ENTREGABLES OBDH

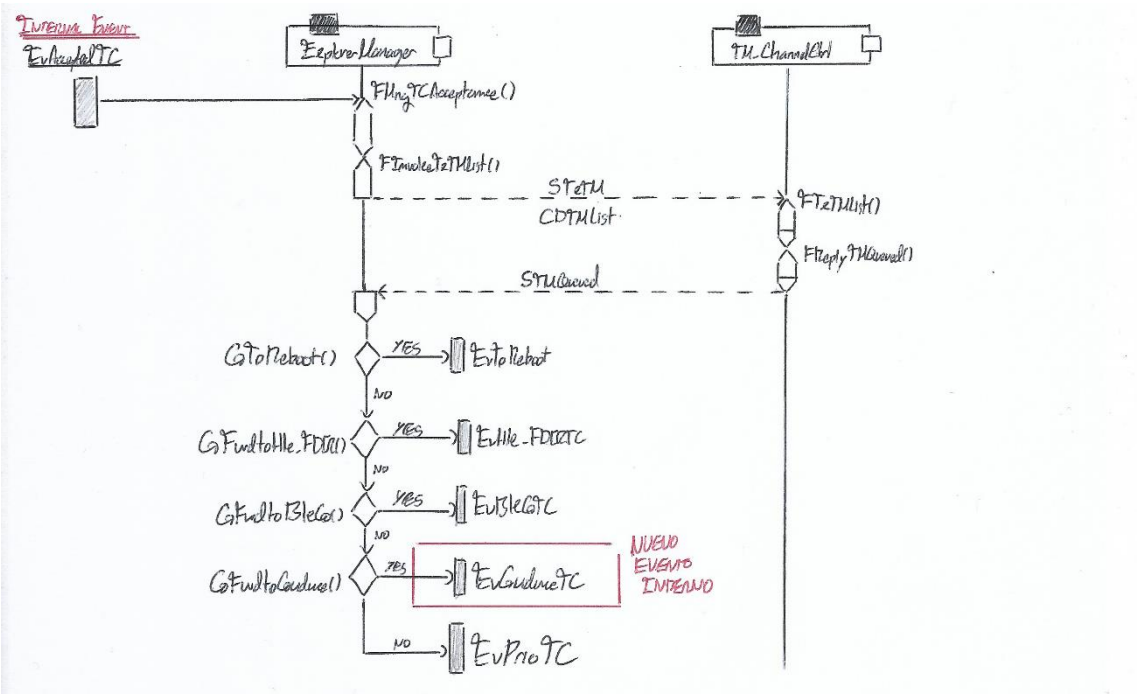
Trabajo realizado por: David Bote Albacete

ENTREGABLE 1: ESCENARIOS PARA LA IMPLEMENTACIÓN DEL SERVICIO 129

Para conseguir la implementación del servicio 129, debemos implementa una nueva tarea en nuestro modelo EDROOM que denominaremos “Guidance” y cuya finalidad será la de completar la ejecución de la función de control de velocidad periódicamente, cada 100 milisegundos y la de procesar los telecomandos TC[129,1] y TC[129,2] que le lleguen al Manager.

Con eso en mente, los diagramas de secuencia que debemos modificar o crear son los siguientes:

ESCENARIO EvAcceptedTC:

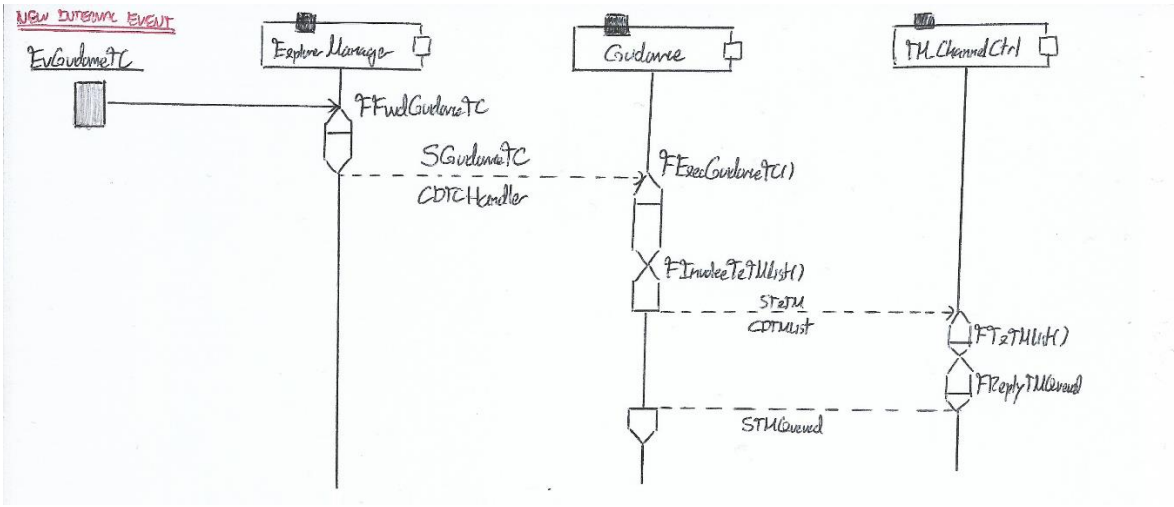


En la que podemos observar que se añade una nueva guarda “GFwdtoGuidance()” asociada a un nuevo evento “EvGuidanceTC”.

Un telecomando del servicio 129 llegaría al Explorer Manager, el cual lo aceptaría o rechazaría como ocurre con cualquier otro servicio implementado hasta ahora.

Suponiendo que el telecomando es válido y es aceptado; debemos prepararlo para enviarlo al nuevo componente “Guidance”. Esto es lo que conseguimos con el nuevo evento “EvGuidanceTC”, que presenta el siguiente diagrama:

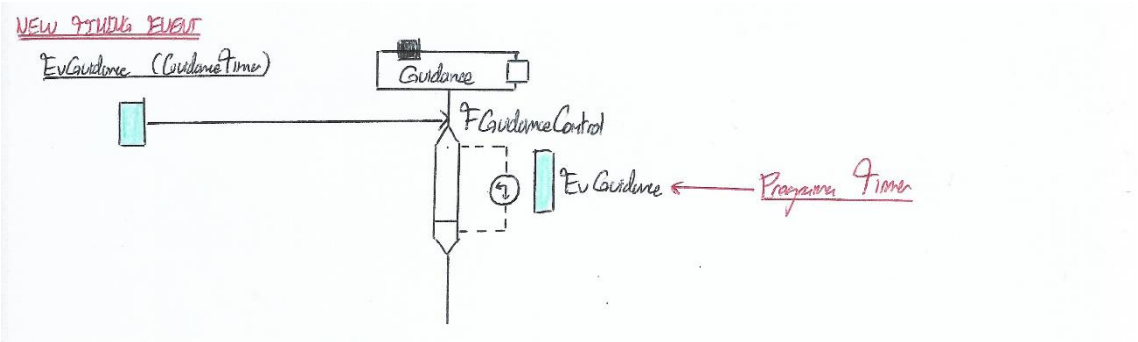
ESCENARIO EvGuidanceTC:



En este diagrama observamos que el protocolo de comunicación entre el componente "Guidance" y el componente "TM_ChannelCtrl" es el mismo que empleamos para comunicar "TM_ChannelCtrl" con otros componentes; de forma que podemos reutilizarlo. Por otro lado, observamos que el protocolo de comunicación entre el Explorer Manager y el componente Guidance se compone de una señal "SguidanceTC" con dato adjunto del tipo "CDTCHandler". Con este protocolo logramos remitir el telecomando TC[129,1] o TC[129,2] del componente Explorer Manager al componente Guidance para su ejecución.

Por último, necesitamos que el nuevo componente Guidance ejecute periódicamente una función de control que no genera telemetrías. Para ello modelamos el comportamiento en base al siguiente diagrama secuencial:

ESCENARIO EvGuidance:



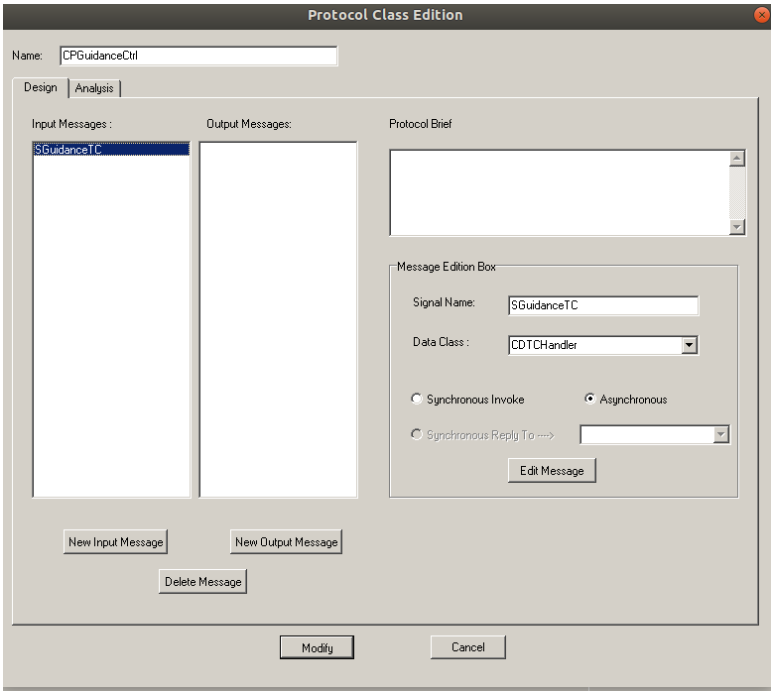
En el que definimos un evento de timing que programa la ejecución de la función de control con el periodo deseado (100 ms).

ENTREGABLE 2: DEFINICIÓN DE LA NUEVA CLASE PROTOCOLO

Como ya se ha comentado en la anterior cuestión, es necesario definir un nuevo protocolo para establecer la comunicación entre el componente “Explorer Manager” y “Guidance”.

Instanciamos la clase protocolo “CPGuidanceCtrl” y creamos un mensaje de entrada con la señal “SGuidanceTC” y un dato asociado con la clase CDTCHandler para remitir el telecomando que llega del componente “Explorer Manager” al componente “Guidance”.

En EDROOM deberíamos observar que el protocolo queda definido así:

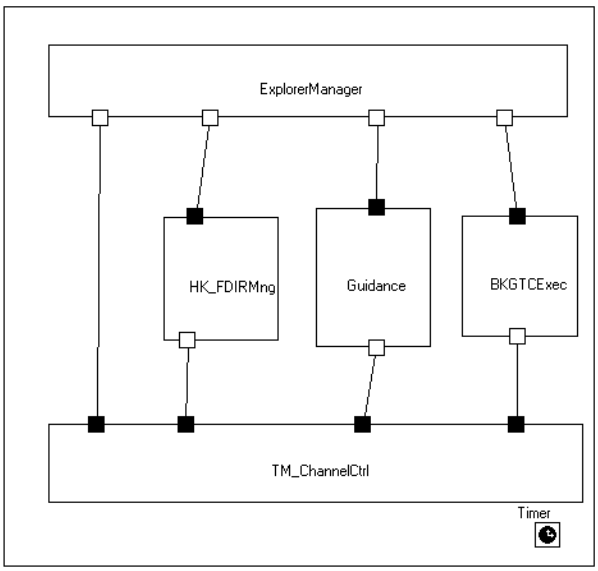


ENTREGABLE 3: DEFINICIÓN DE LA NUEVA CLASE PROTOCOLO

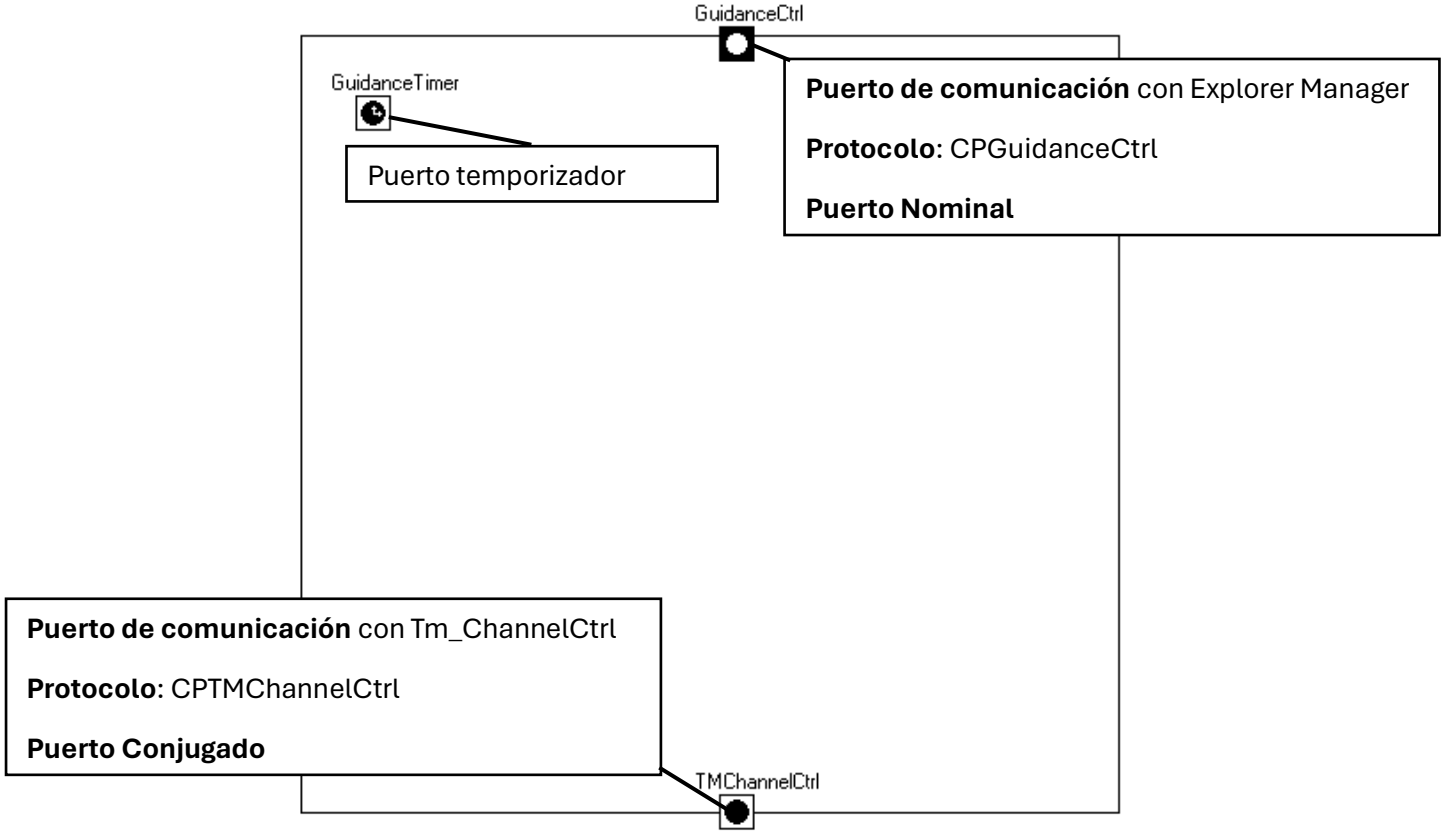
Emplearemos el nuevo protocolo definido “CPGuidanceCtrl” para comunicar el componente “Explorer Manager” con “Guidance” y el protocolo ya establecido “CPTMChannelCtrl” para comunicar el componente “Guidance” con el “TM_ChannelCtrl” (conexión que es necesaria porque como vimos en el diagrama se secuencias, aunque la ejecución de la función periódica no genere telemetrías, el componente generará telemetrías TM[1,7] y TM[1,8] para indicar si la ejecución de los telecomandos TC[129,1] y TC[129,2] ha sido llevada a cabo con éxito o no).

Finalmente, para la ejecución del evento periódico, es necesario añadir un puerto timer en el interior del componente.

Con esto en mente nos quedan unas conexiones tal que:



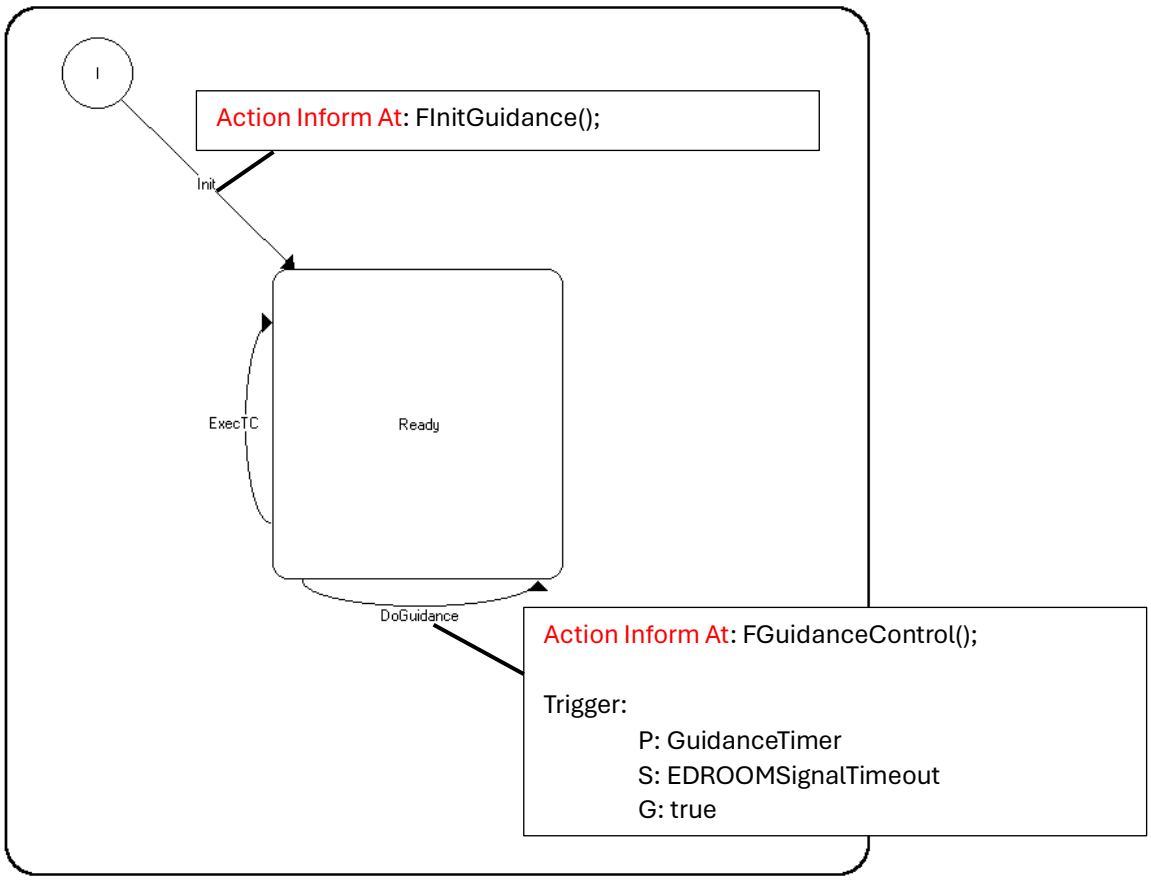
Y el componente “Guidance” queda diseñado:



ENTREGABLE 4: DEFINICIÓN DE COMPORTAMIENTO DE LA NUEVA CLASE CCGUIDANCE

Ya hemos definido las conexiones que son necesarias para que el componente pueda llevar a cabo su cometido. Llegamos a la definición de su comportamiento, que como pudimos observar en los diagramas secuenciales, podemos dividir en dos eventos.

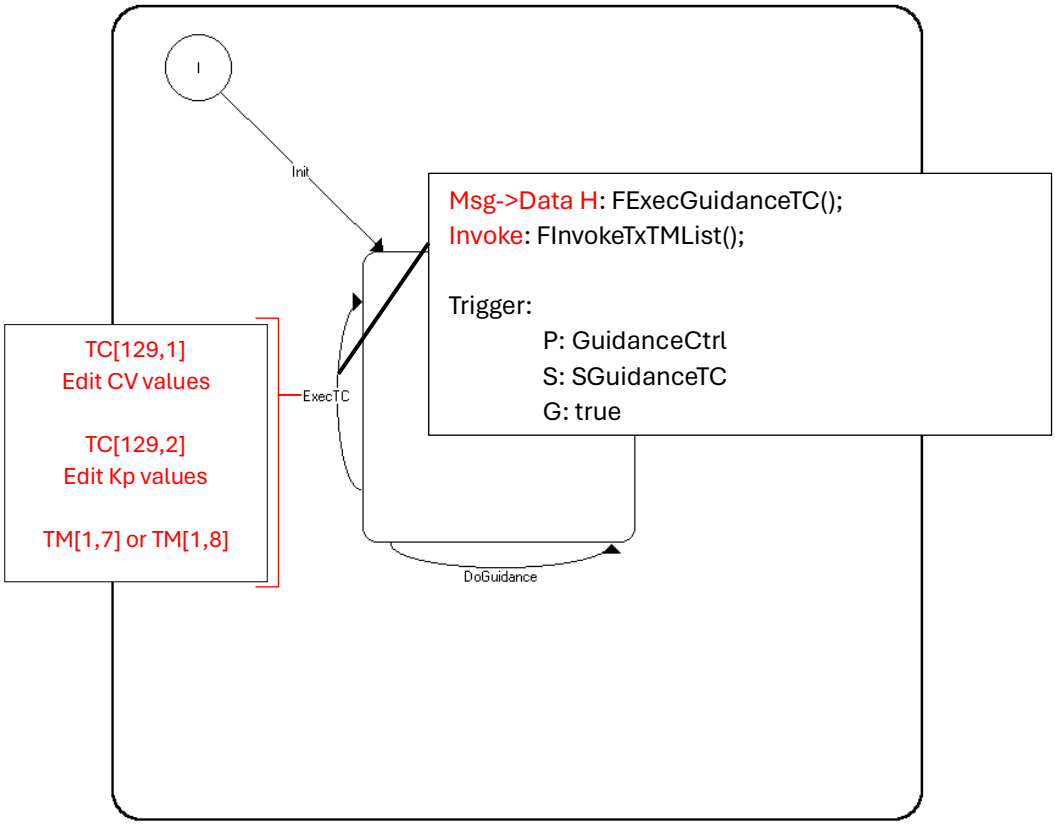
El primero de ellos, **EvGuidance**, consiste en la ejecución periódica de un método asociado al servicio 129, PUSService129::GuidanceControl(). Para ello inicializamos la tarea y programamos la transición "DoGuidance":




```
void FInitGuidance()
{
Pr_Time time;

time.GetTime(); // Get current monotonic time
time+=Pr_Time(0,100000); // Add X sec + Y microsec
VNextTimeout=time;

GuidanceTimer.InformAt(time);
}
```

```
void FGuidanceControl()
{
Pr_Time time;

VNextTimeout+= Pr_Time(0,100000); // Add X sec + Y microsec
time=VNextTimeout;
PUSService129::GuidanceControl(); ← MÉTODO DE CONTROL PERIÓDICO

GuidanceTimer.InformAt(time);
}
```

Finalmente con la transición “ExecTC” implementamos las funciones necesarias para la obtención y procesamiento de telecomandos del servicio 129.

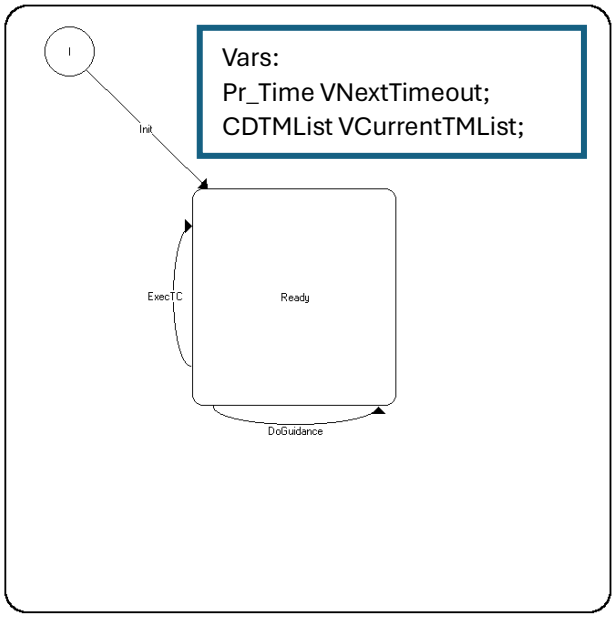



```
void FExecGuidanceTC ()
{
CDTCHandler & varSGuidanceTC = *(CDTCHandler *) Msg->data;

CDEventList TCExecEventList;
PUS_GuidanceTCExecutor::ExecTC(varSGuidanceTC,VCurrentTMList,TCExecEventList);

}
```

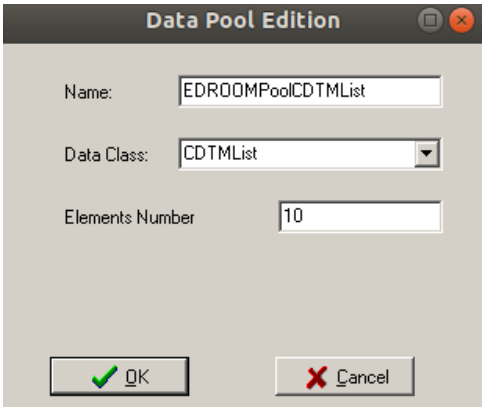
```
void FInvokeTxTMList ()
{
CDTMList * pSTxTM_Data=EDROOMPoolCDTMList.AllocData();

*pSTxTM_Data=VCurrentTMList;
 VCurrentTMList.Clear();

MsgBack=TMChannelCtrl.invoke(STxTM,pSTxTM_Data,&EDROOMPoolCDTMList);
}
```

Para que el componente quede funcional, quedan por declarar las variables, la pool de telemetrías para enviar y definir el módulo externo asociado al componente.

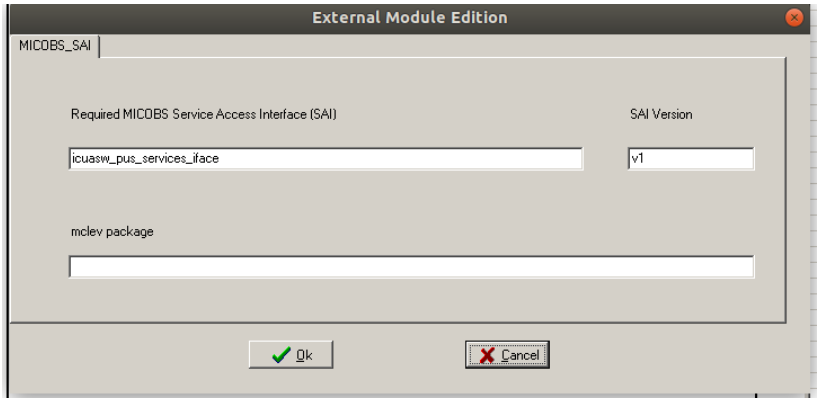
VARIABLES:



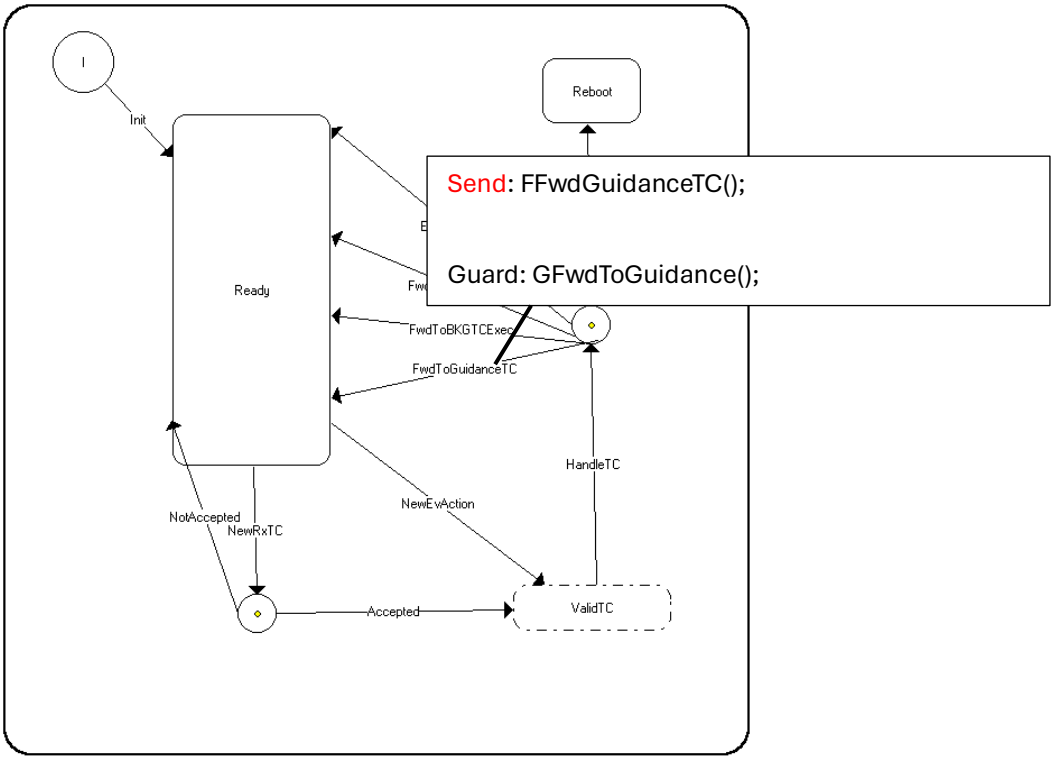
POOL:



MÓDULO EXTERNO:



Aunque el nuevo componente ya esté completamente definido en nuestro modelo, es necesario modificar **el comportamiento el Explorer Manager** para conseguir que le remita a nuestro nuevo componente los telecomandos de la clase 129. Para ello introducimos la rama "FwdToGuidanceTC" tal que:




```
void FFwdGuidanceTC ()
{
 CDTCHandler * pSGuidanceTC_Data = EDROOMPoolCDTCHandler.AllocData();

*pSGuidanceTC_Data=VCurrentTC;

GuidanceCtrl.send(SGuidanceTC ,pSGuidanceTC_Data,&EDROOMPoolCDTCHandler);
}
```

```
void GFwdToGuidance ()
{

return VCurrentTC.IsGuidanceTC();

}
```