3D Shooting Game

Bhuvan Panchananam, Balaram M.

UG Student [1], Assistant Professor [2],
[1, 2] BCA, BMS College of Commerce and Management

Abstract
Team Deathmatch (TDM) is a popular multiplayer game mode known for its fast-paced action and intense team-based combat. The Unreal Engine, renowned for its robust capabilities in game development, provides an ideal platform for creating immersive and visually stunning gaming experiences. This abstract explores the integration of the Team Deathmatch game mode into Unreal Engine, enhancing the gameplay, mechanics, and overall engagement for players.

Keywords: Dynamic Level Design, Responsive Gameplay Mechanics, Networking and Multiplayer Support

1. Introduction
Team Deathmatch (TDM) is a popular game mode in which players are divided into teams and compete against each other to eliminate opposing team members. Unreal Engine, developed by Epic Games, is a powerful game development engine that provides a comprehensive set of tools and features for creating immersive and visually stunning games.

TDM gameplay in Unreal Engine revolves around strategic decision-making, teamwork, and fast-paced combat. It offers an intense multiplayer experience where players must coordinate with their teammates, utilize various weapons and abilities, and fight the opposing team to secure victory.

Unreal Engine provides a wide range of functionalities to support the development of TDM games. These include the following:

Game Mode Blueprint
Unreal Engine allows developers to create custom Game Mode Blueprints that define the rules and mechanics specific to the TDM game mode. This blueprint serves as the foundation for controlling various aspects of the game, such as team management, scoring, and win conditions.

Level Design Tools
Unreal Engine's powerful level editor enables developers to create visually stunning and strategically designed TDM maps. It provides a vast library of assets, terrain sculpting tools, lighting options, and particle effects to create immersive and engaging environments for players.

Player Character Blueprint
Unreal Engine allows developers to design and customize player characters using Blueprint classes. These blueprints define the appearance, animations, movement mechanics, and abilities of the

characters. Players can choose their preferred characters and utilize their unique skills to gain an advantage in the TDM matches.

Networking and Multiplayer Support
Unreal Engine provides robust networking capabilities, allowing developers to implement seamless multiplayer functionality in TDM games. It supports both local multiplayer and online multiplayer modes, facilitating player interaction and ensuring a smooth and responsive gameplay experience.

2. Software and Hardware Requirements

Hardware Requirements

Computer: A capable computer system with sufficient processing power, memory, and storage to handle Unreal Engine and game development tasks.

Graphics Card: A powerful graphics card capable of handling real-time rendering and supporting the desired visual quality of the game.

Software Requirements

Unreal Engine: Choose the appropriate version of Unreal Engine based on your project requirements and compatibility. Download and install the Unreal Engine editor, which provides a comprehensive set of tools for game development.

Integrated Development Environment (IDE): Unreal Engine uses Visual Studio as its default IDE for coding in C++. Install Visual Studio or another compatible IDE for writing and debugging code.

Source Control: Utilize a source control system such as Git or Perforce to manage versioning and collaboration among the development team.

Modeling and Animation Software: Use industry-standard software like Autodesk Maya, 3ds Max, or Blender for 3D modeling, rigging, and animation.

Texture and Material Creation: Utilize tools like Substance Painter, Photoshop, or GIMP for creating textures and materials for the game's assets.

Audio Editing Software: Use software like Audacity, Adobe Audition, or Pro Tools for editing and processing audio assets for the game.

Programming Languages and Frameworks: C++: Unreal Engine provides a powerful and flexible framework for game development using C++. Knowledge of C++ is essential for implementing complex game mechanics, networking functionalities, and performance optimizations.

Blueprint Visual Scripting: Unreal Engine's Blueprint system allows for visual scripting, enabling designers and developers to create game logic and interactions without writing code.

Additional Tools and Plugins

Marketplace Assets: Explore the Unreal Engine Marketplace for pre-made assets, plugins, and tools that can accelerate development, such as character controllers, UI frameworks, or visual effects.

Performance Profiling Tools: Utilize profiling tools provided by Unreal Engine, such as the built-in Performance Analyzer, to optimize the game's performance and identify bottlenecks.

Bug and Issue Tracking: Employ bug and issue tracking software, such as JIRA or Trello, to manage and track reported issues, tasks, and development progress.

Documentation and Resources

Unreal Engine Documentation: Access the official Unreal Engine documentation, tutorials, and guides to learn and understand the engine's features and functionalities.

Online Communities and Forums: Join Unreal Engine online communities, forums, and Discord channels to engage with other developers, seek assistance, and share knowledge.

3. Goals and Objectives

Player Elimination

The primary objective of TDM is to eliminate members of the opposing team.

Objective Implementation: Design a system that tracks player health and death, and assign points to the team that successfully eliminates the opponent's players.

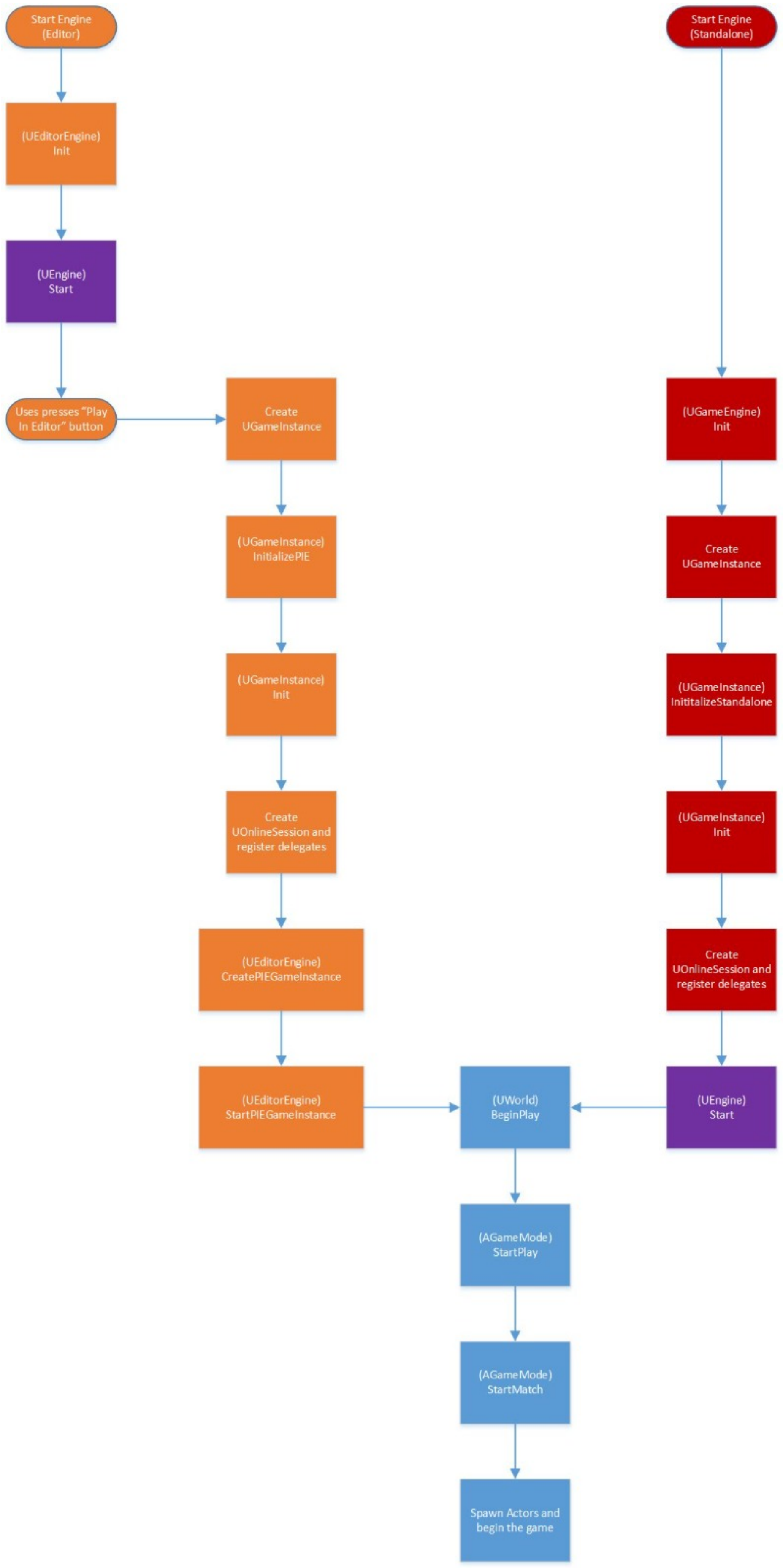
Objective: Team-Based Gameplay

Goal: Encourage teamwork and collaboration among players within each team.

Objective Implementation: Implement a communication system, such as voice chat or in-game text chat, to enable players to strategize and coordinate their actions effectively.

Objective: Balanced Team Sizes

4. DFD


Start Engine (Editor)
(UEditorEngine) Init
(UEngine) Start
Uses presses "Play In Editor" button
Create UGameInstance
(UGameInstance) InitializePIE
(UGameInstance) Init
Create UOnlineSession and register delegates
(UEditorEngine) CreatePIEGameInstance
(UEditorEngine) StartPIEGameInstance
Start Engine (Standalone)
(UGameEngine) Init
Create UGameInstance
(UGameInstance) InititalizeStandalone
(UGameInstance) Init
Create UOnlineSession and register delegates
(UEngine) Start
(UWorld) BeginPlay
(AGameMode) StartPlay
(AGameMode) StartMatch
Spawn Actors and begin the game

5. Conclusion

In conclusion, creating a Team Deathmatch (TDM) game mode in Unreal Engine requires careful consideration of various goals and objectives to ensure a balanced, engaging, and fun multiplayer experience. By implementing the right mechanics and features, you can foster teamwork, strategic gameplay, and fair competition among players. Here's a summary of the key points:

Player Elimination: The primary objective is to eliminate members of the opposing team, and points are awarded accordingly.

Team-Based Gameplay: Encourage collaboration and communication within teams to enhance teamwork.

Balanced Team Sizes: Implement systems to ensure fair and equal distribution of players across teams.

Respawn System: Allow players to respawn after elimination to keep them engaged throughout the match.

Score Limit or Time Limit: Define conditions for winning the match based on reaching a score limit or within a specific time frame.

Map Design and Layout: Design well-balanced maps with multiple routes, cover spots, and choke points to cater to different playstyles.

Weapon and Power-up Placement: Strategically place weapons and power-ups to reward exploration and tactical decision-making.

Anti-Spawn Camping Measures: Implement mechanisms to prevent unfair advantages from spawn camping.

Spectator Mode: Include a spectator mode for players to watch ongoing matches and learn from others.

Post-Match Statistics: Provide players with feedback through post-match statistics to track performance and identify areas for improvement.

Remember that game development is an iterative process, and player feedback is invaluable. Continuously play-test, gather feedback, and make adjustments to refine the TDM game mode until it delivers a rewarding and enjoyable multiplayer experience for players.

With Unreal Engine's powerful tools and resources, you have the opportunity to create an exciting TDM game mode that captures the essence of team-based competition and keeps players coming back for more action-packed battles.