UNREAL ENGINE

Online MultiPlay Game Design

Epic Games Japan / Support Engineer
Ken Kuwano

Gale

- C++とBlueprintの使い分けた設計
- Engineが提供する機能
- 各種機能を実現するフロー
- ネットワーク機能

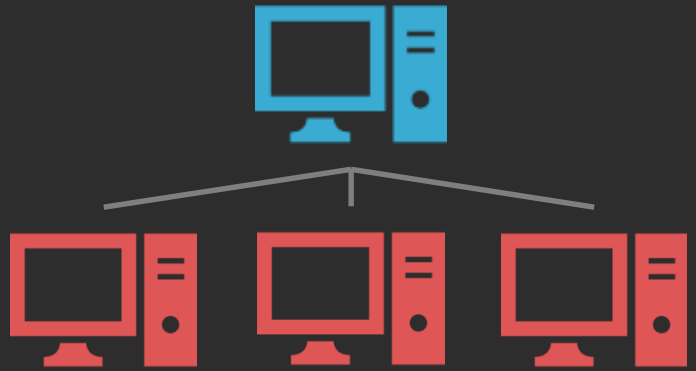
Agenda

- UE4のネットワーク機能
- Online MultiPlay Game の 概要
- Online MultiPlay Game の 設計
- Online MultiPlay Game の ネットワーク機能
- ネットワークTips

UE4のネットワーク機能

UE4 Networking

- Server / Client 構成



Listen Server

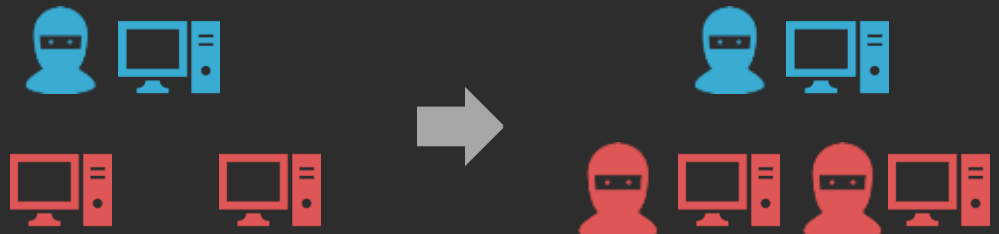
Dedicated Server

UE4 Networking

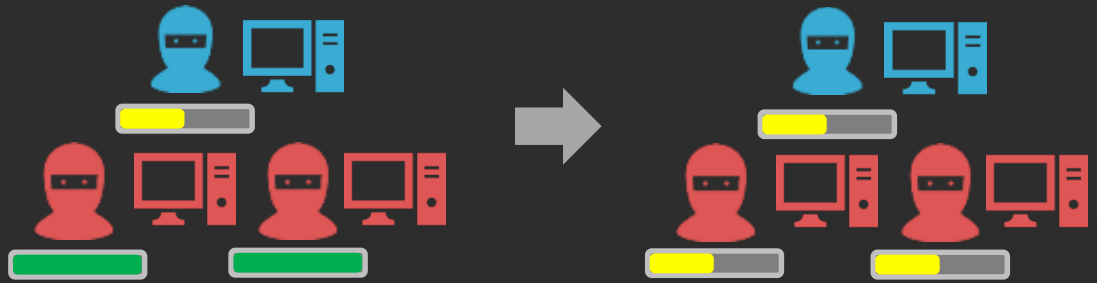
- Replicate

Server / Client間でのパラメータの複製、同期

Actor
Replicate



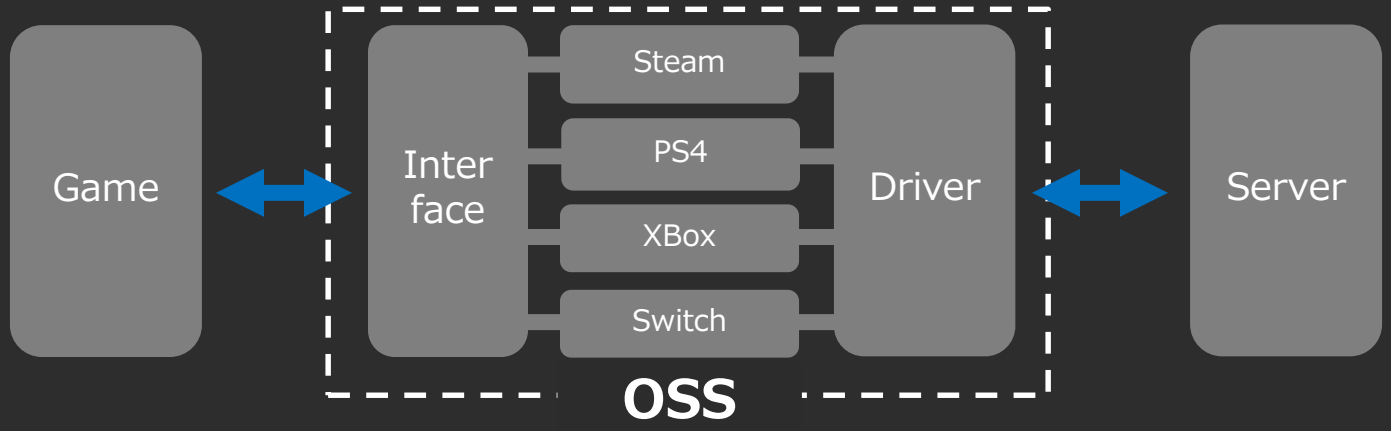
Property
Replicate



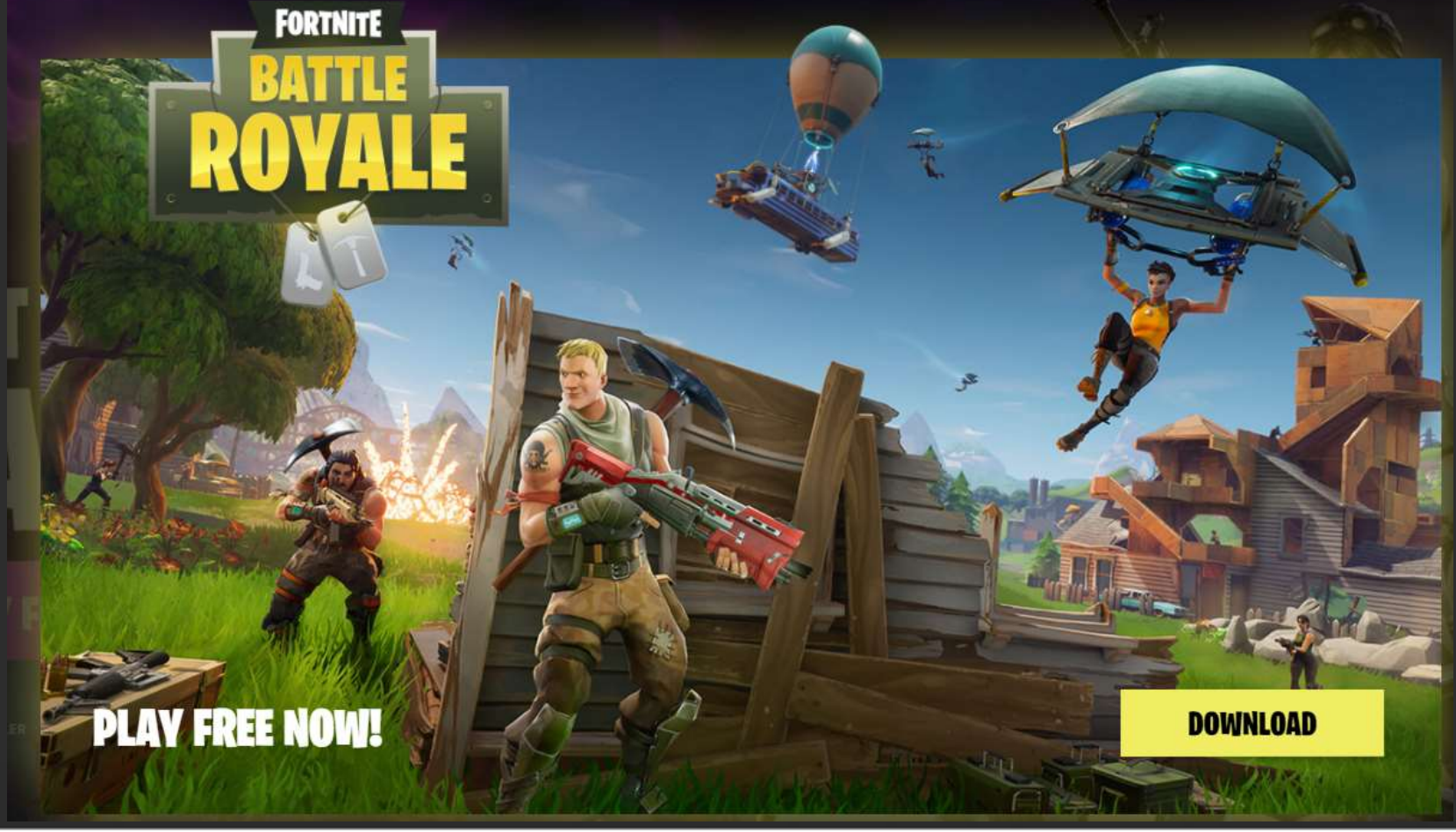
UE4 Networking

- OSS (Online Subsystem)

UE4が提供する、各種プラットフォームに対してオンライン通信機能を提供するための抽象化されたサブシステム、フレームワーク



Profile, Matchmake, Leaderboards, SharedCloud, Achievements, Voice, など...


FORTNITE
BATTLE
ROYALE
PLAY FREE NOW!
DOWNLOAD


MAIN MENU
HOST
JOIN
LEADERBOARDS
DEMOS
OPTIONS
QUIT
FREE FOR ALL
TEAM DEATHMATCH
NUMBER OF BOTS 1
Map Sanctuary
LAN ON
Record Demo OFF

Online MultiPlay Game の 概要

概要

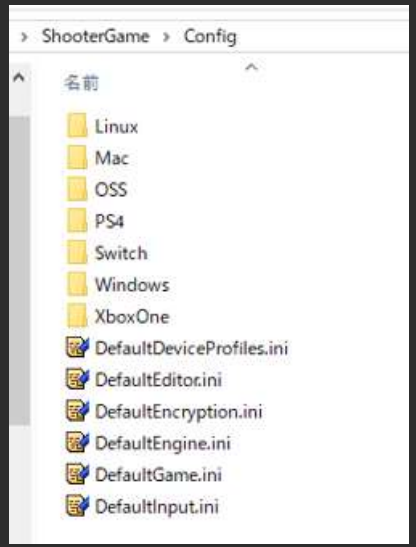
- ゲーム
  - FPS形式のガンシューティング
  - PvP または PvE

- 機能
  - 対戦
  - マッチング
  - ランキング
  - アチーブメント
  - リプレイ

概要

- コンテンツ
  - Blueprint + C++
  - 各種プラットフォーム対応
  - ネットワーク機能のサンプル

- 起動方法
  - PackageまたはVSからDevelepmentなどで起動
  - PC1台でもOnline接続の確認は可能




```
#if PLATFORM_PS4 || PLATFORM_XBOXONE || PLATFORM_SWITCH || SHOOTER_SIMULATE_CONSOLE_UI
    #define SHOOTER_CONSOLE_UI 1
#else
    #define SHOOTER_CONSOLE_UI 0
#endif
```

ゲームフロー

Menu → Matching → Game → Menu

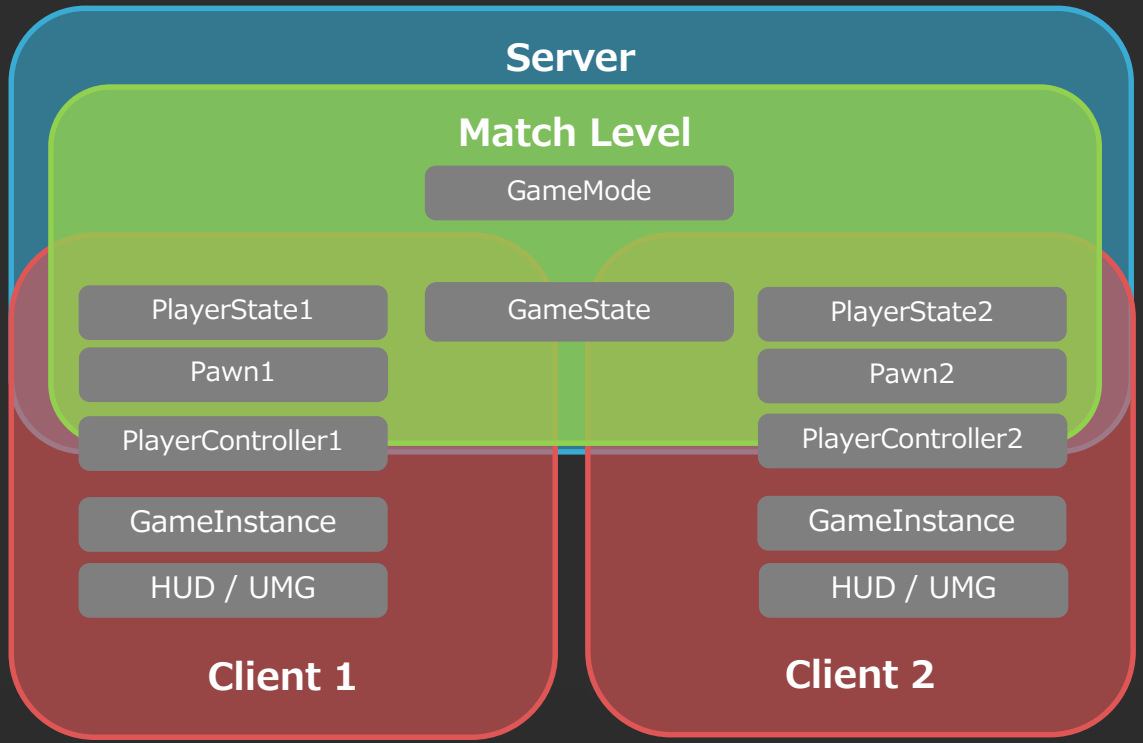
Online MultiPlay Game の 設計

シングルプレイの場合は

Standalone Game

マルチプレイの場合は

Multiplay game



Game Design

- Core

GameEngine, GameInstance

- System

GameMode, GameState, GameSession, GameViewportClient, GameOnlineSessionClient

- Player Contents

PlayerState, PlayerController, PlayerCameraManager

- Game Contents

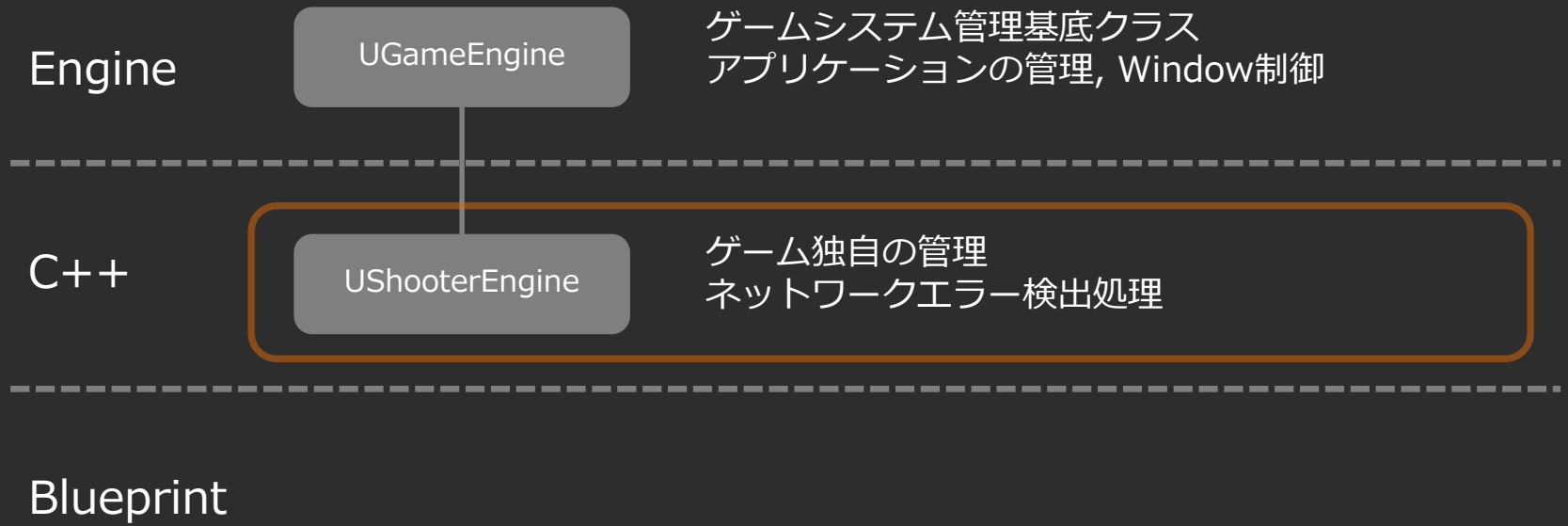
Character, Weapon, Bot, Item

- Misc

GameUserSettings, CheatManager, SaveGame

※ Shooter Gameでの例です…

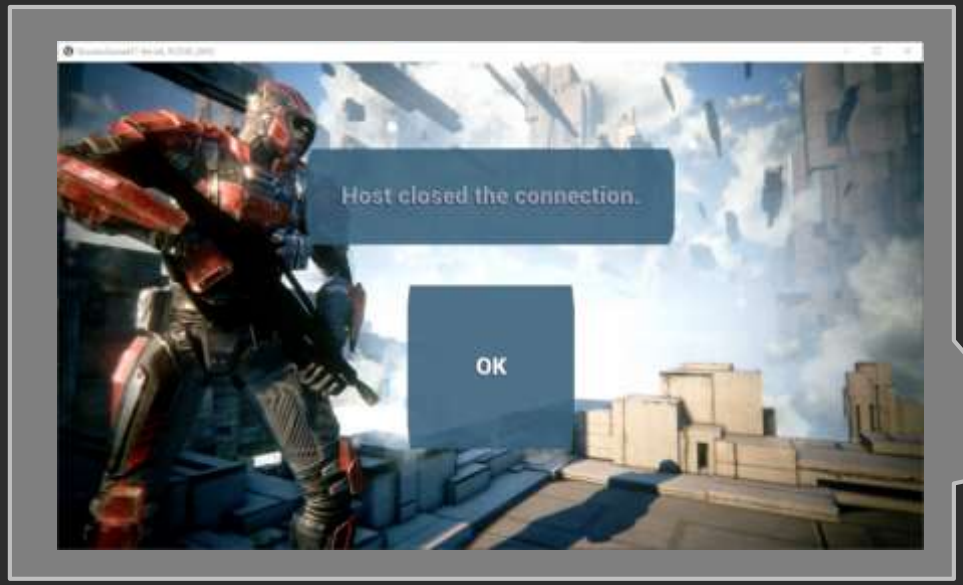
GameEngine

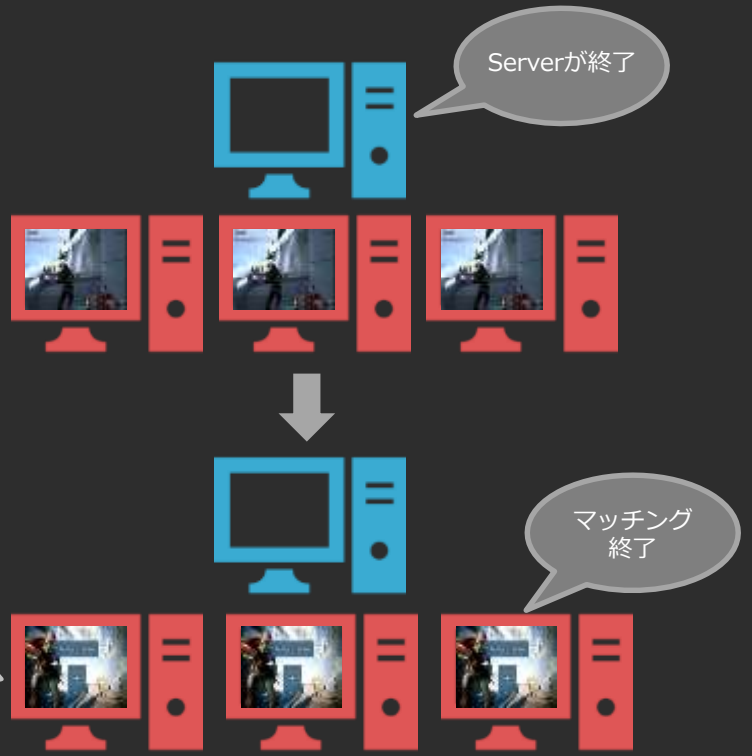

Engine
UGameEngine
ゲームシステム管理基底クラス
アプリケーションの管理, Window制御
C++
UShooterEngine
ゲーム独自の管理
ネットワークエラー検出処理
Blueprint

GameInstance

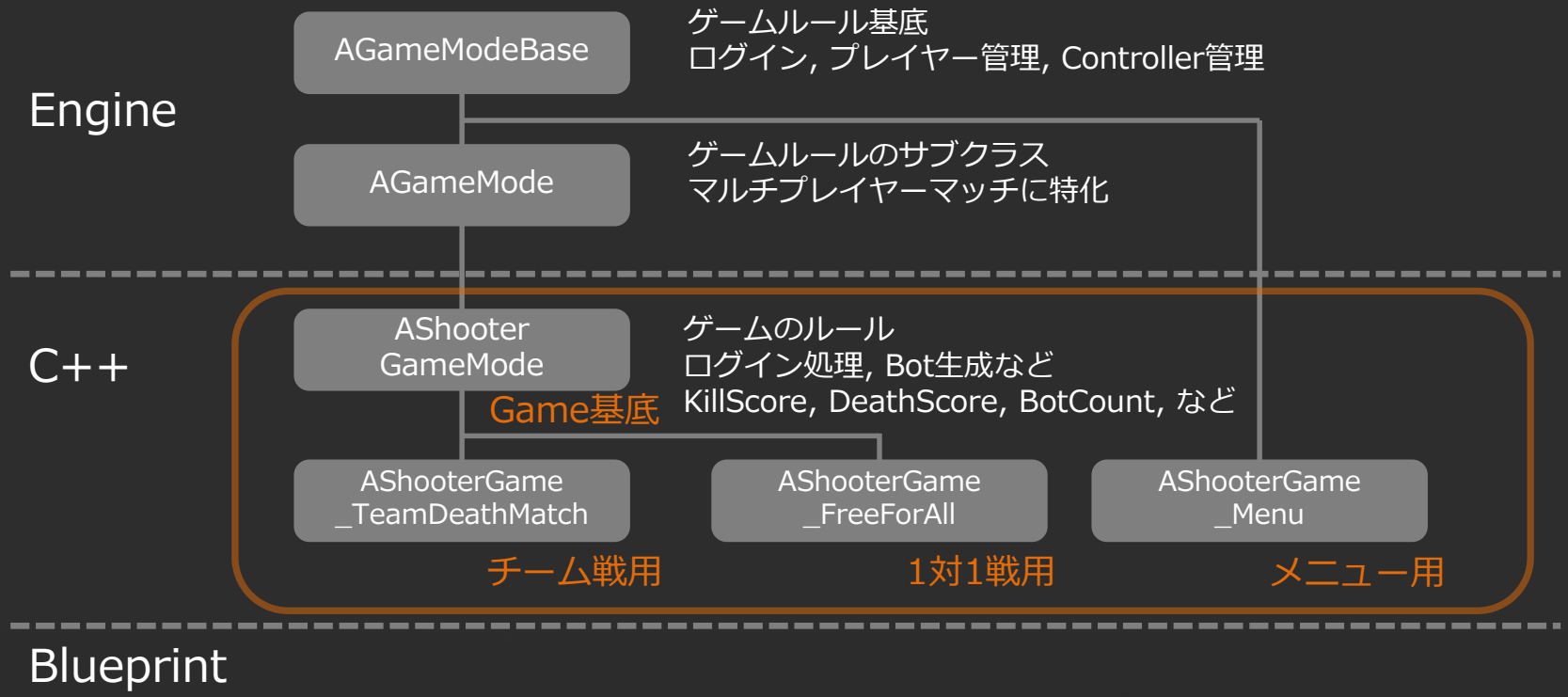
GameInstance

- サーバーとの接続エラー表示



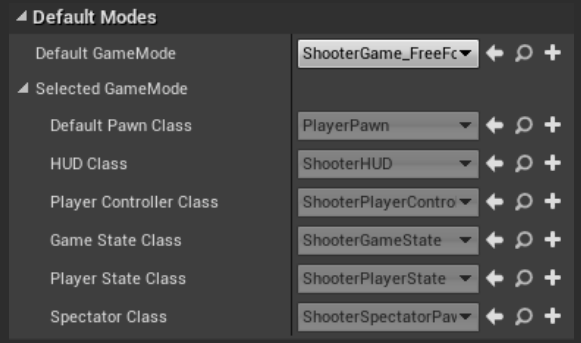


GameMode

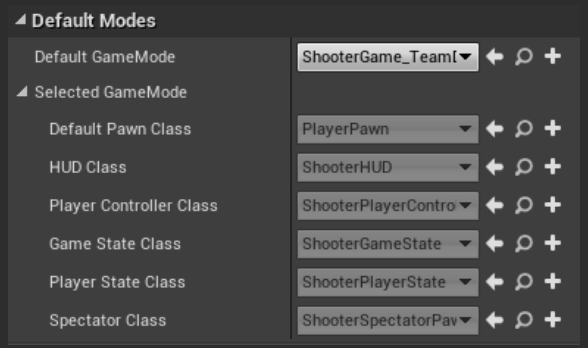

Engine
C++
Blueprint
AGameModeBase
ゲームルール基底
ログイン, プレイヤー管理, Controller管理
AGameMode
ゲームルールのサブクラス
マルチプレイヤーマッチに特化
AShooter
GameMode
ゲームのルール
ログイン処理, Bot生成など
KillScore, DeathScore, BotCount, など
Game基底
AShooterGame
_TeamDeathMatch
AShooterGame
_FreeForAll
AShooterGame
_Menu
チーム戦用
1対1戦用
メニュー用

GameMode

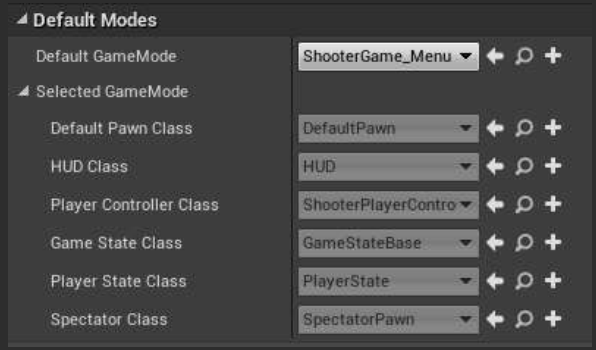
- World毎に設定
  - ゲームルール毎に設定



チーム戦用

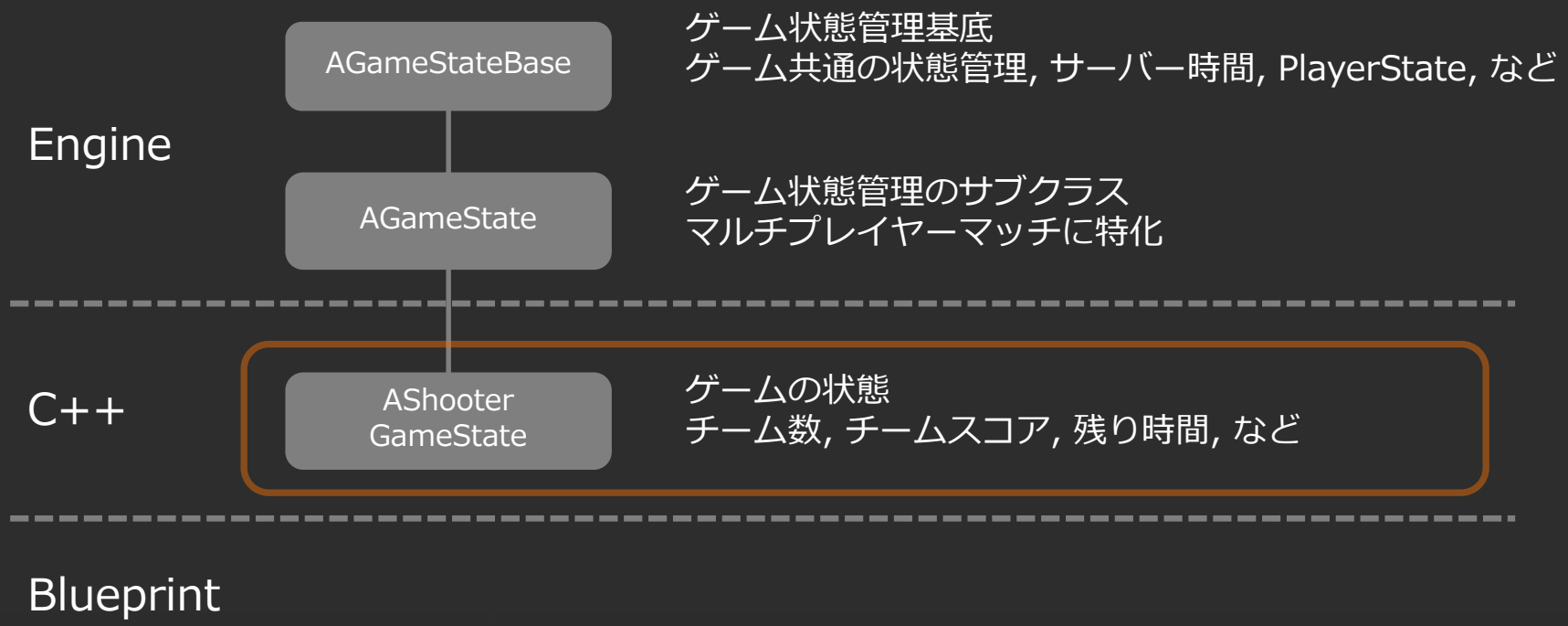


1対1戦用



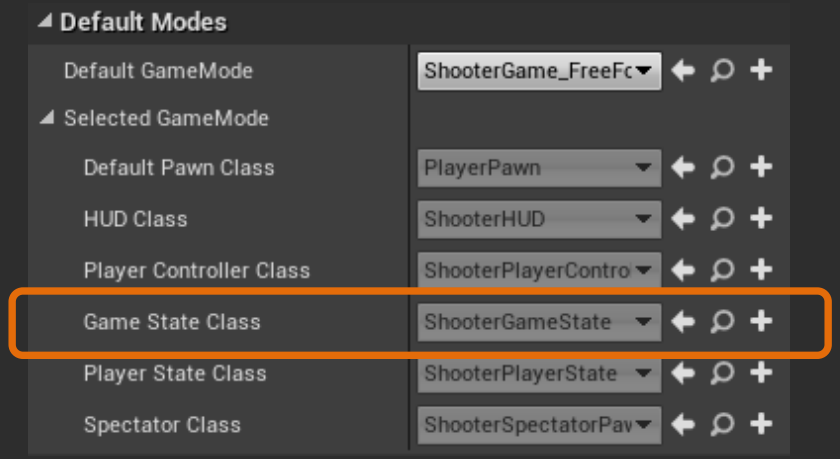
メニュー用

GameState

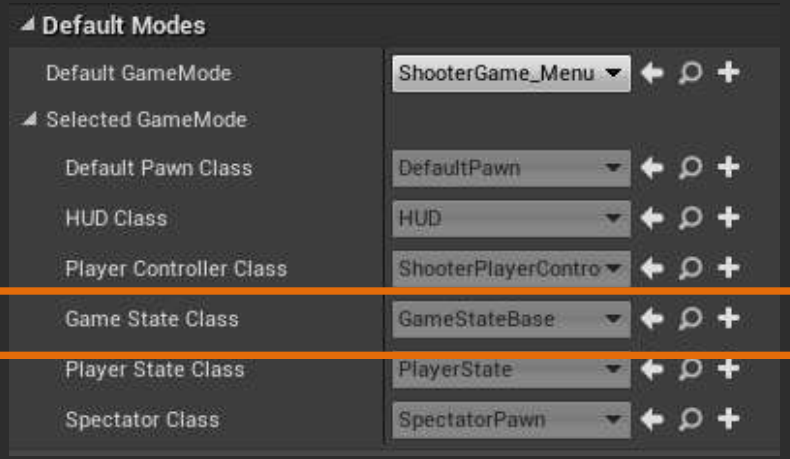

AGameStateBase
ゲーム状態管理基底
ゲーム共通の状態管理, サーバー時間, PlayerState, など
Engine
AGameState
ゲーム状態管理のサブクラス
マルチプレイヤーマッチに特化
C++
AShooter
GameState
ゲームの状態
チーム数, チームスコア, 残り時間, など
Blueprint

GameState

- GameMode毎に設定



1対1戦用 GameMode

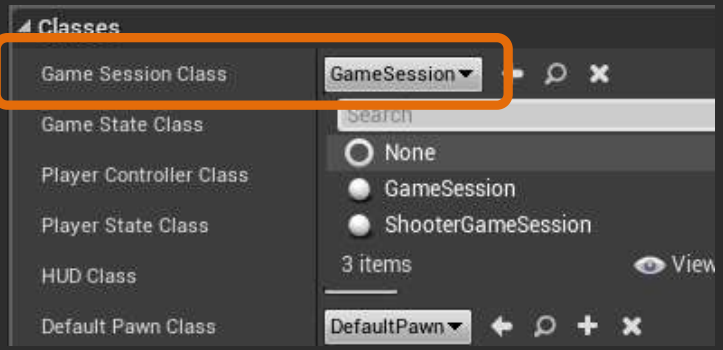


メニュー用 GameMode

GameSession

GameSession

- GameMode毎に設定, 未設定も可能



GameMode

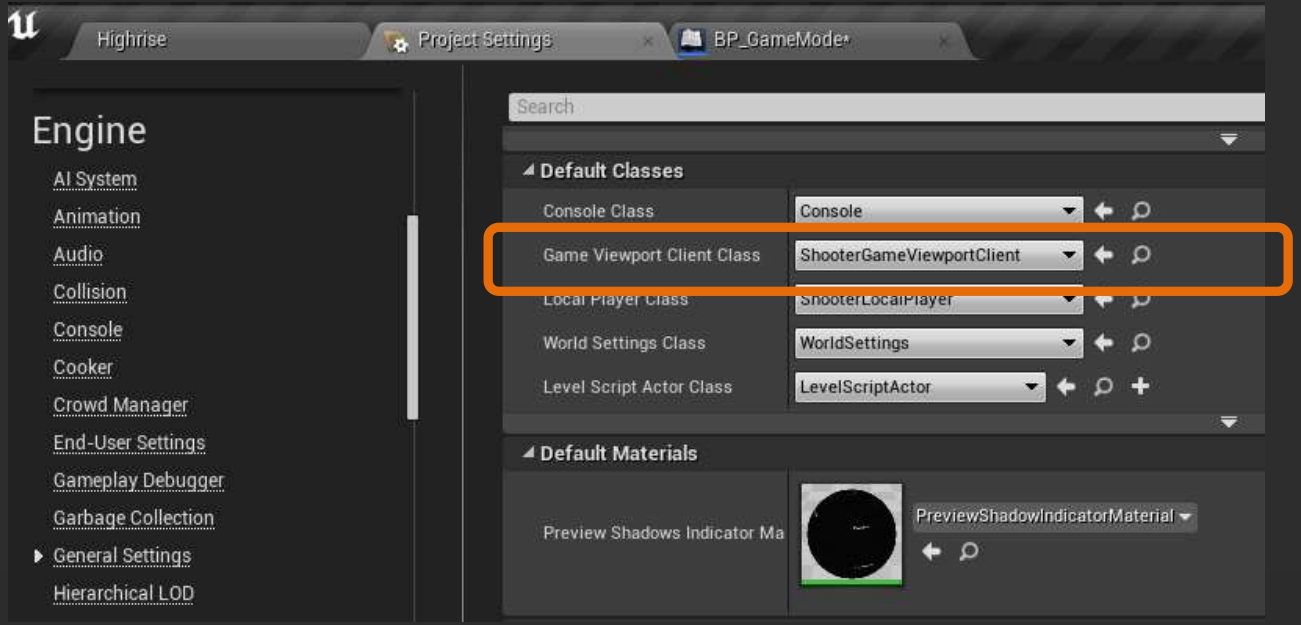
```
/** Class of GameSession, which handles login approval and online game
UPROPERTY(EditAnywhere, BlueprintReadWrite, Category=Classes)
TSubclassOf<AGameSession> GameSessionClass;

/** Class of GameState associated with this GameMode. */
UPROPERTY(EditAnywhere, NoClear, BlueprintReadOnly, Category=Classes)
TSubclassOf<AGameStateBase> GameStateClass;

/** The class of PlayerController to spawn for players logging in. */
UPROPERTY(EditAnywhere, NoClear, BlueprintReadOnly, Category=Classes)
TSubclassOf<APlayerController> PlayerControllerClass;
```

GameSessionClassの選択は任意
("NoClear"を付与するとNoneが未選択)

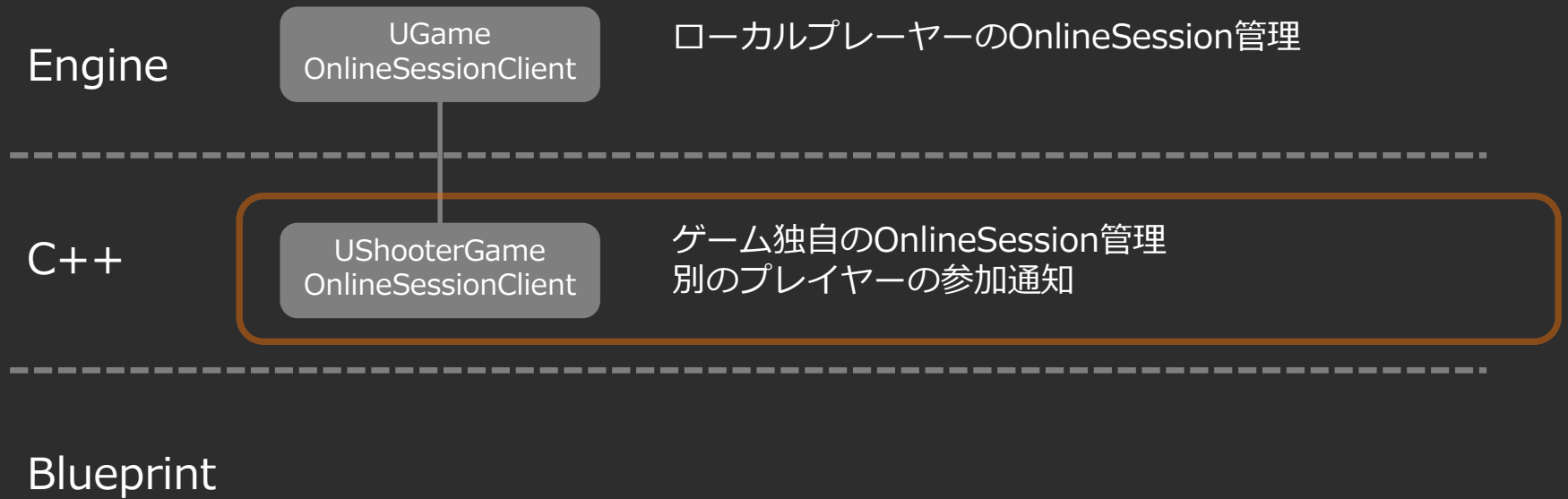
GameViewportClient

GameViewportClient

- ゲーム中で1つのみ



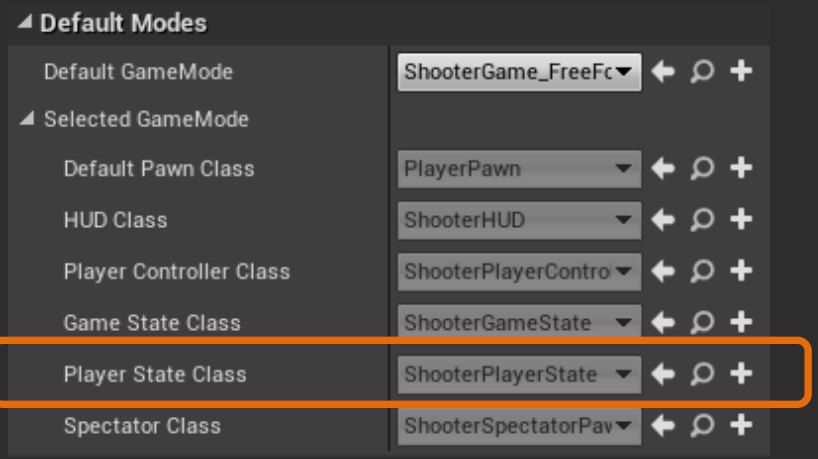
GameOnlineSessionClient



PlayerState

PlayerState

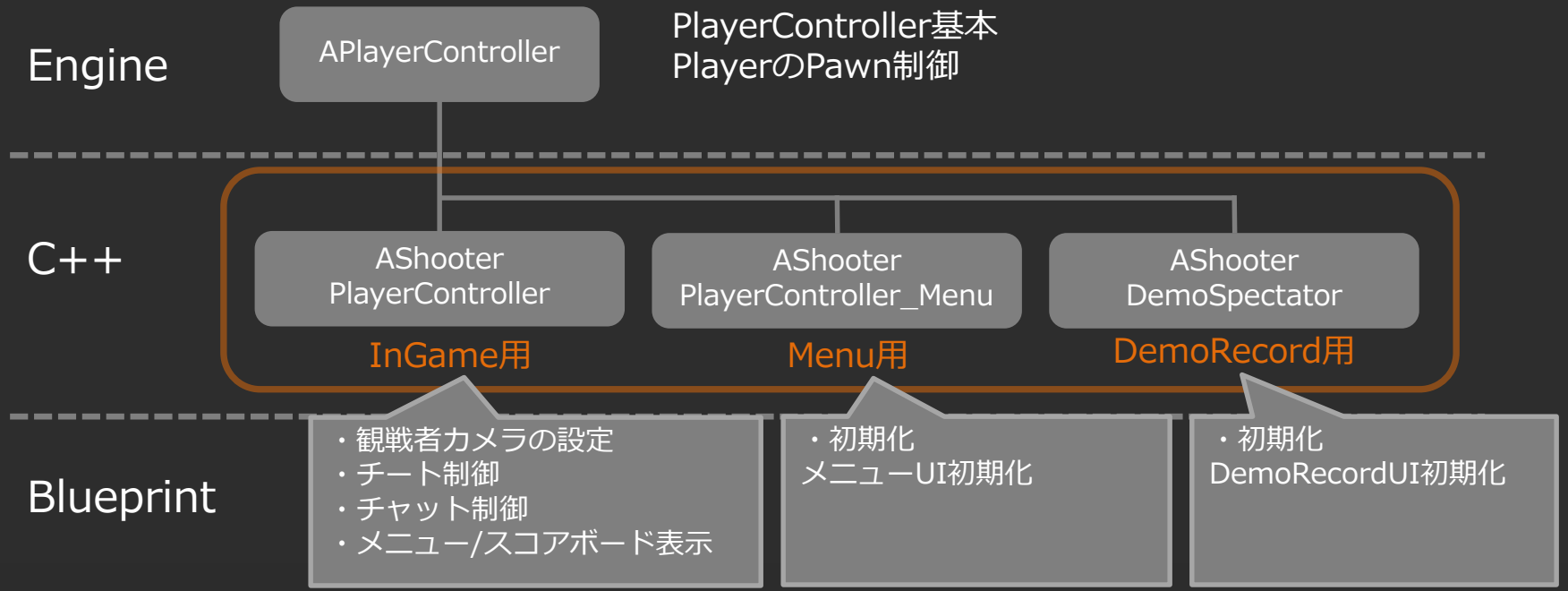
- GameMode毎に設定



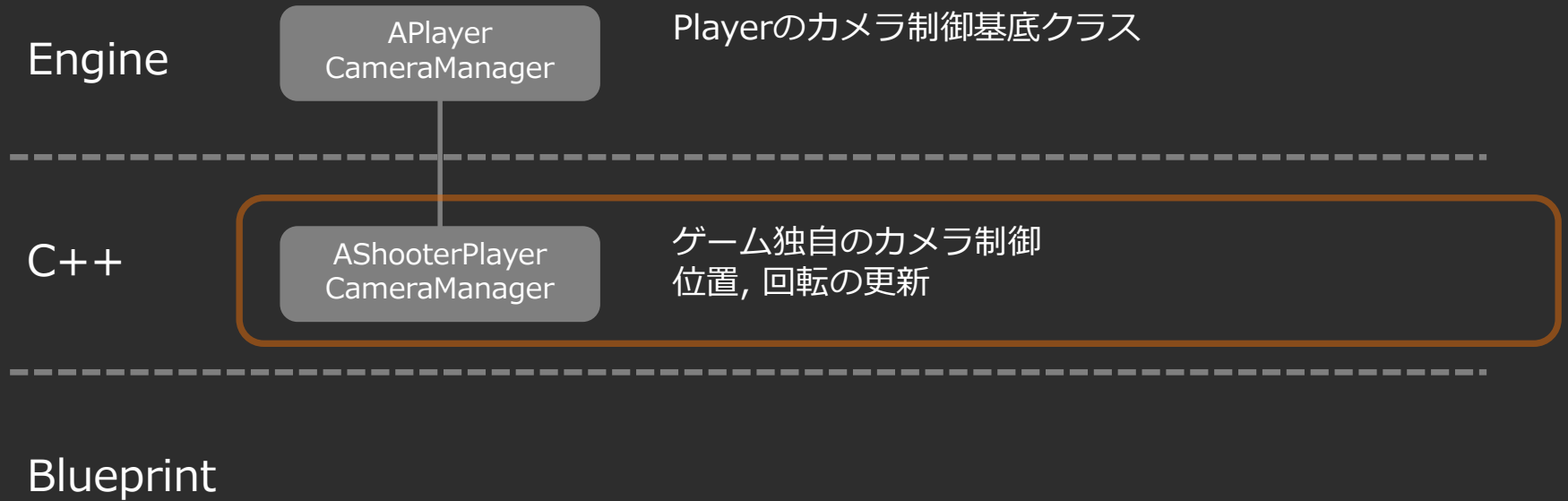
1対1戦用 GameMode

情報の管理/制御先に注意

PlayerController

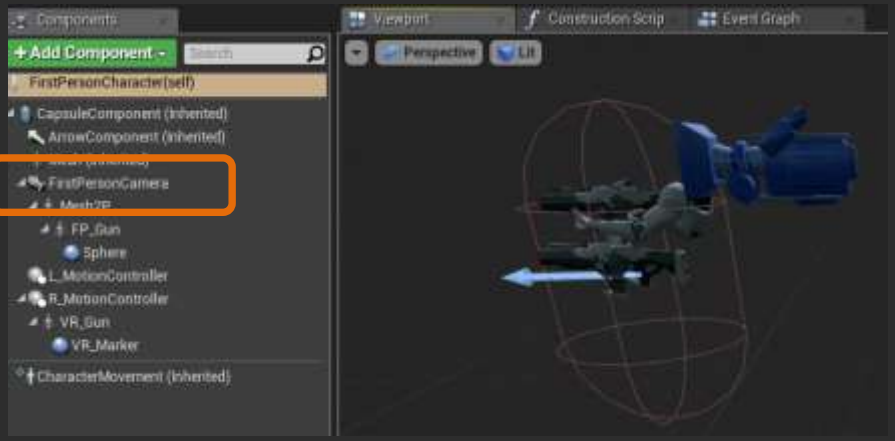


PlayerCameraManager



PlayerCameraManager

- FirstPersonTempleteと異なりCharacterにCameraがない
  - ViewTargetを更新、PCMはキャラクターに追従し同期



FirstPerson

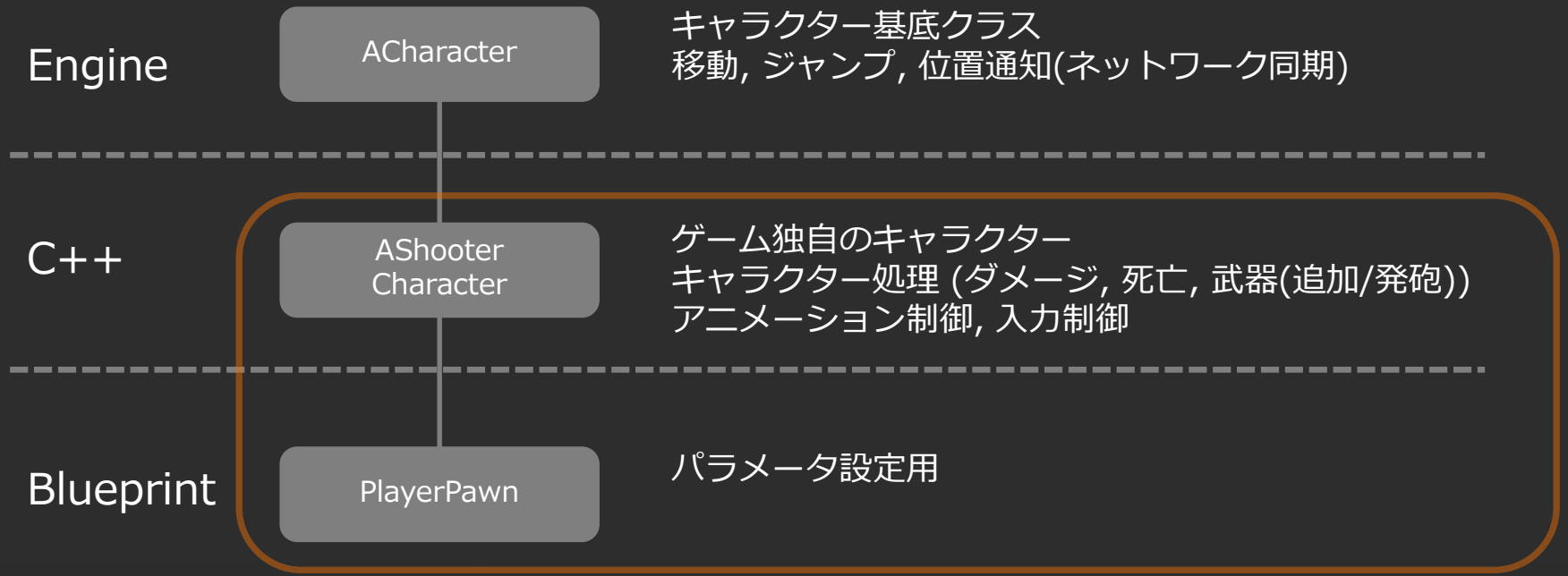
ShooterGame

Character

- プレイヤーキャラクター
  - 移動
  - ジャンプ
  - ターン
  - 走行
  - 射撃
  - エイム
  - 武器切替
  - リロード

主に入力の制御処理

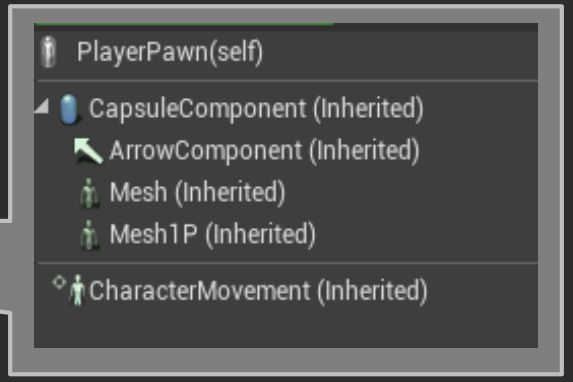
Character

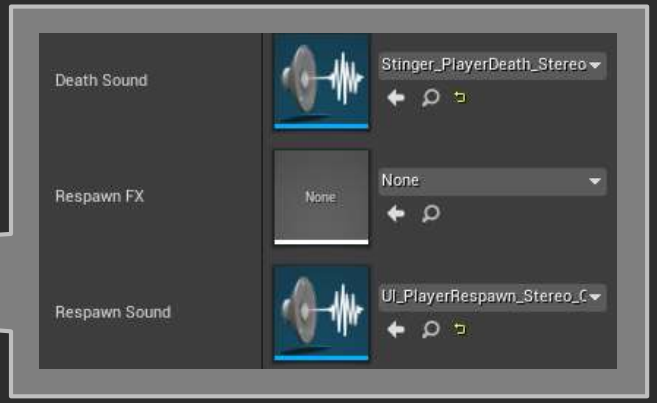

Engine
ACharacter
キャラクター基底クラス
移動, ジャンプ, 位置通知(ネットワーク同期)
C++
AShooter
Character
ゲーム独自のキャラクター
キャラクター処理 (ダメージ, 死亡, 武器(追加/発砲))
アニメーション制御, 入力制御
Blueprint
PlayerPawn
パラメータ設定用

Character - Input

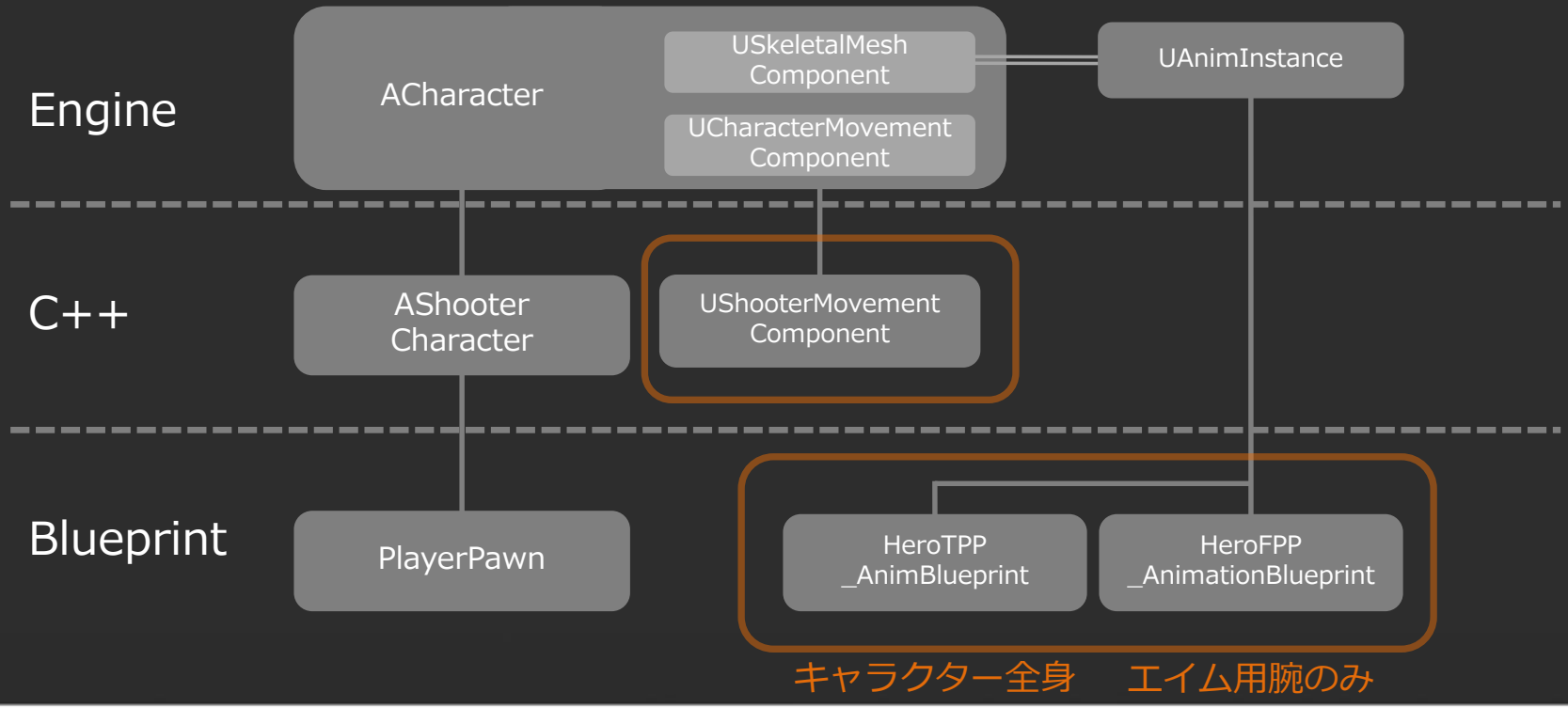
```
void AShooterCharacter::SetupPlayerInputComponent(class UInputComponent* PlayerInputComponent)
{
    check(PlayerInputComponent);
    PlayerInputComponent->BindAxis("MoveForward", this, &AShooterCharacter::MoveForward);
    PlayerInputComponent->BindAxis("MoveRight", this, &AShooterCharacter::MoveRight);
    PlayerInputComponent->BindAxis("MoveUp", this, &AShooterCharacter::MoveUp);
    PlayerInputComponent->BindAxis("Turn", this, &APawn::AddControllerYawInput);
    PlayerInputComponent->BindAxis("TurnRate", this, &AShooterCharacter::TurnAtRate);
    PlayerInputComponent->BindAxis("LookUp", this, &APawn::AddControllerPitchInput);
    PlayerInputComponent->BindAxis("LookUpRate", this, &AShooterCharacter::LookUpAtRate);

    PlayerInputComponent->BindAction("Fire", IE_Pressed, this, &AShooterCharacter::OnStartFire);
    PlayerInputComponent->BindAction("Fire", IE_Released, this, &AShooterCharacter::OnStopFire);

    PlayerInputComponent->BindAction("Targeting", IE_Pressed, this, &AShooterCharacter::OnStartTargeting);
    PlayerInputComponent->BindAction("Targeting", IE_Released, this, &AShooterCharacter::OnStopTargeting);

    PlayerInputComponent->BindAction("NextWeapon", IE_Pressed, this, &AShooterCharacter::OnNextWeapon);
    PlayerInputComponent->BindAction("PrevWeapon", IE_Pressed, this, &AShooterCharacter::OnPrevWeapon);

    PlayerInputComponent->BindAction("Reload", IE_Pressed, this, &AShooterCharacter::OnReload);

    PlayerInputComponent->BindAction("Jump", IE_Pressed, this, &AShooterCharacter::OnStartJump);
    PlayerInputComponent->BindAction("Jump", IE_Released, this, &AShooterCharacter::OnStopJump);

    PlayerInputComponent->BindAction("Run", IE_Pressed, this, &AShooterCharacter::OnStartRunning);
    PlayerInputComponent->BindAction("RunToggle", IE_Pressed, this, &AShooterCharacter::OnStartRunningToggle);
    PlayerInputComponent->BindAction("Run", IE_Released, this, &AShooterCharacter::OnStopRunning);
}
```

Character

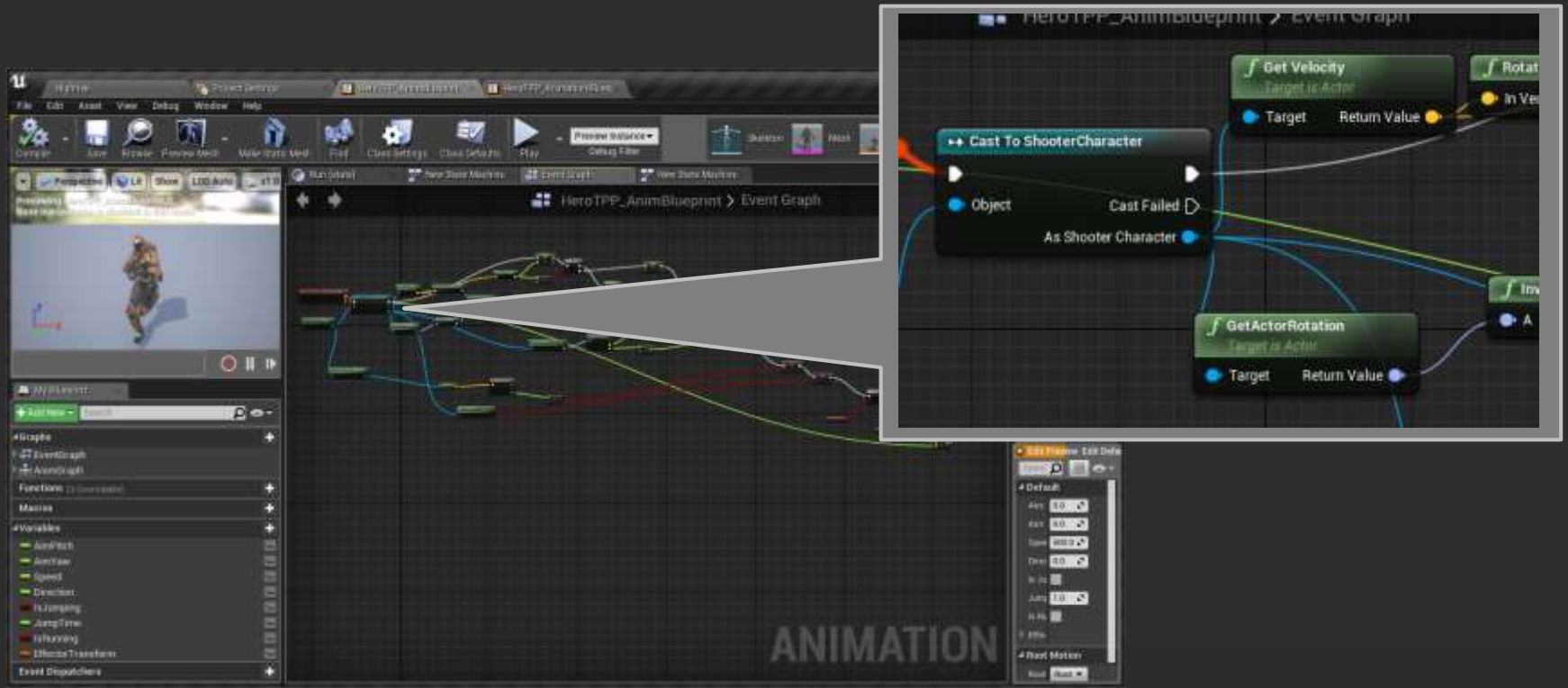

PlayerPawn(self)
CapsuleComponent (Inherited)
ArrowComponent (Inherited)
Mesh (Inherited)
Mesh1P (Inherited)
CharacterMovement (Inherited)


Death Sound
Stinger_PlayerDeath_Stereo
Respawn FX
None
None
Respawn Sound
UI_PlayerRespawn_Stereo_C

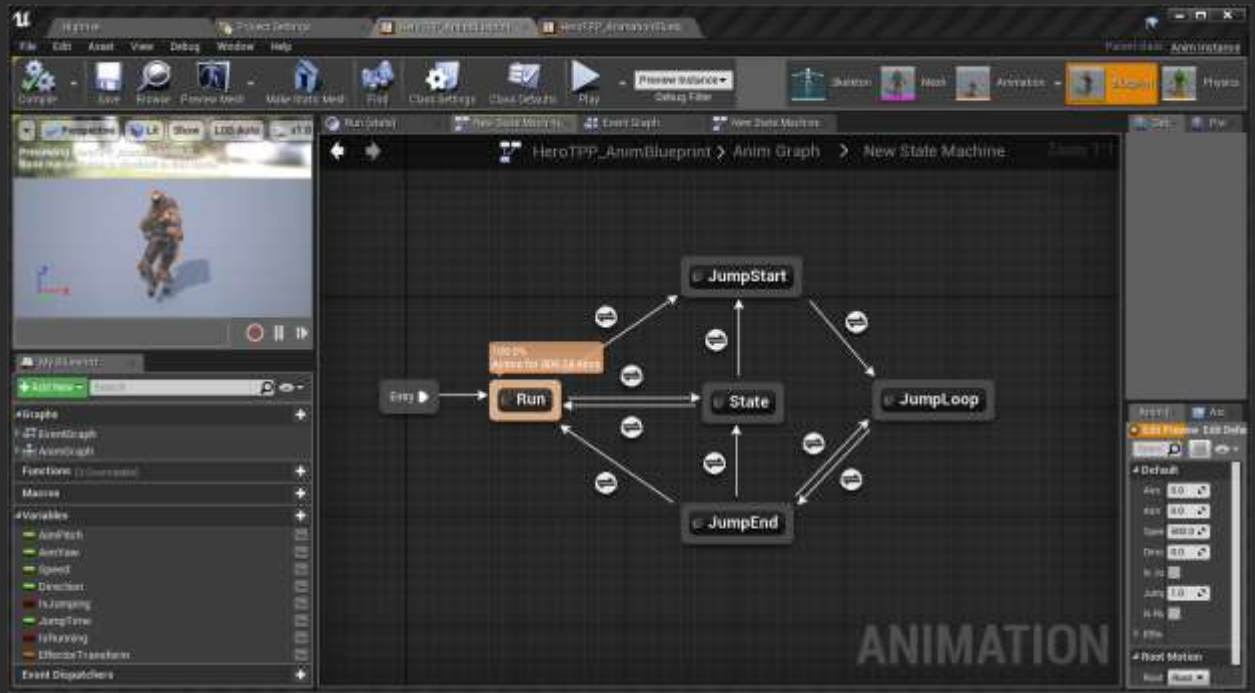
Character


Engine
C++
Blueprint
ACharacter
USkeletalMesh Component
UCharacterMovement Component
UAnimInstance
AShooter Character
UShooterMovement Component
PlayerPawn
HeroTPP _AnimBlueprint
HeroFPP _AnimationBlueprint
キャラクター全身
エイム用腕のみ

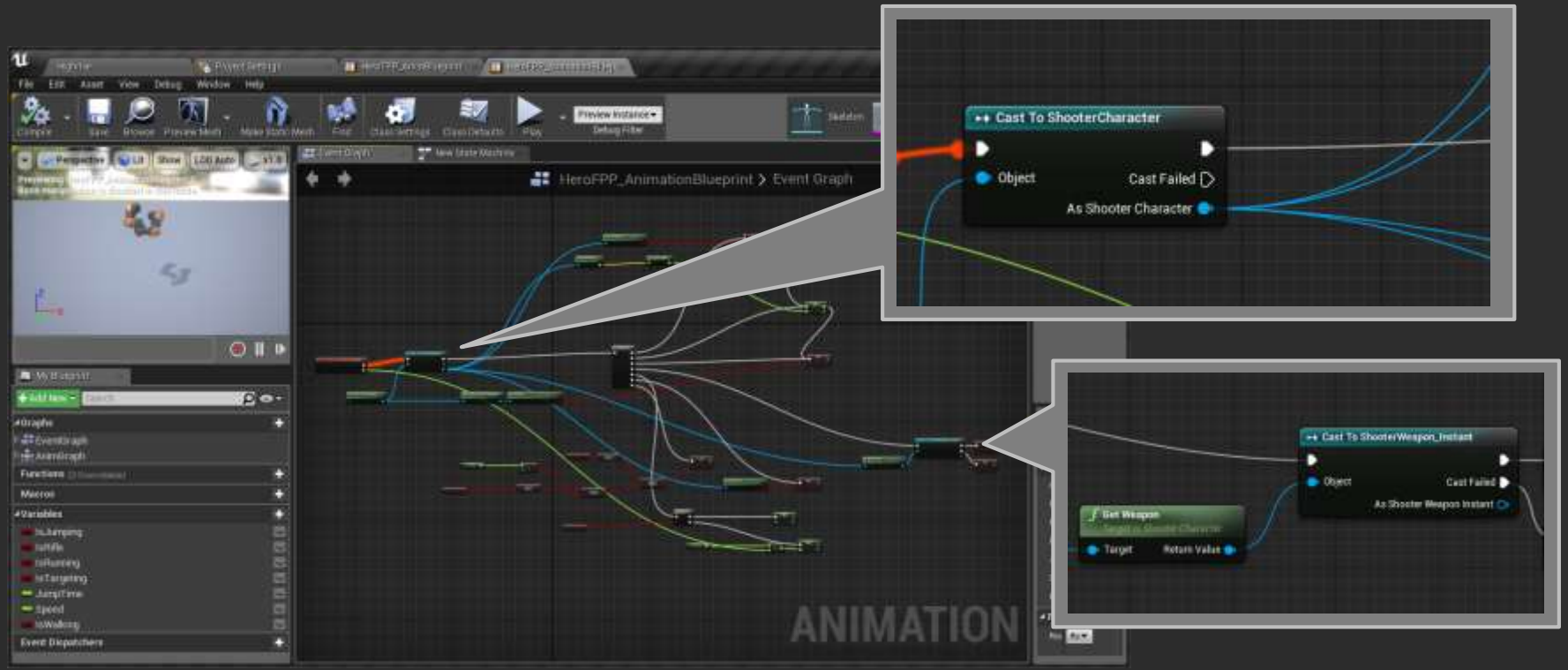
Character - Animation



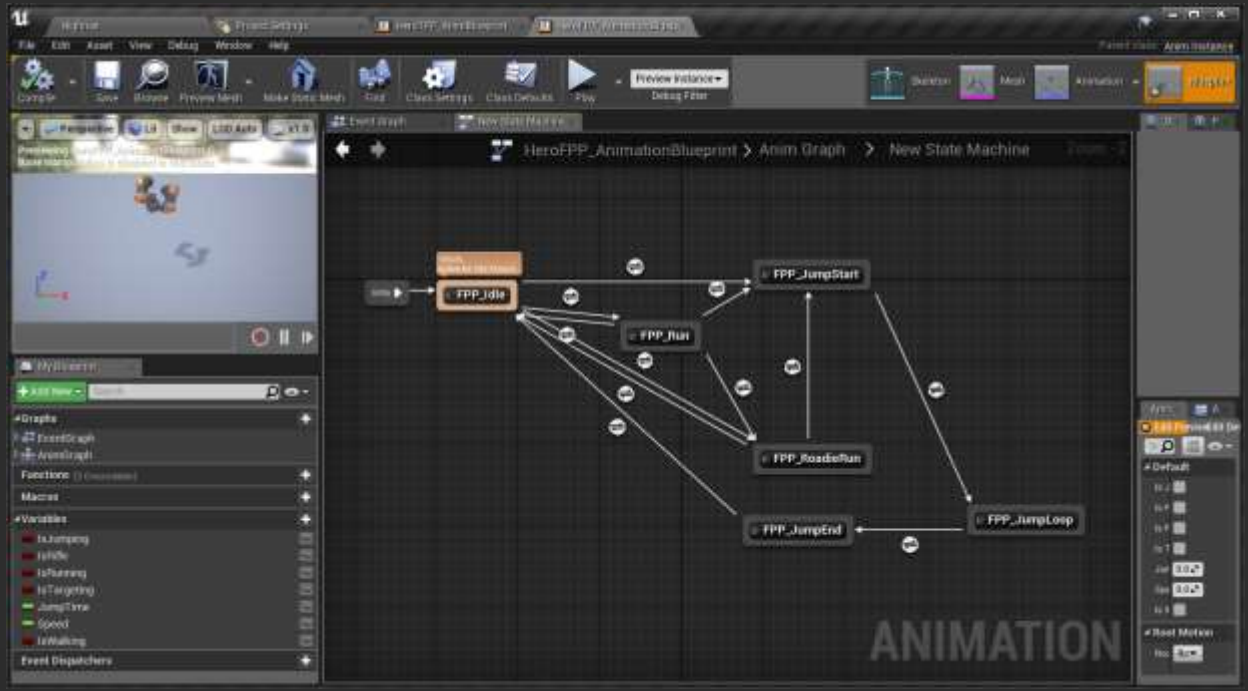
Character - Animation



Character - Animation



Character - Animation



Weapon

Gun

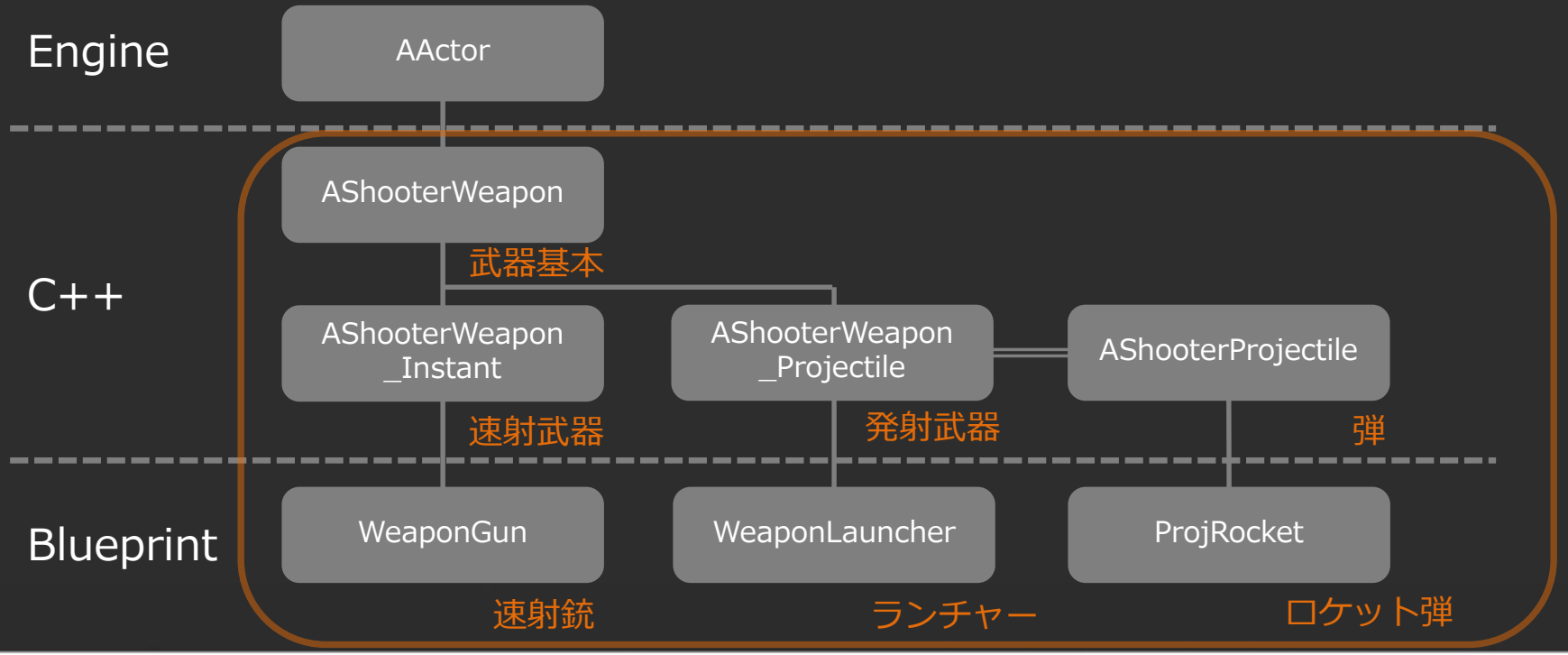
Launcher

2種類の武器(Gun/Launcher)
- 攻撃がヒットしたらダメージを与える
- 攻撃がヒットしたら指定のエフェクトを発生させる
- 攻撃を行ったら指定のサウンドを再生する

• Gun
入力している間攻撃し続ける(速射式)
攻撃の判定はTraceで行いDamageを与える

・Launcher
入力する毎に弾を発射する(発射式)
弾がHitした位置を基準に範囲内でDamageを与える

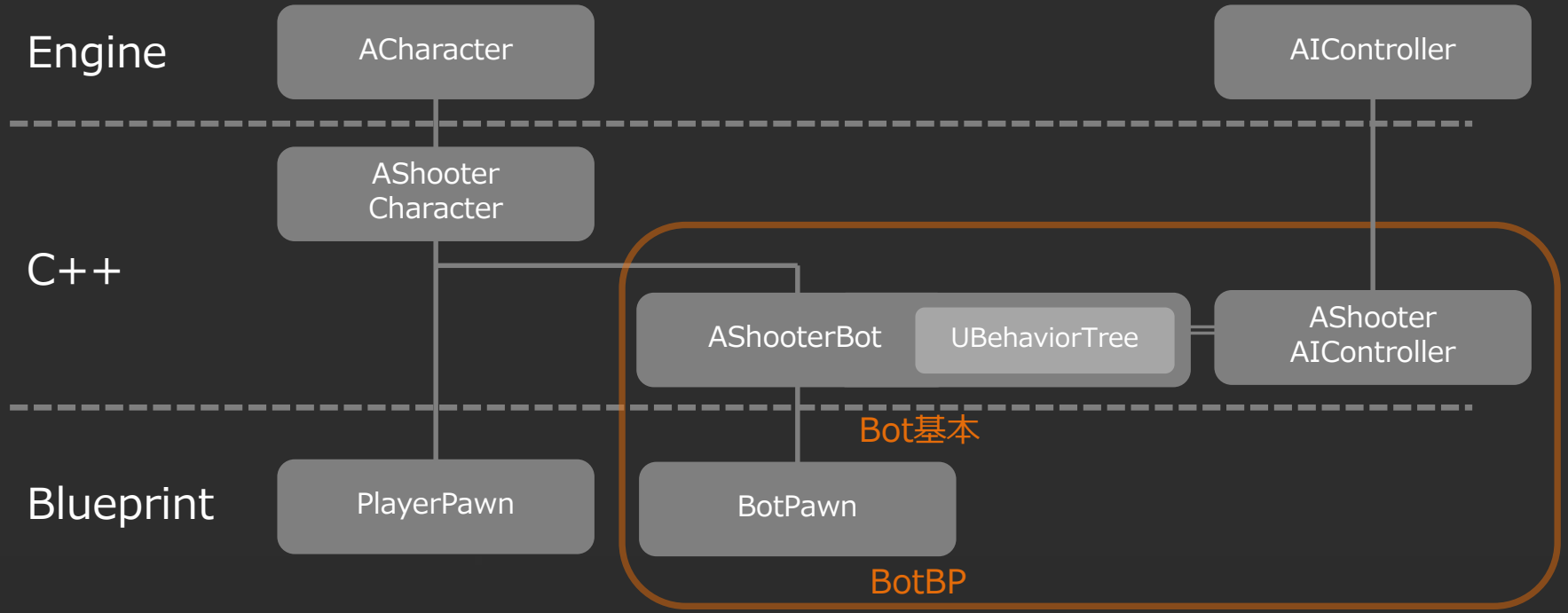
Weapon


Engine
C++
Blueprint
AActor
AShooterWeapon
武器基本
AShooterWeapon
_Instant
AShooterWeapon
_Projectile
AShooterProjectile
速射武器
発射武器
弾
WeaponGun
WeaponLauncher
ProjRocket
速射銃
ランチャー
ロケット弾

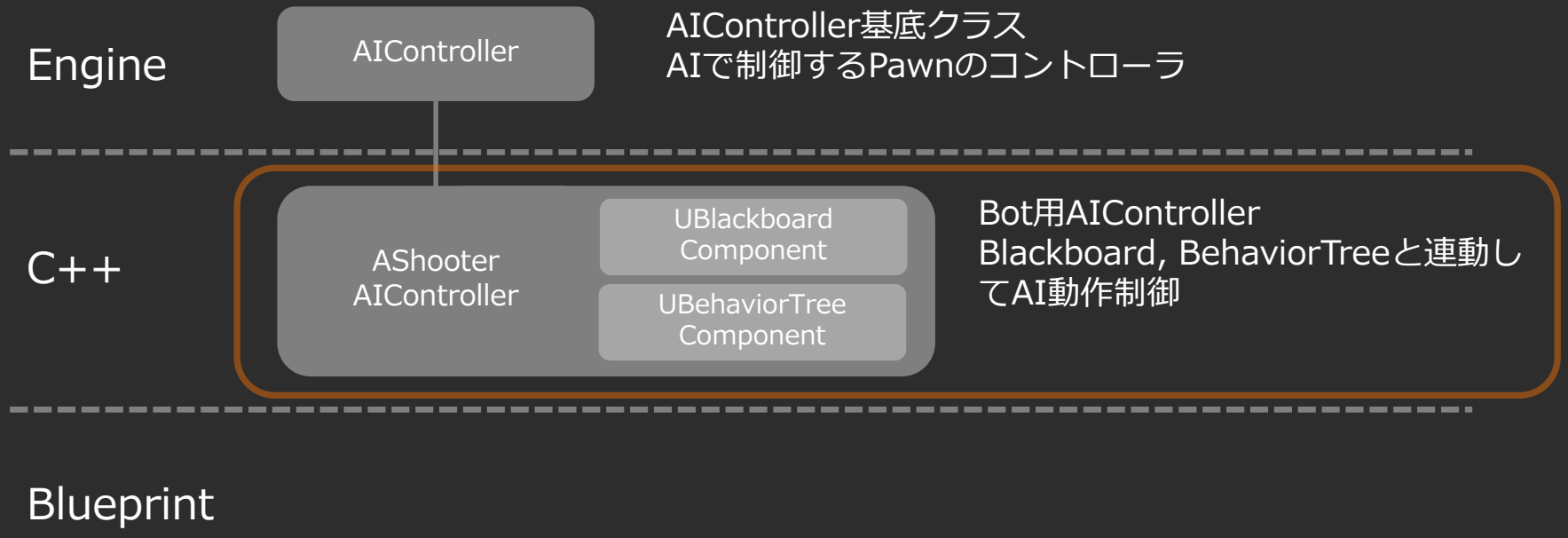
Bot

- プレイヤーと同型のキャラクター
- 動作は全てAI
- Gunのみ使用(Launcherは未使用)

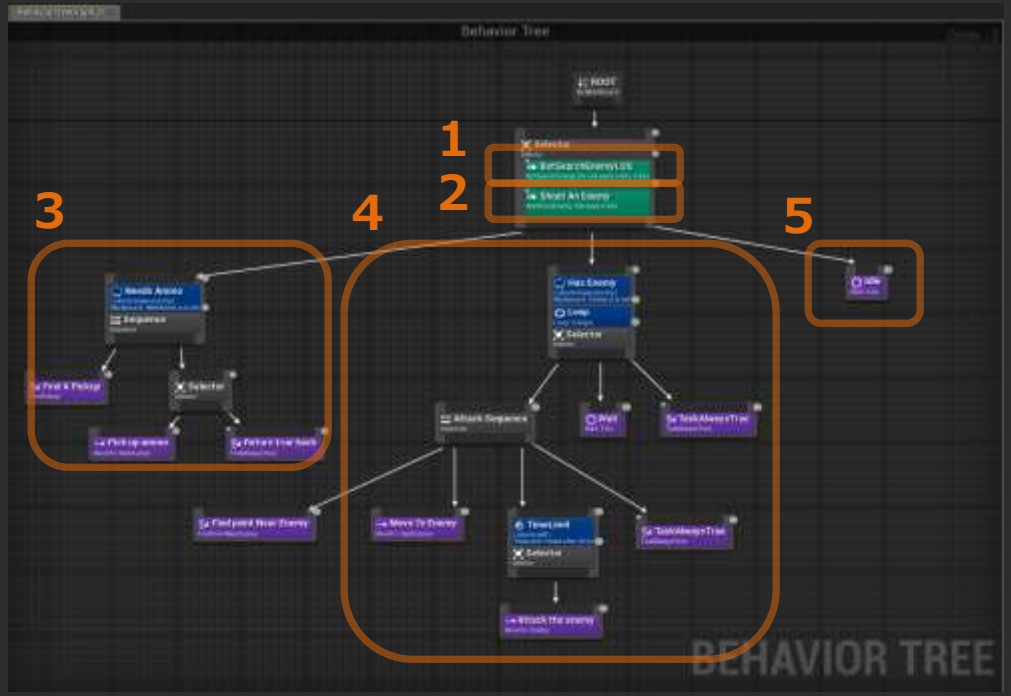
Bot


Engine
C++
Blueprint
ACharacter
AIController
AShooter
Character
AShooterBot
UBehaviorTree
AShooter
AIController
Bot基本
PlayerPawn
BotPawn
BotBP

Bot - AIController



Bot - Behavior Tree

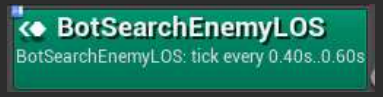


行動パターン
1. 索敵：最も近い敵を見つける
2. 攻撃：ターゲットに攻撃
3. 補充：弾薬を見つけて移動
4. 移動：攻撃対象まで移動
5. 待機：待機

毎フレーム実行する処理が多い

Bot - Service

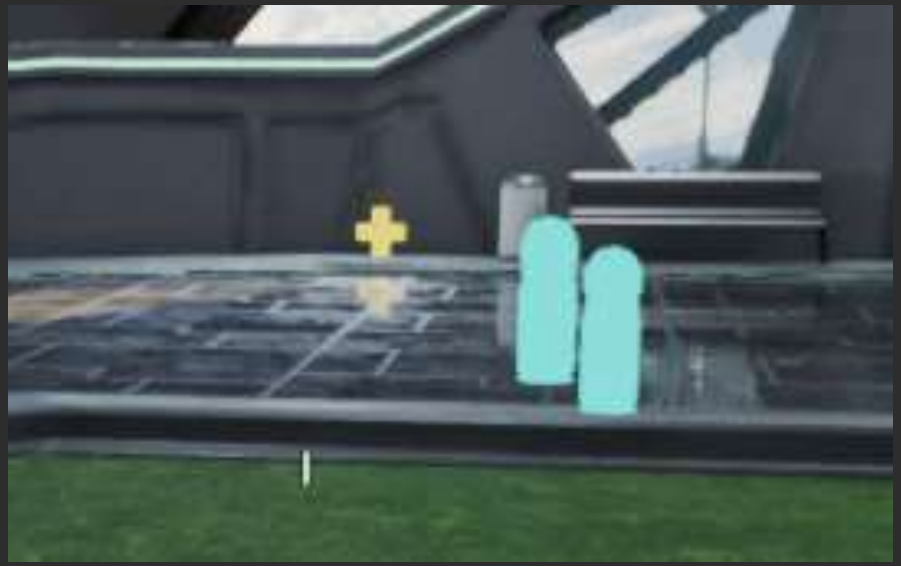
索敵



射撃

ShooterAIController
(C++)

Item

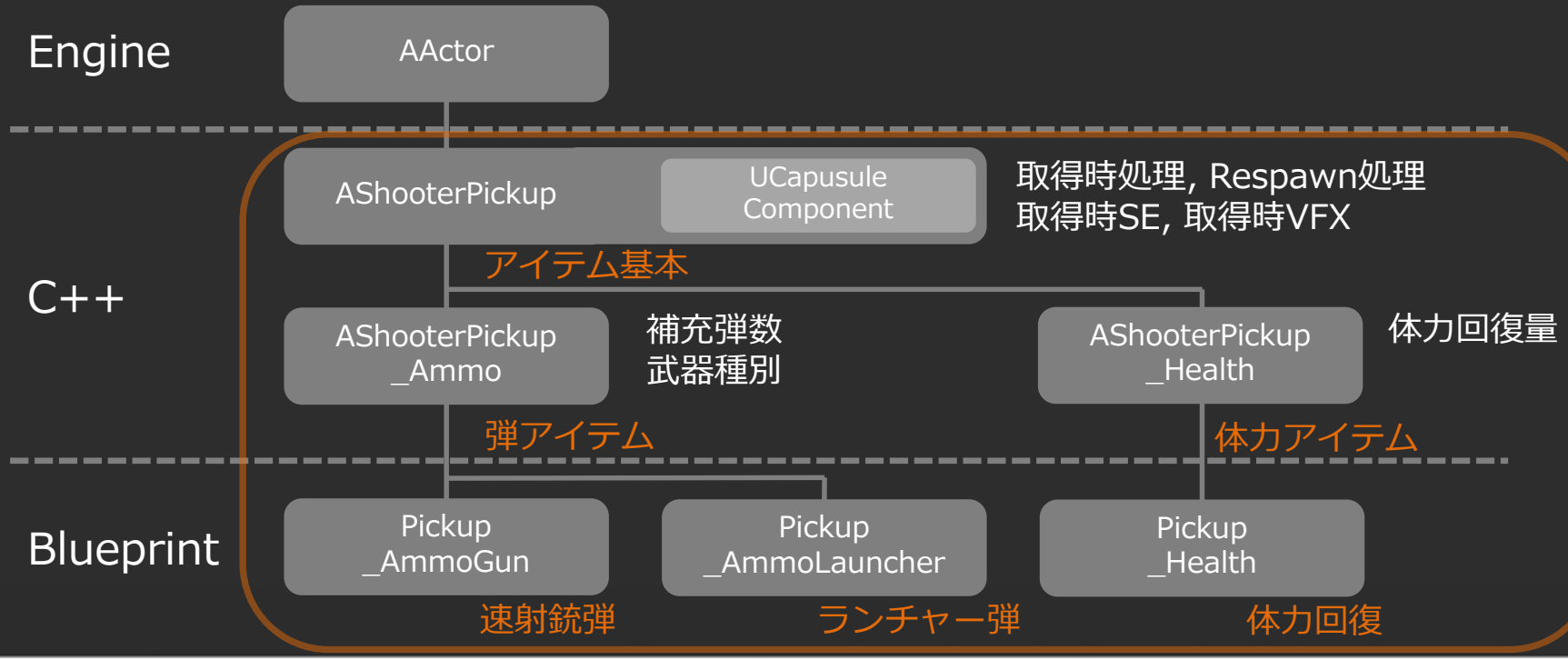


- 2種類のアイテム
  - 弾薬回復(Gun, Launcher)
  - 体力回復

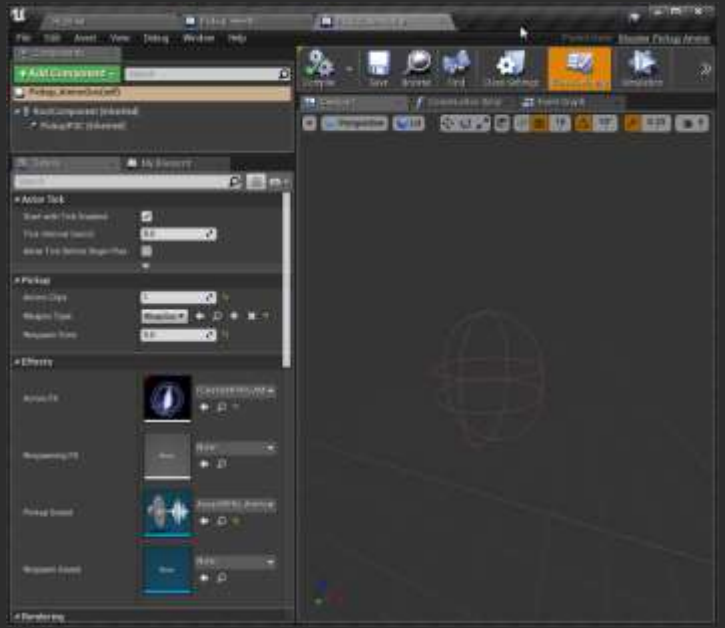
- 一定時間で自動的にRespawn
- 取得時にSE再生
- 取得時にVFX再生

など

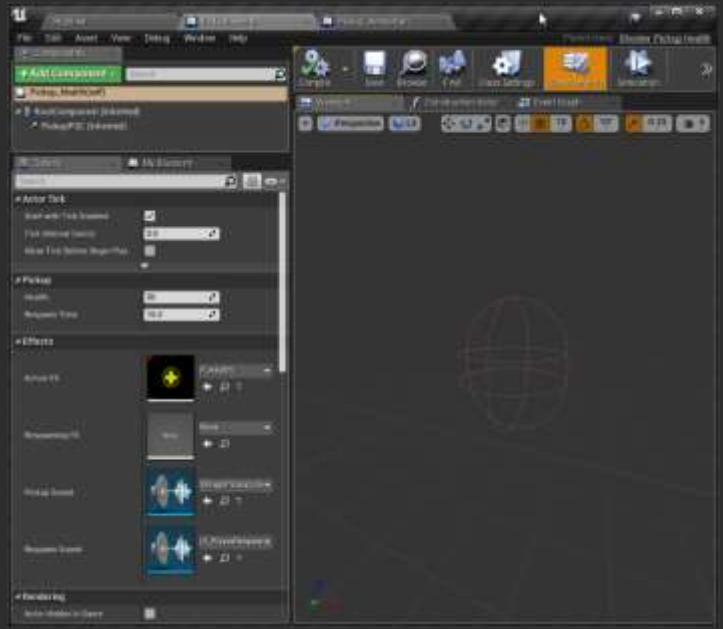
Item


Engine
C++
Blueprint
AActor
AShooterPickup
UCapusule Component
取得時処理, Respawn処理
取得時SE, 取得時VFX
アイテム基本
AShooterPickup _Ammo
補充弾数
武器種別
AShooterPickup _Health
体力回復量
弾アイテム
体力アイテム
Pickup _AmmoGun
Pickup _AmmoLauncher
Pickup _Health
速射銃弾
ランチャー弾
体力回復

Item

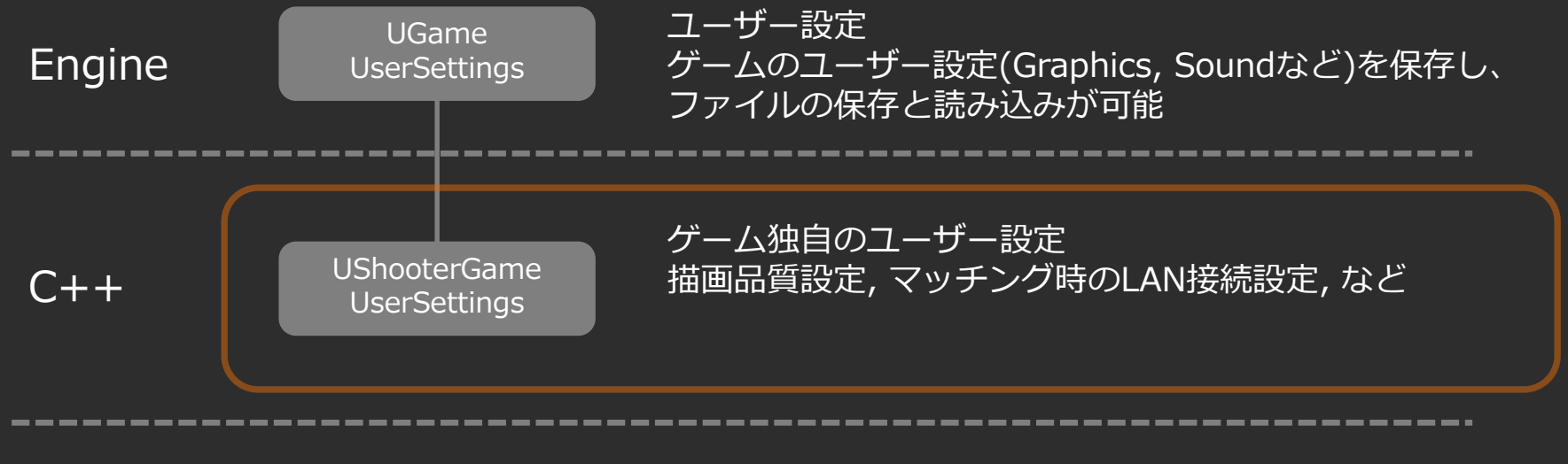


Pickup_AmmoGun

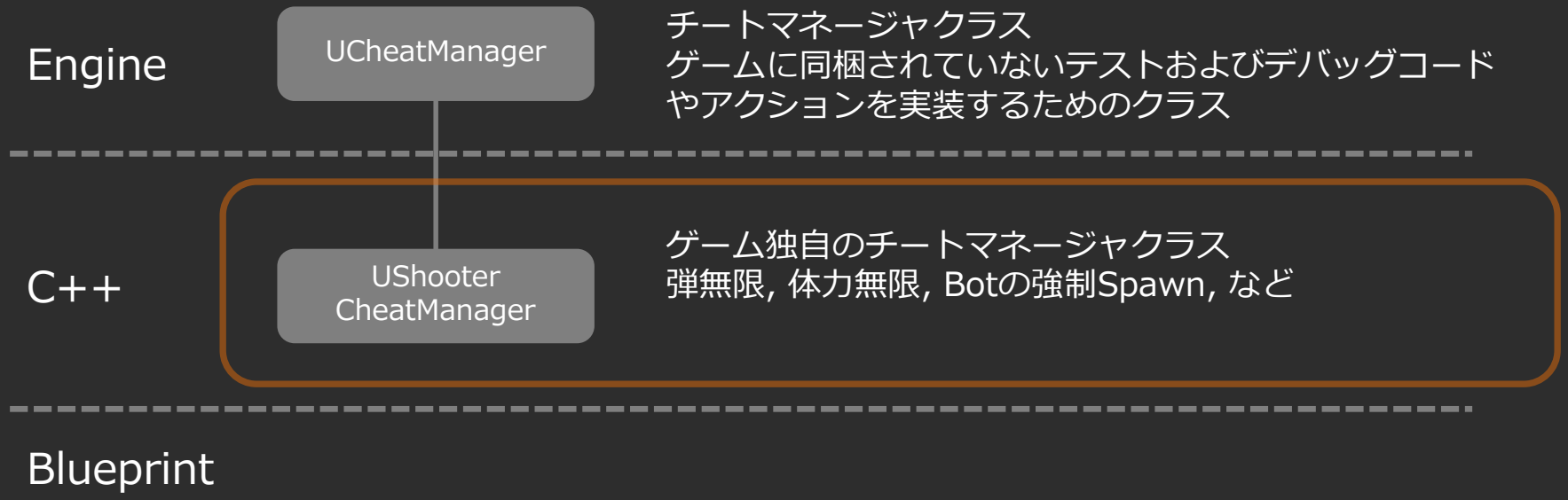


Pickup_Health

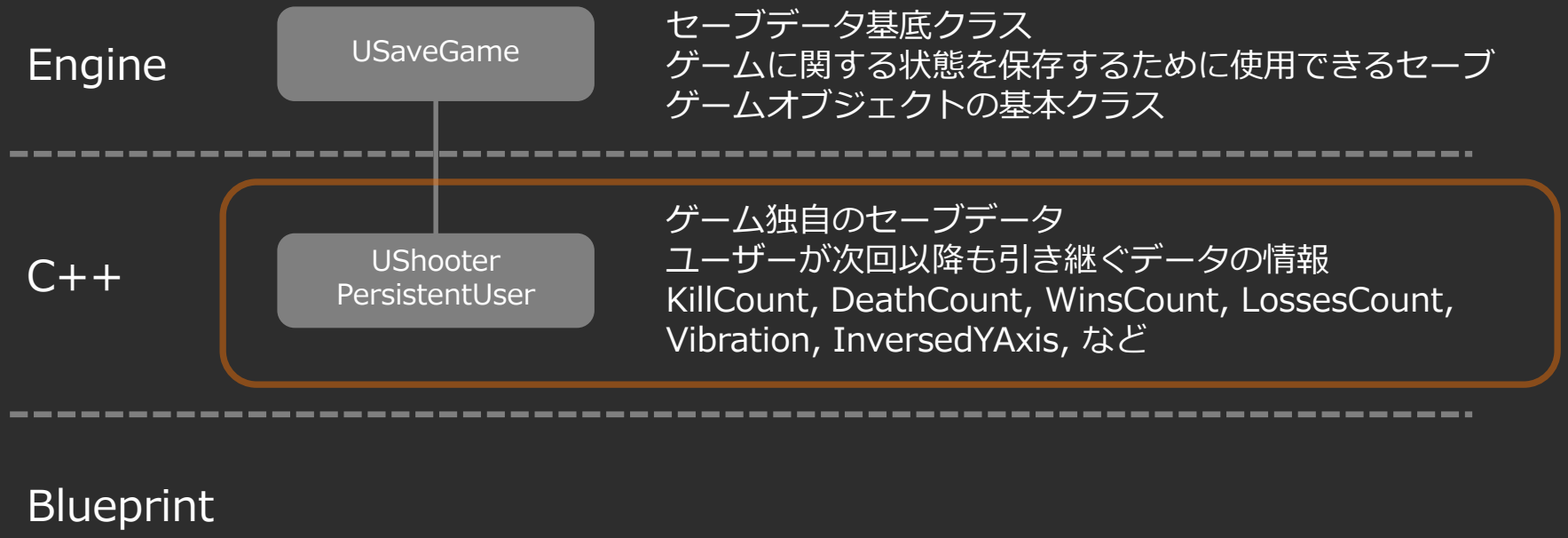
GameUserSettings



CheatManager



SaveGame

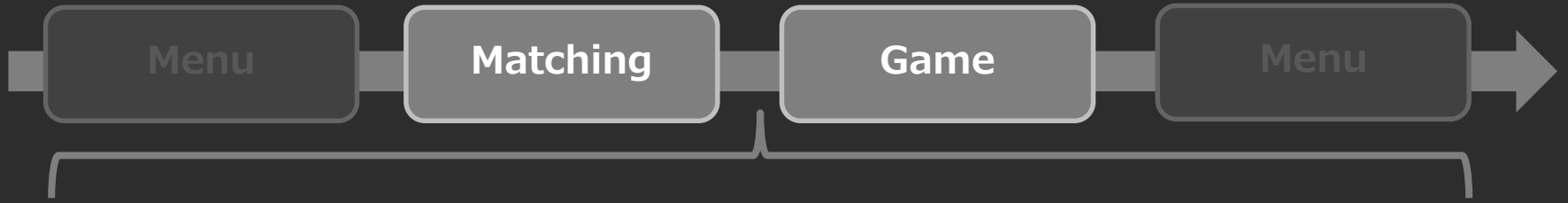

Engine
USaveGame
セーブデータ基底クラス
ゲームに関する状態を保存するために使用できるセーブ
ゲームオブジェクトの基本クラス
C++
UShooter
PersistentUser
ゲーム独自のセーブデータ
ユーザーが次回以降も引き継ぐデータの情報
KillCount, DeathCount, WinsCount, LossesCount,
Vibration, InversedYAxis, など
Blueprint

Online MultiPlay Game の ネットワーク機能

ゲームフロー

Menu → Matching → Game → Menu

ゲームフロー



Matchmake

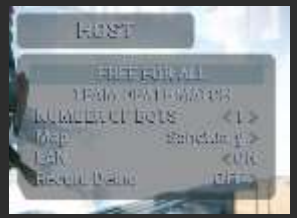
Character Move
Shoot

Matchmake

- Server / Clientの接続はSessionで行う

Menu Map

Server



Client

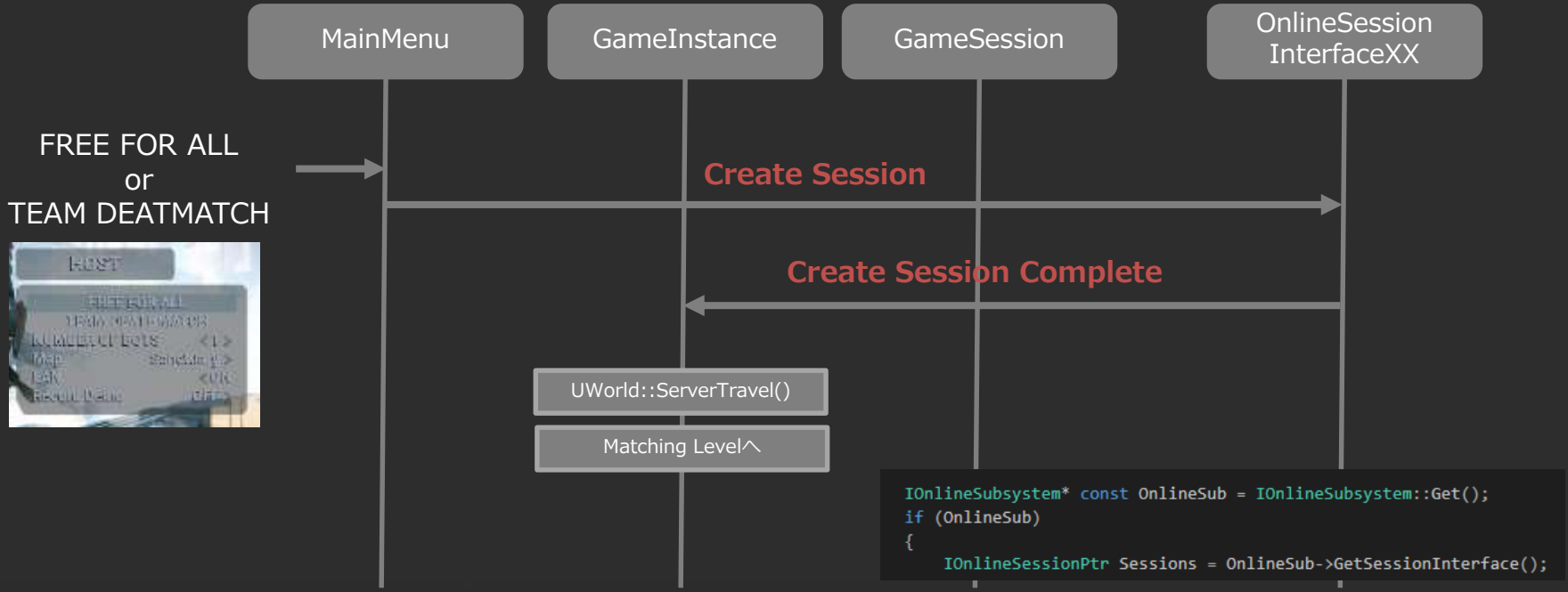
Create Session

Matching Map

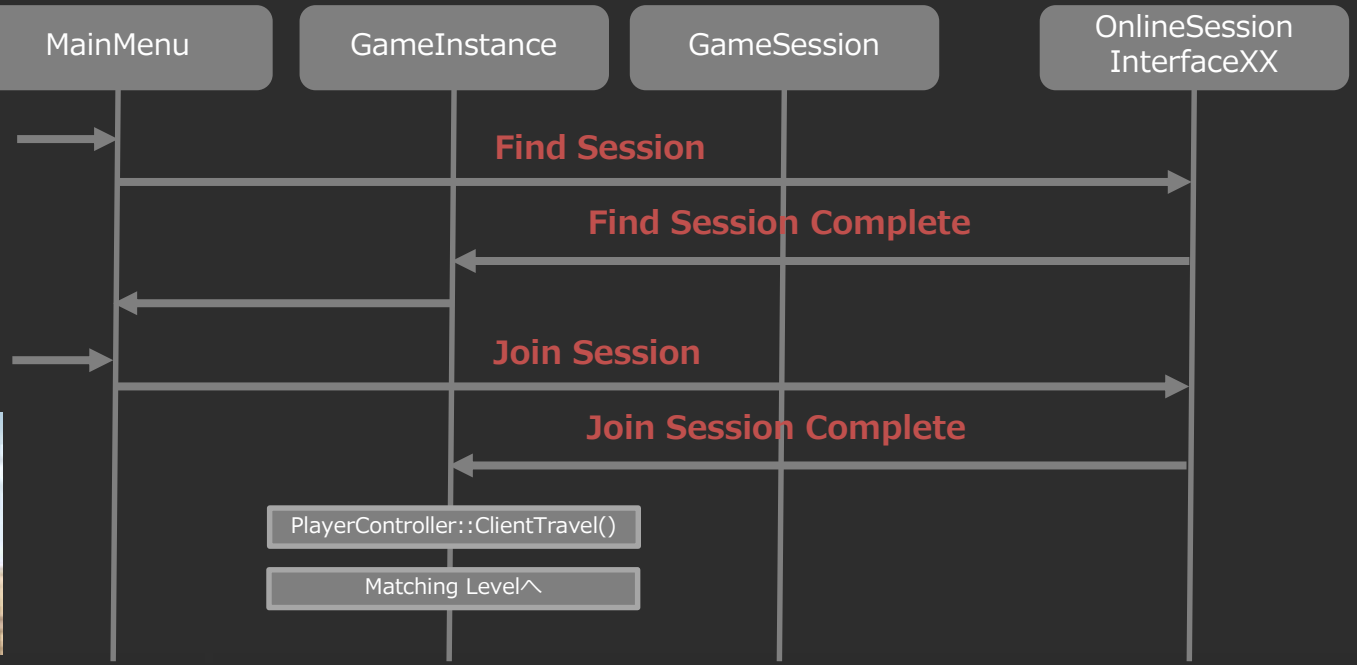
Find Session

Join Session

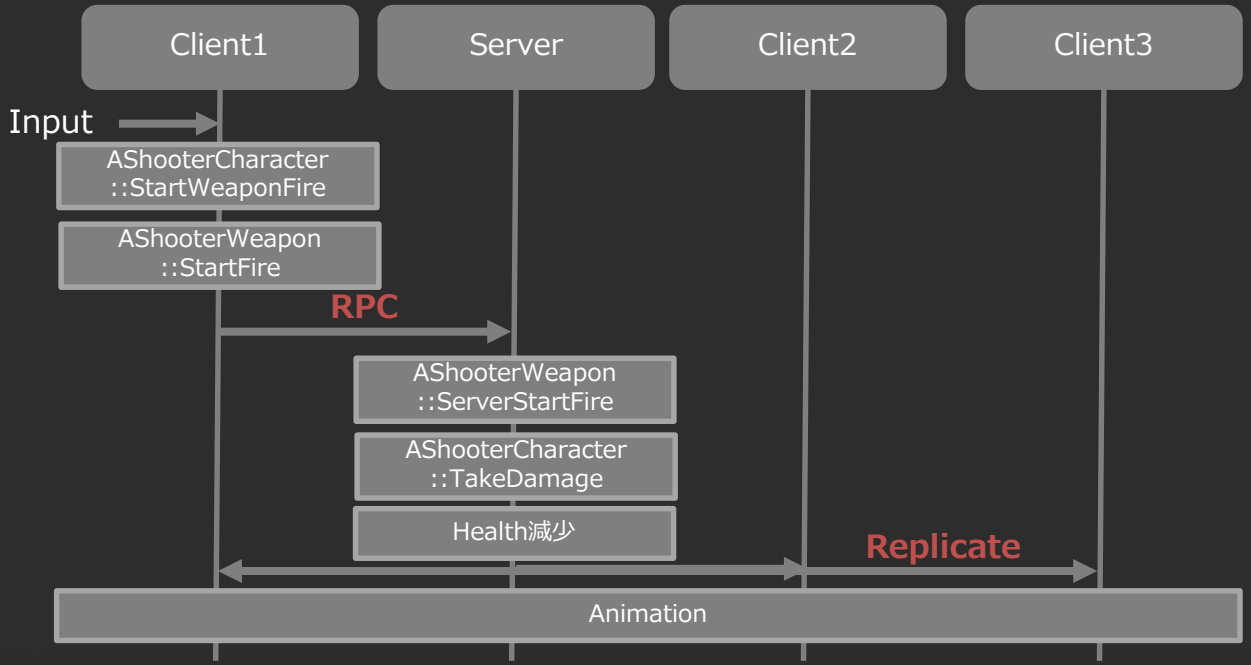
Matchmake - Server



Matchmake - Client



Game - Shoot


Client1
Server
Client2
Client3
Input
AShooterCharacter
::StartWeaponFire
AShooterWeapon
::StartFire
RPC
AShooterWeapon
::ServerStartFire
AShooterCharacter
::TakeDamage
Health減少
Replicate
Animation

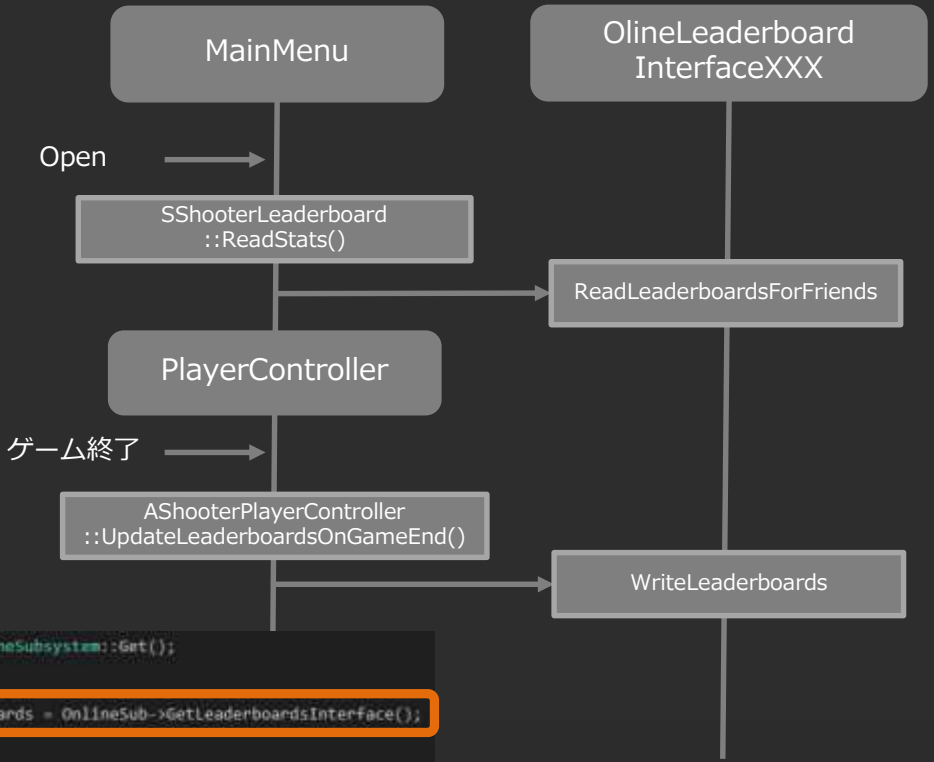
ネットワークTips

ネットワークTips

- Leaderboard
- Achievements
- ReplaySystem
- Profiler
- Logging
- Console Command
- 制約

Leaderboard

- スコア, ランキング機能
  - ShooterGameではSShooterLeaderboardでUI表示が実装

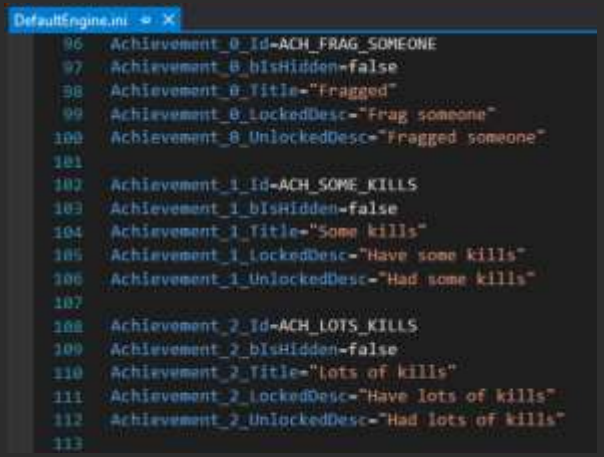


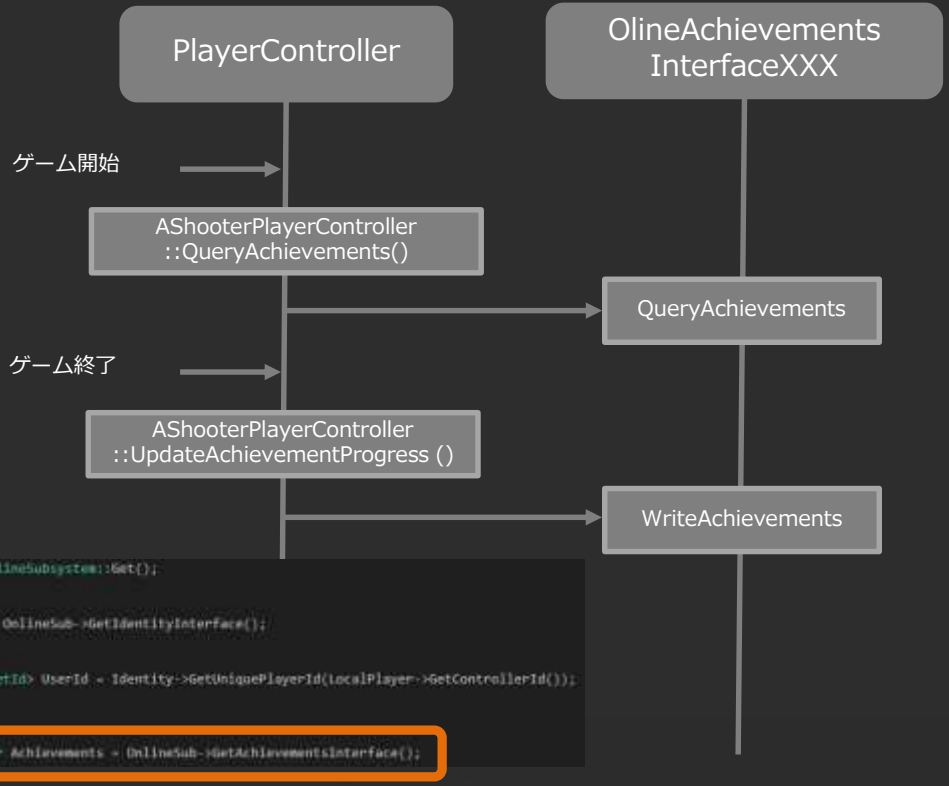

```
/** leaderboard row display information */
struct FLeaderboardRow
{
    /** player rank*/
    FString Rank;

    /** player name */
    FString PlayerName;

    /** player total kills to display */
    FString Kills;

    /** player total deaths to display */
    FString Deaths;

    /** Unique Id for the player at this rank */
    const TSharedPtr<const FUniqueNetId> PlayerId;

    /** Default Constructor */
    FLeaderboardRow(const FOnlineStatsRow& Row);
};
```

```
IOnlineSubsystem* OnlineSub = IOnlineSubsystem::Get();
if (OnlineSub)
{
    IOnlineLeaderboardsPtr Leaderboards = OnlineSub->GetLeaderboardsInterface();
```

Achievements

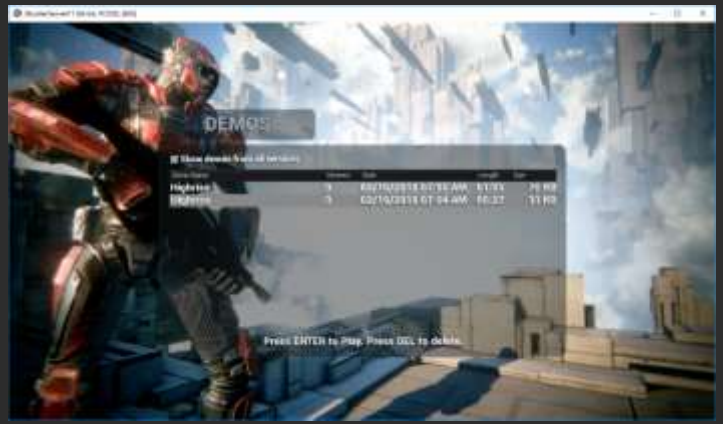
- 実績機能
  - ShooterGameでは主にPlayerControllerからアクセス
  - DefaultEngine.iniで各種設定





ReplaySystem

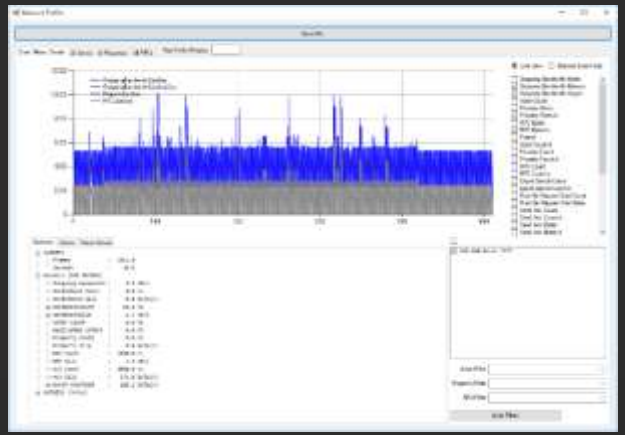
- リプレイ機能
  - Replicationの情報を元に状況を再現
  - DemoNetDriverの機能を利用 (.iniファイルで設定が設定要)
  - リプレイ情報保存先：[Project]/Saved/Demos



公式ドキュメント
https://docs.unrealengine.com/latest/INT/Engine/Replay/

Network Profiler

- NetworkTrafficを解析するツール
  - コンソールコマンドでパケット情報を収集 (“netprofile”など)
  - アプリケーション保存先：Engine¥Binaries¥DotNET
  - 収集ファイル保存先：[Project]/Saved/Profiling



公式ドキュメント
https://docs.unrealengine.com/latest/INT/Gameplay/Tools/NetworkProfiler/

Logging

- ログ出力レベルを変更することでネットワーク関連の情報が取得

- ログ出力レベルについて
[UE4] UE_LOGについてあれこれ
http://historia.co.jp/archives/5532/

- ログ出力マクロについて
Log Macro with Netmode and Colour
https://wiki.unrealengine.com/Log_Macro_with_Netmode_and_Colour

Network関連Log (1/2)

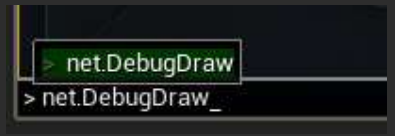
#	Log	Description	Verbosity (橙色:使用, ○:初期設定)								Note
			No Logging	Fatal	Error	Warning	Display	Log	Verbose	Very Vorbose	
1	LogCharacterNetSmoothing	キャラクタ移動補間関連ログ出力						○			
2	LogCookedIterativeNetworkFile	CookOnTheFly時ファイル関連ログ出力					○				
3	LogGameNetworkManager	ネットワークマネージャ関連ログ出力						○			
4	LogHandshake	3-Wayハンドシェィク関連ログ出力						○			
5	LogNet	ネットワーク関連イベント出力						○			
6	LogNetDormancy	ネットワーク休止イベント出力				○					
7	LogNetFastTArray	ネットワーク動的配列関連ログ出力				○					
8	LogNetPackageMap	パッケージマップ関連ログ出力				○					
9	LogNetPartialBunch	データチャネル関連ログ出力				○					
10	LogNetPlayerMovement	キャラクタ移動関連ログ出力				○					
11	LogNetSerialization	シリアライズ関連ログ出力				○					
12	LogNetTraffic	ネットワーク情報パケットログ出力				○					
13	LogNetVersion	ネットワークバージョン出力						○			

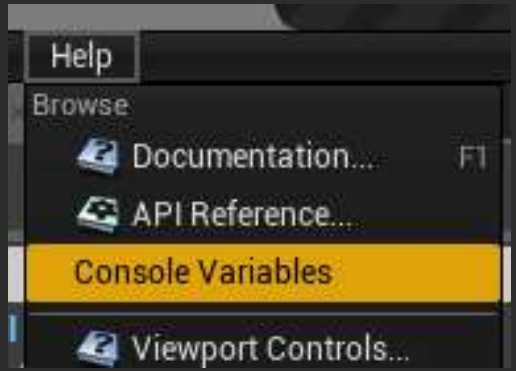
Network関連Log (2/2)

#	Log	Description	Verbosity (橙色:使用, ○:初期設定)								Note
			No Logging	Fatal	Error	Warning	Display	Log	Verbose	Very Vorbose	
14	LogNetworkAutomationTests	ネットワーク自動テスト時ログ出力						○			UnUsed
15	LogNetworkPlatformFile	ネットワーク経由ファイル操作関連ログ出力						○			
16	LogOnline	OSS関連イベントやパラメータの出力							○		
17	LogOnlineChat	Chatroom関連イベントの出力					○				
18	LogOnlineGame	LobbyBeacon使用時のユーザー情報出力					○				
19	LogOnlineParty	パーティ関連のログ出力					○				UnUsed
20	LogRep	レプリケーションイベント/パラメータ出力						○			
21	LogRepTraffic	レプリケーション情報出力				○					
22	LogSkippedRepNotifies	レプリケーション通知スキップ情報出力	○								
23	LogTcpMessaging	TCPメッセージ関連出力						○			
24	LogUdpMessaging	UDPメッセージ関連出力						○			
25	PacketHandlerLog	PacketHandler、暗号化、復号関連出力						○			

Console Command

- 設定を動的に変更したり指定の情報を出力することが可能
- ネットワーク関連のコマンドは”net.***”
- Help->Console Variablesから一覧表示





Network関連Console Command (1/3)

#	Command	Description	Type	Default	Note
1	net.AllowAsyncLoading	非同期ロード設定 PackageMapを元にNetGUIDからオブジェクトを非同期でロードする	Int32	0	0：非許可 1：許可
2	net.AllowEncryption	PacketHandlerコンポーネントをロードして?EncryptionToken = URLに基づいて NMT_HelloメッセージのEncryptionTokenパラメータを入力	Int32	1	0：無効 1：有効
3	net.ContextDebug	Replication情報にデバッグ用文字列を設定してログに詳細な情報を表示	Int32	0	0：無効 1：有効
4	net.DormancyEnable	頻繁に更新されるアクターのCPUおよび帯域幅のオーバーヘッドを削減するネットワーク休止システムの有効化	Int32	1	0：無効 1：有効
5	net.DormancyDraw	ネットワーク休止システムのデバッグ情報の描画	Int32	0	0：無効 1：有効
6	net.DormancyDrawCullDistance	net.DormancyDrawのCulling距離, World単位 localのViewから離れているActorの休止状態を描画する	Float	5000	
7	net.DormancyValidate	休止状態のときに休止状態のActorが状態を変更しないことを検証	Int32	0	0：検証しない 1：起動時に検証 2：更新時に検証
8	net.DelayUnmappedRPCs	マッピングされていないプロパティを持つ受信RPCを遅延させる設定	Int32	0	0：無効 1：有効
9	net.DeleteDormantActor	休止中のActorを削除	ー	ー	
10	net.DoPropertyChecksum	ReplicationデータのSumCheck実行可否	Int32	0	0：無効 1：有効
11	net.DumpRelevantActors	次回ネットワーク更新時に関連するActorの情報を出力	ー	ー	
12	net.IgnoreNetworkChecksumMismatch	Network上でのSumCheckエラーを無視する設定	Int32	0	0：エラー無視 1：エラー検知
13	net.InstantReplayProcessQueuedBunchesMillisecondLimit	インスタントリプレイ中にQueに入れられたBunchを処理するための時間の閾値 閾値を超過する場合次のフレームまで待ってQueに入れられたBunchの処理を続行	Int32	8	0：無制限

Network関連Console Command (2/3)

#	Command	Description	Type	Default	Note
14	net.ListActorChannels	ActorChannelのリスト一覧表示	–	–	
15	net.ListNetGUIDs	NetGUIDのリスト一覧表示	–	–	
16	net.ListNetGUIDExports	NetGUIDとエクスポート回数のリスト一覧表示	–	–	
17	net.MaxSharedShadowStateUpdateDelayInSeconds	特定の接続で新しいCLが使用可能になっても指定時間が経過した場合は、強制的にすべてのプロパティを結合	Float	0.25	
18	net.MaxPlayersOverride	最大プレイヤー数の上書き指定	Int32	0	0：無効 1～：プレイヤー人数
19	net.MaxRPCPerNetUpdate	1回の更新で許可されるRPCの最大数	Int32	2	
20	net.MaxRepArraySize	Replication情報の動的配列最大要素数指定 (element)	Int32	2048	
21	net.MaxRepArrayMemory	Replication情報の動的配列最大サイズ指定 (Byte)	Int32	65535	
22	net.Montage.Debug	AnimMontageに関するReplication情報のログ出力	Int32	0	0：出力なし 1：出力あり
23	net.OptimizedRemapping	最適化されたパスを使用してマッピングされていないネットワークGUIDのRemap	Int32	1	
24	net.PackageMap.LongLoadThreshhold	Objectのシリアライズのロードが長い時に警告を出力する時間	Float	0.02	
25	net.PackageMap.DebugAll	すべてのオブジェクトのPackageMapシリアライズデバッグ	Int32	0	0：無効 1：有効
26	net.PackageMap.DebugObject	Object指定でのPackageMapのシリアライズデバッグ 指定したObjectシリアライズ、またはReplication時にログ出力	FString	""	
27	net.Packagemap.FindNetGUID	指定されたNetGUIDに割り当てられたオブジェクトを検索して表示 指定あり/なしで表示が変わる	TArray<FString>	–	指定なし：全て表示 指定あり：対象のみ表示

Network関連Console Command (3/3)

#	Command	Description	Type	Default	Note
28	net.PartialBunchReliableThreshold	部分的Bunchの高信頼性閾値	Int32	0	
29	net.PingExcludeFrameTime	Server-Client間のNICのRTT時間を計算 ACKデータに書き込まれる	Int32	0	0：無効 1：有効
30	net.PingDisplayServerTime	サーバーのフレーム時間を表示 LogNetTraffic/Warning時に有効	Int32	0	0：出力なし 1：出力あり
31	net.ProcessQueuedBunchesMillisecondLimit	Queに入れられたBunchを処理するための時間の閾値 1つのフレームでこれより時間がかかる場合は、次のフレームまで待ってQueに入れられたBunchの処理を続行	Int32	30	0：無制限
32	net.RandomizeSequence	初期パケットシーケンスのランダム化	Int32	1	0：無効 1：有効
33	net.Reliable.Debug	全てのReliable bunchログ出力	Int32	0	0：出力なし 1：送信時のみ出力 2：更新時に出力
34	net.Replication.DebugProperty	名前によるプロパティのReplicateをデバッグ	FString	""	
35	net.RelinkMappedReferences	再リンクマッピング参照 このオブジェクトを参照しているレプリケータをすべて検索し、参照をマップされていないものとしてマークする	Int32	1	
36	net.RPC.Debug	全てのRPC bunchログ出力	Int32	0	0：出力なし 1：出力あり
37	net.ShareShadowState	プロパティを比較する作業は接続間で共有される	Int32	1	
38	net.TestObjRefSerialize	全クライアントにオブジェクト参照をReplicateするテスト	TArray<FString>	ー	
39	net.TickAllOpenChannels	ネットワーク接続においてTick毎にすべてのOpen Channelをチェック (有効時はパフォーマンスが低下)	Int32	0	0：無効 1：有効
40	net.UseAdaptiveNetUpdateFrequency	Replicate周期Actor依存設定 詳細はServerReplicateActors_BuildConsiderList	Int32	1	0：1.0f固定 1：Actor依存

制約

- ホストマイグレーション
  - Listen Serverで接続時のHost移譲など
- 複数セッションへの接続
  - セッション接続中に別のセッションへの参加など
- 異なるプラットフォーム間のOSS連動
  - WindowsとiOSのネットワークコミュニケーションなど

- その他…

まとめ

- Engineの機能を把握した上で適切な設計を
- BlueprintとC++で出来ることを把握して設計を
- 特にオンラインマルチプレイ時には注意

参考資料

UE4 MultiPlayer Online Deep Dive Slide

- UE4 MultiPlayer Online Deep Dive 基礎編1 -Getting Started-
https://www.slideshare.net/EpicGamesJapan/ue4-multiplayer-online-deep-dive-getting-started-ue4dd
- UE4 MultiPlayer Online Deep Dive 基礎編2 -Traveling-
https://www.slideshare.net/EpicGamesJapan/ue4-multiplayer-online-deep-dive-traveling-ue4dd
- UE4 MultiPlayer Online Deep Dive 実践編1
https://www.slideshare.net/EpicGamesJapan/ue4-multiplayer-online-deep-dive-1-byking-ue4dd?next_slideshow=1
- UE4 MultiPlayer Online Deep Dive 実践編2
https://www.slideshare.net/EpicGamesJapan/ue4-multiplayer-online-deep-dive-2-ue4dd

参考資料

- UE4 Networking

https://www.slideshare.net/JoeGraf1/ue4-networking?qid=bba12faf-ef8f-456d-9ecf-ee3656e4c3d9&v=&b=&from_search=1

- Multiplayer Network Compendium

http://cedric-neukirchen.net/Downloads/Compendium/UE4_Network_Compendium_by_Cedric_eXi_Neukirchen.pdf