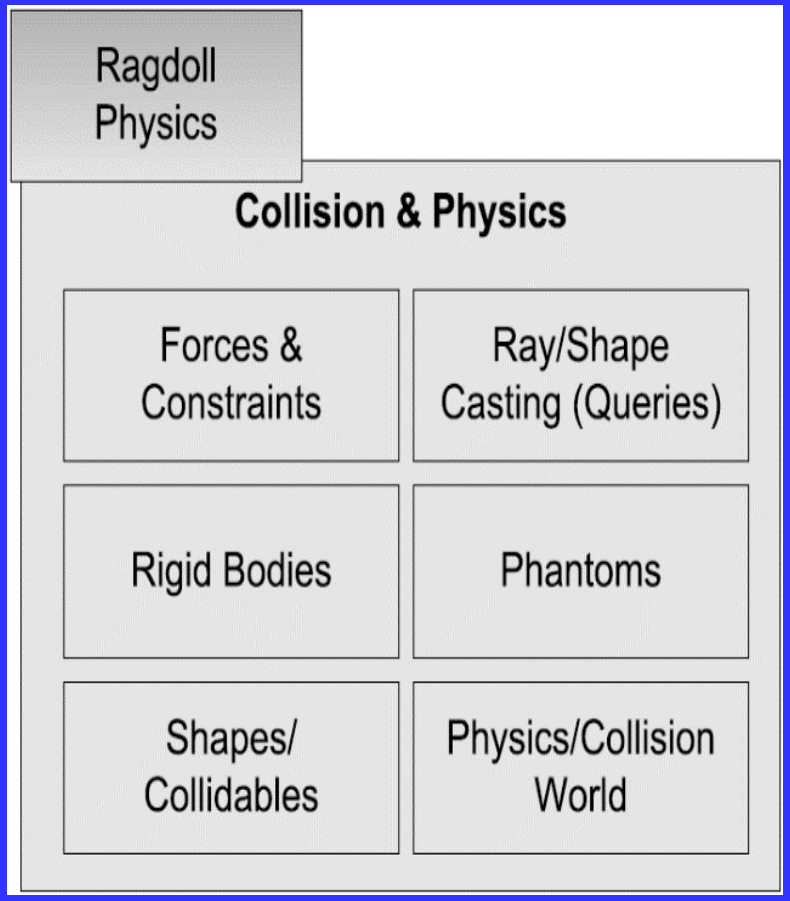
- User Interface (UI) Components
- Buttons, Labels, Menus, etc.
- GUI for character manipulation
- Full-motion video for cut scenes



Profiling/Debugging

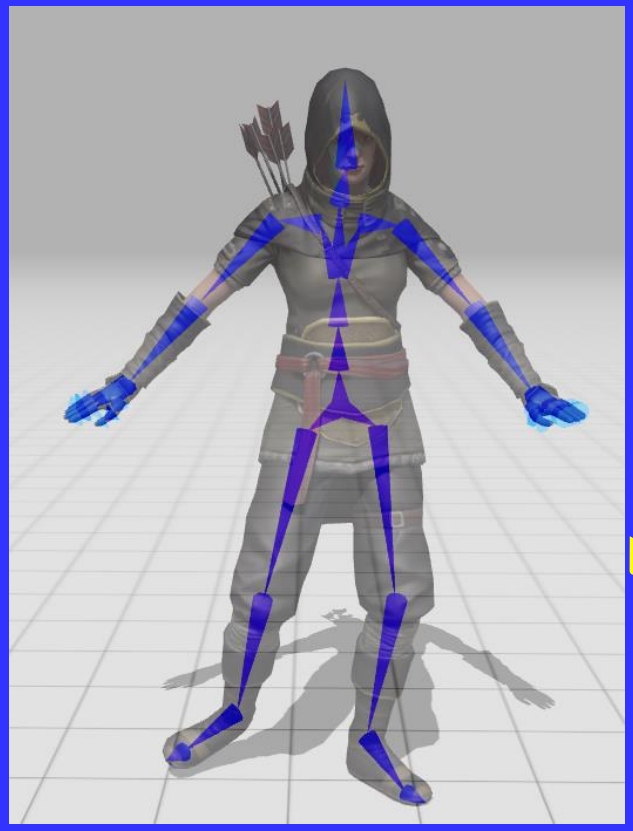
- Code timing
- Display stats on the screen
- Dumping performance stats
- Determining memory usage
- Determining memory leaks
- Dumping memory usage
- Record and playback game events
- Print statement output control

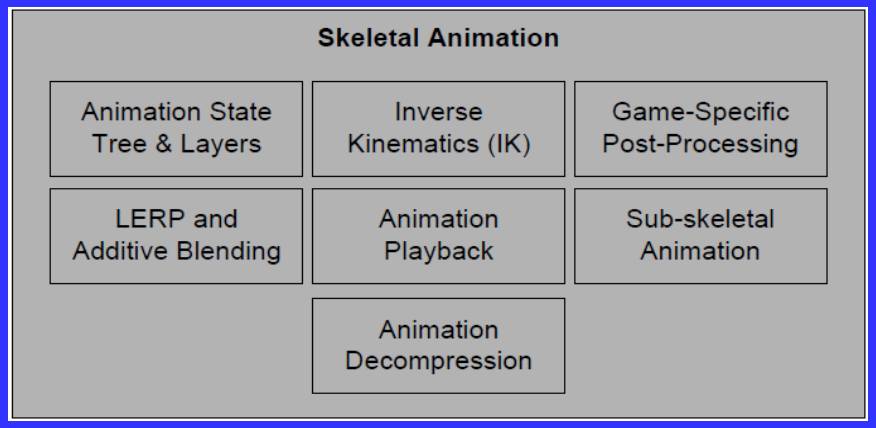
Collisions and Physics



- Usually rigid body dynamics
- Simulate the movement of objects over time,
- Detect when objects collide with one another, and
- Determine appropriate response in the event of a collision
- Physical simulation can be very difficult
- Many companies use available libraries
  - PhysX
  - Havok
  - ODE
  - Bullet

Animation



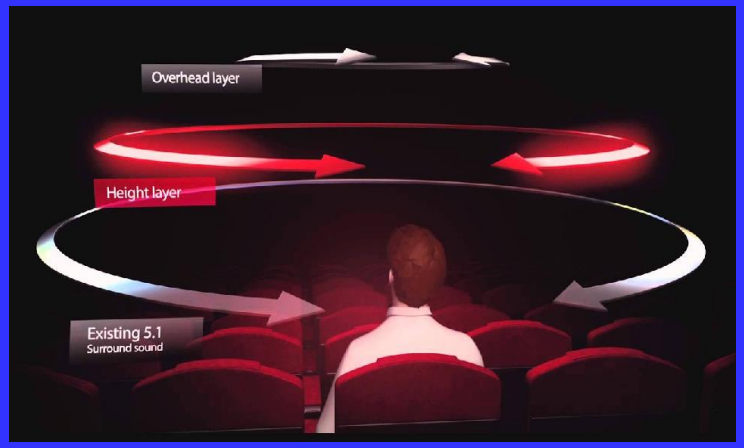


- While game direct a character to move from a location to another,
- Job of the animation system is to make this motion look natural,
- For instance, by moving the arms and legs in a manner that is similar to a normal walking behavior.
- Five types of animation are used
  - Sprite/texture animation
  - Rigid body animation
  - Skeletal animation
  - Vertex animation
  - Morphing
- Skeletal animations still the most popular

Human Interface Devices (HID)

- These components process inputs from the users
- Keyboard and mouse
- Touch pads
- Wheels, Pedals, Joysticks
- Specialized controllers
- Massages raw data into useful information

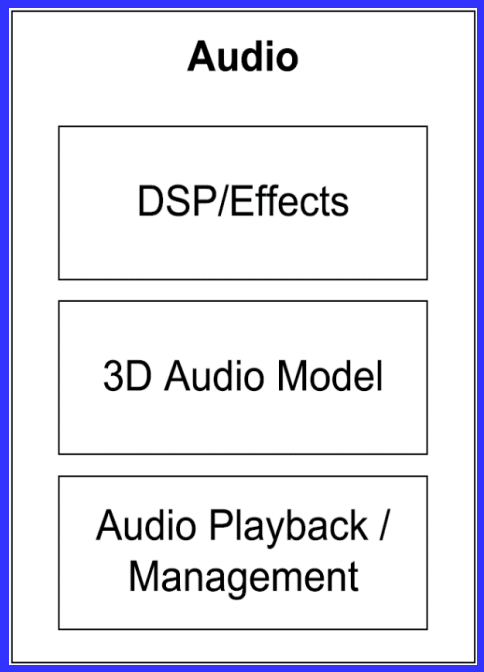
Audio



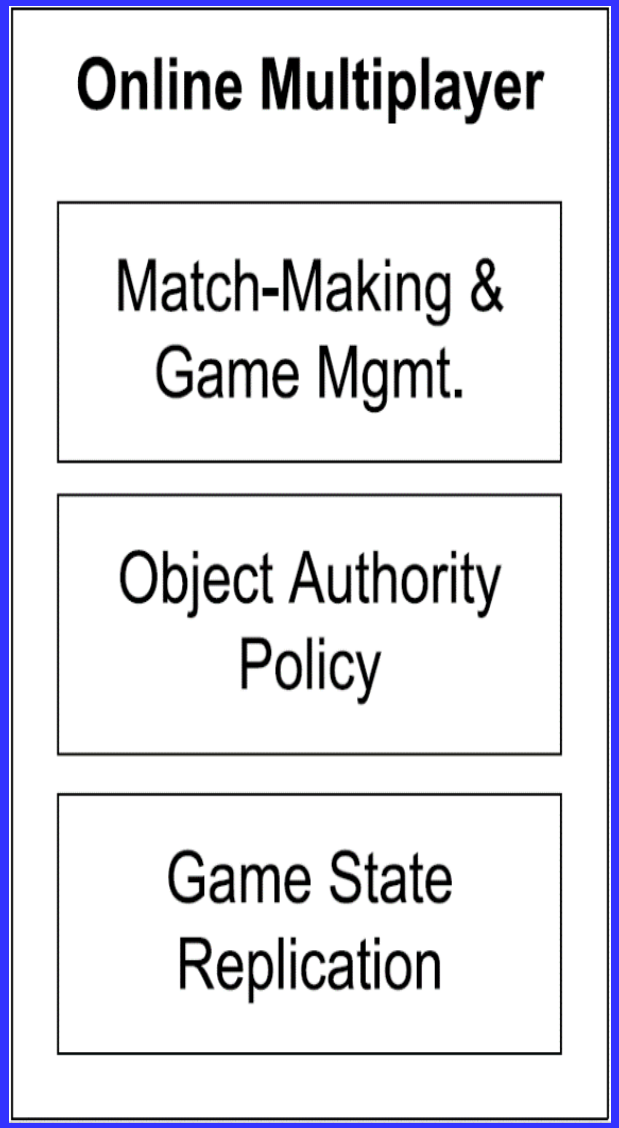
- Audio components handle simple things like
- Play background music,
- Explosions, fire, car engines, etc.

- But they may also generate special audio effects, such as stereo 3D effects to give a sense of location

- Often skipped until the very end
- Since adding sound is not changing the game architecture so much

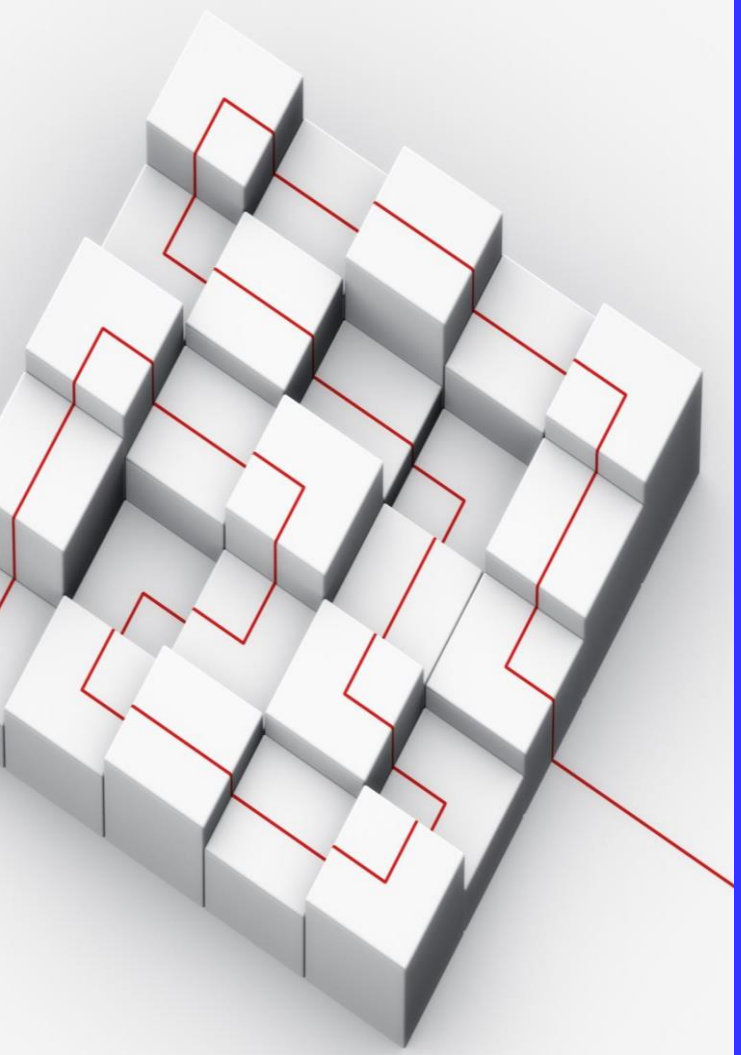


Multiplayer / Networking



- Multiplayer and online games require a number of supporting components.
- For example, multiplayer games may have a split-screen capability.
- Online games require support for wide area network communication

- Four main flavors
  - Single screen – multiple players on the same screen
  - Split-screen multiplayer – multiple perspectives on the same screen
  - Networked multiplayer – multiple computers networked together
  - Massive multiplayer online games – run in a central server
- Difficult to convert single to multiplayer, easy to do the opposite



Artificial Intelligence

- Used to control the behavior of non-player characters.
- Typically modeled as AI agents.
- An agent is an object that can
  - Sense its environment,
  - Maintain a memory/state, and
  - Respond in complex ways depending on the current input and state.

- Some of AI building blocks
  - Navigation mesh generation
  - Path planning
  - Object avoidance
  - Sensing
  - Making Decisions, and
  - Responding

Digital content creation

Game engines deal with data in many forms

The data has to be created somehow

- 3D Meshes
- Textures
- Sound
- Animations

Often created using advanced external tools

- Maya / 3ds Max / Blender
- Photoshop / Adobe Premier / After Effects
- SoundForge

These modeling tools need to be easy to use and reliable

Game world editor

Usually integrated into the engine

Essential to allow game designers to work with the engine

Your Own World Editors

- But sometimes you may need a seperate world editor apart from the game engine world editor for you special requirements
- SimRealize Simulation World Editor (SimBT Ltd.Şti.)

https://www.youtube.com/watch?v=ca2yU-VikB8

Resource database

- Need a way to store and manage the vast amounts of data
- Some companies use relational databases
  - MySQL or Oracle
- Some companies use version control software
  - Subversion, GIT, etc.
- Still others use custom software

Advantages and Disadvantages

- Advantages in using a game engine
  - Less development time required
  - Less testing and debugging
  - Many features directly available
  - Better focus on the game design
  - Ready other libraries/toolkits linked with the game engine

- Disadvantages in using a game engine
  - No or less control over the implementation of features
  - Adding features not yet in the game engine might be problem
  - Dependent on licensing scheme for release
  - New other libraries/toolkits linked with the game engine