In Unreal Engine, Replication is the name for the process of synchronizing data and procedure calls between clients and servers. The Replication system provides a higher-level abstraction along with low-level customization to make it easier to deal with all the various situations you might encounter when creating a project designed for multiple simultaneous users.

<https://cedric-neukirchen.net/Downloads/Compendium/UE4_Network_Compendium_by_Cedric_eXi_Neukirchen.pdf>

#include “Net/UnrealNetwork.h”

bReplicated = true 여도 클라에서 만들면 그 클라에만 존재함.(Autonomous)

#

UPROPERTY(Replicated)

RepVar

RepActor::GetLifetimeReplicatedProps

DOREPLIFETIME(RepActor, RepVar); //Opt: COND\_OwnerOnly

#

#

UPROPERTY(ReplicatedUsing=RepFunc)

RepVar

UFunction()

RepFunc()

#

RPC

BP: CustomEvent Replicates

#

C++: UFunction(Server,unreliable,WithValidation); //Client/NetMulticast, reliable/unreliable

(un)reliable없으면 컴파일이 안됨.

Server\_RPC()

Server\_RPC\_Implementation()

Server\_RPC\_Validate() {return true;} //when return false, disconnects caller

#

클라<->서버 서버->(특정)클라들

클라에서 rpc를 호출하면 서버에서 validate 등을 거쳐서 replicate

If the RPC is being called from Server to be executed on a Client, only the Client who actually owns that Actor will execute

the function.

If the RPC is being called from Client to be executed on the Server, the Client must own the Actor that the RPC is being called on.

Multicast RPCs are an exception:

If they are called from the Server, the Server will execute them locally, as well as execute them on all currently connected Clients.

If they are called from Clients, they will only execute locally, and will not execute on the Server.

서버/클라별 주요 클래스

{

Server Only

GameMode

Server & Clinents

GameState PlayerState

Server & Owning Client

PlayerController

Owning Client Only

HUD

}

Ownership

월드에 생성된 액터들은 레벨에 배치된 것을 포함한 대부분의 경우 서버가 소유하기 때문에

클라이언트가 그런 액터에 대해 RPC를 호출해도 소유권이 없어서 작동하지 않음.

그래서 PlayerController(first class) 같은 소유권이 있는 액터에서 서버로 RPC를 호출하고 그 서버 함수에서 해당 액터에 대해 처리 및 Replicate.

Connection은 PC를 갖고 있는데 액터들의 소유권을 볼 땐 most outer가 PC면 그것도 소유. 예)Pawn - PC가 Character를 Possess할 때 Own하기 때문.

Role

Server

Role == Role\_Authority

ReomoteRole == Role\_SimulatedProxy

Client is reversal.

SimulatedProxy

standard, 서버의 업데이트와 클라의 업데이트의 간격에 차이가 있기 때문에 이전 서버의 업데이트에 보간하여 클라 업데이트.

AutonomousProxy

일반적으로 PC가 소유하는 것에서 플레이어 인풋에 따라 보간.

AGameMode::bUseSeamlessTravel

seamless: non-blocking

non-seamless: blocking (disconnect and reconnect)

There are three ways in which a 'Non-seamless' travel must occur:

• When loading a Map for the first time

• When connecting to a Server for the first time as a Client

• When you want to end a Multiplayer Game, and start a new one

persist

• The GameMode Actor (Server only)

◦ Any Actors further added via 'AGameMode::GetSeamlessTravelActorList'

• All Controllers that have a valid PlayerState (Server only)

• All PlayerControllers (Server only)

• All local PlayerControllers (Server and Client)

◦ Any Actors further added via 'APlayerController::GetSeamlessTravelActorList' called on local PlayerControllers

mark persist to transition map and travel -> repeat to final map

Online Subsystem

provide a clear abstraction to common online functionality across the available

set of platforms in a given environment. (Steam)

default SubsystemNULL host LAN sessions (not internet)

subsytem Master Server that provides the Client with a list of Servers/Sessions

주요 프로퍼티 정리

AActor::bReplicates - 이 액터에 대해 (어떤 방식으로든)replicate를 할지. 따라서 이 액터를 레플리케이트 할 것이라면 일단 이 변수를 true로 설정해야 함(SetReplicates). 기본값은 false.

AActor::bReplicateMovement - 이 액터의 위치 및 이동을 레플리케이트 할지.

AActor::bNetLoadOnClient - 클라이언트에서 맵을 로드 할 때 맵에 배치된(placed) 이 액터를 로드할지. 이렇게 로드되면 bNetStartup도 true이며 #? IsNetStartup체크 이후 패키지된 어셋 경로를 네트워크에서 레퍼런스로 쓰는 것이 보장 됨. 기본값은 true.

실제로는 맵에 배치된 액터는 이 값에 관계없이 일단 다 생성이 되는데

이 값이 true이면, 위에서 말한대로 맵을 로드할 때 정상적으로 생성되고, 계속 클라이언트에 남아 레플리케이트 정보를 받음.

반면에 false면, 클라이언트에서 ULevel::InitializeNetworkActors할 때 Destory함. 이 후에 해당 액터는 bReplicates가 true이며 Relevant할 때 레플리케이트되어 생성되고 그렇지 않게 될 때마다 삭제됨. 따라서 이를 반복할 때마다 오버헤드임.

또한 일반적으로 true면 클라이언트에 반응이 빠르지만 싱크가 어긋나기 쉬워지고,

false면 반대로 싱크가 크게 어긋날 일은 없지만 반응은 더디다는 면면이 있음.

#? simualte physics하는 컴포넌트가 있을 때 그 컴포넌트 자체를 레플리케이트 하는 것은 너무 데이터가 많을 것이기 때문에 물리 적용 결과만 보내야될 것이다.

Pawn::IsLocallyControlled

AI는 서버에서 true 그 외 false.

플레이어는 로컬에서 true 그 외 false.