# Prácticas de Programación en Red

Sergio Jiménez Galiana

Ejercicio 1:

La alternativa número 1 consiste en realizar una bomba de área que ralentiza al resto de coches. Al pulsar la tecla espacio (o un botón del mando), todos los coches que estén en un radio de 200 unidades ven reducida su velocidad máxima a la mitad. El tiempo de vida de la bomba es de 5 segundos, por lo que cualquier coche que entre en el área durante ese tiempo se verá afectado. No es necesario usar triggers o queries de Unreal, se puede recorrer la lista de coches y calcular la distancia con ellos.

## GameNetMgr

Se ha creado un enum en GameNetMgr.h para distinguir los tipos de mensajes que puede recibir o emitir una entidad durante el juego:

* Transform: mensajes que envía el server a los clientes en el método Tick para comunicar la posición, rotación y escala de los coches. Junto con este mensaje se envía la velocidad instantánea de ese tick para que los clientes puedan mostrarla por la UI.
* Input: mensajes que envían los clientes al servidor para comunicar el input del jugador al server, dónde se ejecutará la lógica del movimiento en función de estos datos.
* SpawnBomb: mensaje para notificar al server que un cliente quiere spawnear una bomba.
* DestroyBomb: mensaje para notificar a los clientes que deben destruir la bomba spawneada.

## NetComponent

Se ha dejado la funcionalidad "base" de la comunicación por red en NetComponent, y la funcionalidad más específica de serializar y deserializar datos del input o del transform se ha implementado en una clase hija llamada InputNetComponent. De esta manera, se pueden crear otras entidades que contengan un NetComponent sin una funcionalidad específica.

En NetComponent se ha creado un método CreateEntityMessage(EEntityMessageType type), para que cada vez que se quiera crear un mensaje ENTITY\_MSG se serializen los datos en el mismo orden que la deserialización.

## BombSpawnerComponent

BombSpawnerComponent es un ActorComponent que se ha creado con la finalidad de gestionar la creación y destrucción de las bombas. Este componente guardará un puntero a la bomba creada para evitar que un coche pueda crear más de una bomba al mismo tiempo.

## Bomb

Bomb es un actor con un StaticMesh y un DecalComponent para poder visualizar en el mundo la posición y alcance de la bomba. Además contiene toda la lógica de ralentizar a los coches que entren en su área de efecto (los coches que no sean su creador). Esta lógica sólo se realizará en el servidor, por lo que esta entidad contendrá un NetComponent para poder hacer esta comprobación. Se comprobará la distancia con todos los coches en el Tick para ralentizar a los afectados. Estos se guardarán en un array para que, al destruir la bomba, se reestablezca la velocidad de los coches (a no ser que estén afectados por otra bomba).

## InputNetComponent

InputNetComponent se encargá de mandar al servidor los mensajes de Input y a los clientes los mensajes de Transform.

Cuando el jugador accione la tecla asignada para spawnear la bomba, se creará un mensaje de tipo SpawnBomb que se mandará al servidor de manera segura, para asegurarse que llegue. Al llegar al servidor, se spawneará la bomba a través del BombSpawnerComponent y se reenviará el mensaje a los clientes para que también la spawneen (aunque en los clientes solo va a ser algo visual para identificar dónde está la bomba).

Al spawnear la bomba en el servidor, se activará un timer de 5 segundos. Pasado este tiempo se destruirá la bomba y se notificará a los clientes para que hagan lo mismo.

## Notas

* Se podrían haber enviado los mensajes de SpawnBomb y DestroyBomb junto con los mensajes de Transform para aprovechar el buffer y optimizar los mensajes mandados, pero se ha preferido separarlo en distintos mensajes ya que los mensajes de tipo SpawnBomb y DestroyBomb son importantes que se dan en un momento específico, por lo que no se pueden perder. Para ello se marcan estos mensajes como "reliable", asegurandose de que lleguen correctamente a su destino.
* Toda la lógica de ralentizar a los coches se hace por parte del servidor, por lo que no es necesario replicar la velocidad máxima que llevan los coches. En este caso se ha replicado la velocidad instantánea para poder mostrarlo por la UI en los clientes y poder ver cuándo se reduce la velocidad o se recupera la velocidad máxima.