There are different kinds of 3D game engines out there. Looking at various aspects including the pros and cons of these game engines, we finally decided to choose Unity3D game engine for developing this Adwa 3D interactive game. To mention some of the game engines we were looking at pros and cons of different kinds of game engines



**Pros**

* Easy to use and compatible with every game platform.
* Great community support.
* Low learning curve.

Cons

* Time-consuming for making games with complex and diverse effects.

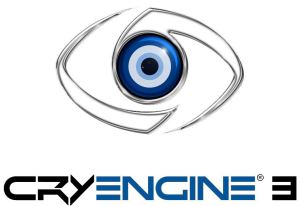


Pros

* With so many developers using it, Unreal offers the largest community support. Several lifetime hours of video tutorials and assets are available
* There is widest range of easy to maneuver tools up under its sleeve. There are few tools that can be maneuvered even by a school kid.
* Compatible with diverse operating platforms including iOS, Android, Linux, Mac, Windows, and most game consoles.

Cons

* Some developers complain a lot about the unfriendly tools that involve a bit of a higher learning curve.



**Pros**

* The game engine also offers the easiest Al coding of any tech on the market.
* Cry Engine 3 makes the game ambience pretty with its artist-level programming capability in its Flow graph tool.
* For a beginning developer, UI scale form comes handy.

Cons

* Being relatively new to the industry, the engine is yet to find a robust community
* Learning curve is pretty challenging for a starter



Pros

* It offers several open-world maps and lively instances. It also offers a seamless transition between them.  
  Offers fairly robust A.I.s.
* Easy and handy mapping tools and integrated tool set.
* The scripting is powerful enough to help you develop complex missions, crafting, and gathering resources.

Cons

* Scripting engine is powerful but no intuitive.
* HeroEngine together with HeroCloud is expensive for a new developer or startups.
* Higher learning curve for a new developer.

Looking at these all game engine s we finally choose Unity3D since the unity3D game engine has low learning curve and there is wider community base on unity compared to other game engines.

**Top-Level Components of Adwa 3D Game**

The Adwa game is organized in levels that correspond to the different stages and roles. The player progresses in the game by accumulating experience and points in return for accomplishing simple tasks. Each role has its own objectives and tasks.

Adwa 3D game architecture has the high-level structure as shown in Figure 1. The functional components of Adwa 3D interactive gaming are, **The Game Engine, The Simulation, Game Object System**  and **The Data Manager**.

**Simulation Controller**

**Object System**

**Data Manager**

**The Game Engine**

Unity3D Engine offer reusable components that can be manipulated to bring a game to life. Loading, displaying, and animating models, collision detection between objects, physics, input, graphical user interfaces, and even portions of a game's artificial intelligence can all be components that make up the engine. In contrast, the content of the game, specific models and textures, the meaning behind object collisions and input, and the way objects interact with the world, are the components that make the actual game.

The game engine is responsible for interacting with the player, through the computer hardware and operating system. In most cases, the game engine can be designed and developed as generic software for a particular game genre.

The primary modules in the game engine are **Graphics**, **Audio**, **OS interface** and the like

Graphics



**Game Object**

Audio

Interfaces

Game Objects

Game Engine

Computer

Player

One characteristic of the game engine modules is that their functional decomposition is natural and there is little or none coupling between them. In addition, these modules have very little internal state and the data flow through them is uni-directional.

While the internal structure and algorithmic design of these modules is non-trivial, they are better understood and documented than many other components in game software.

**Game Objects**

Game objects represent the actors of the game world. A game object can be either static or dynamic. Static game objects do not react to external stimuli and therefore, do not interact with the world or other game objects. They are often scenery elements, like a tree or a rock. Dynamic game objects, on the other hand, can interact with other game objects.

The object system is responsible for maintaining the state information describing all objects in the game world. This system resides in memory for real-time access to all game state information.

Camera Component

Drawable

Audio Component

**Game Object**

Video Component

**The Simulation**

The simulation is the heart of the game itself. This component realizes the virtual world of the game and maintains the rules of the game. The simulation control module receives player input information from the game engine and passes it to **The Player AI**, **The Enemies AI**, **The Bot AI module**, **The Game Logic** or to **The Physics.**

Player AI

**Simulation Controller**

**Game Object**

Physics

Game Logic

Bot AI

Enemies AI

The physics module is concerned with the interaction of objects in the virtual world according to the physical properties of that world. The AI module is concerned with the internal thought and decision process within (virtually) animate objects. Finally, the game logic module handles those issues that are not specifically part of the virtual world in which the game is played, but are rather computations and constraints inferred from the rules of the game. These modules do not interact directly; rather they interact through the information that describes the state of the objects in the game world that is stored in the object system. In our discussions, we found it useful to think of the interaction between AI and physics as follows. AI generates impulses based on the internal state of animate objects. For example, the AI for an object might generate an impulse to move towards water or to swing a fist. Physics turns impulses into actions and ensures that the results of those actions do not violate the physical laws of the virtual world.

**The Data Manager**

The data manager is responsible for retrieving game data from the file system or some other persistent storage and for managing storage and retrieval of game state for save/load game functionality.

The object system is responsible for maintaining the state information describing all objects in the game world. This system resides in memory for real-time access to all game state information.

**Why Unity3D?**

Unity3D game engine has many diversified components which make it easy to make any interactive game in less effort and high quality. Among those different kinds of components that the unity system provides, here are some of the components that we are going to use to develop the Adwa 3D interactive game.

Collider

Cloth

Character Joint

Constant Forces

Character Controller

Sprite

Lighting

Train

Camera

**Graphics Elements**

**Scripting Tools**

**Physics Components**

**Version Controller**

**User Interface Systems**

Canvas Components

Event System

Visual Components

Time & Frame rate mng

Interaction Components

Control Game Objects

**Version Controller**

Using a **version control system** makes it easier for a user or multiple users to manage their code. It is a repository of files with monitored access, which in the case of Unity, will be all the files associated with a Unity project. With version control it is possible to follow every change to the source along with information on who made the change, why they made it and what they changed/added. This makes it easy to revert back to an earlier version of the code or to compare differences in versions. It also becomes easier to locate when a bug first occurred along with what code might have caused it.

**Graphics Elements**

* **Camera**

A camera is an object that defines a view in scene space. The object’s position defines the viewpoint, while the forward (Z) and upward (Y) axes of the object define the view direction and the top of the screen, respectively. The Camera component also defines the size and shape of the region that falls within the view. With these parameters set up, the camera can display what it currently “sees” to the screen. As the camera object moves and rotates, the displayed view will also move and rotate accordingly.

* **Lightning**

This tool is used to calculate the shading of a 3D objects in the game. There are many advanced lighting features available in Unity, which are intended to help in creating a better

* **Train**

Unity’s Terrain system allows adding vast landscapes to game. At runtime, terrain rendering is highly optimized for rendering efficiency, a selection of tools is available to make terrains easy and quick to create. This section explains the various options available for terrains and how to make use of them.

* **Sprite**

Sprite is a two-dimensional image or animation that is integrated into a larger scene.

**Physics Components**

* **Colliders**

Collider components define the shape of an object for the purposes of physical collisions. A collider, which is invisible, need not be the exact same shape as the object’s mesh and in fact, a rough approximation is often more efficient and indistinguishable in game play.

* **Character Joint**

Used to attach one rigid body object to another or to a fixed point in space using a Joint component. Generally, a joint to allow at least some freedom of motion and so Unity provides different Joint components that enforce different restrictions. For example, a Hinge Joint allows rotation around a specific point and axis while a Spring Joint keeps the objects apart but lets the distance between them stretch slightly.

* **Cloth**

The Cloth component provides a physics-based solution for the simulation of fabrics and works in conjunction with the Skinned Mesh Renderer. While it has been specifically designed for character clothing it is still possible to use arbitrary, non-skinned meshes.

* **Character Controller**

The Character Controller is mainly used for third-person or first-person player control that does not make use of Rigid body physics.

* **Constant Forces**

Constant Force is a quick utility for adding constant forces to a Rigid body

**Scripting Tools**

* **Event System**

The Event System is a way of sending events to objects in the application based on input, be it keyboard, mouse, touch, or custom input. The Event System consists of a few components that work together to send events.

* **Time and Frame rate manager**
* **Control Game objects**

This tool helps a lot in controlling the game objects in the scenes using the game object components

**User Interfaces**

* **Canvas Components**

The Canvas component represents the abstract space in which the UI is laid out and rendered. All UI elements must be children of a Game Object that has a Canvas component attached.

* **Visual Components**
  + **Text**

The **Text** component, which is also known as a Label, has a Text area for entering the text that will be displayed. It is possible to set the font, font style, font size and whether or not the text has rich text capability.

* + **Image**

An Image has a Rectangular Transform component and an **Image** component. A sprite can be applied to the Image component under the Target Graphic field, and its color can be set in the Color field. A material can also be applied to the Image component.

* + **Mask**

A Mask is not a visible UI control but rather a way to modify the appearance of a control’s child elements. The mask restricts the child elements to the shape of the parent. So, if the child is larger than the parent then only the part of the child that fits within the parent will be visible.

* + **Effects**

Visual components can also have various simple effects applied, such as a simple drop shadow or outline. See the Effects reference page for more information.

* **Interaction Components**

Interaction Components handles interaction, such as mouse or touch events and interaction using a keyboard or controller.