HLA. Introducción a los entornos virtuales distribuidos

Tercer Encuentro de programadores Java. Universidad Jaume I.

1. Introducción

El presente documento pretende ser una pequeña introducción a las funcionalidades básicas que presenta la arquitectura *High Level Architecture* (HLA).

1.1 Antecedentes

El paso del sistema tradicional de simulación única y cerrada a nuevos sistemas distribuidos, abiertos e interactivos supuso la demanda de nuevas técnicas de desarrollo.

Por ello, en 1995 el *Defense Modeling & Simultion Office* (DMSO) *propone* HLA como elemento de unión para combinar simulaciones individuales dentro de una simulación global.

Con el tiempo esta propuesta se ha convertido en un estándar de la simulación (definido en IEEE 1516), demostrando que su uso no solo es aplicable a simulaciones de guerra, sino también a campos tan diversos como la educación, el entrenamiento o el análisis en ingeniería.

1.2 Definición

HLA nace con el propósito de facilitar al programador la interacción entre simulaciones de forma sencilla.

Para ello define una arquitectura software de alto nivel basada en componentes (federados), que intercambia objetos e interacciones a través de una capa de servicios. Estos servicios ofrecen funcionalidades básicas como suscripción a eventos o control de tiempo global de la simulación.

Para comprender la arquitectura es necesario conocer definiciones propias como:

- Federado: es el elemento fundamental de una simulación (componente).
- Federación: conjunto de federados.
- *Run-Time Infracturcture* (RTI): es una implementación de la especificación HLA (como por ejemplo el software Portico).
- Interacción: son estructuras para compartir información a través del RTI que se caracterizan por su carácter temporal. Un símil podría ser un mensaje a través del chat.
- Parámetro: son los componentes elementales de una interacción.

- Objeto: son estructura para compartir información a través del RTI que se caracterizan por su carácter persistente. No hay que confundir con la definición de objetos en programación, más bien su representación sería una clase.
- Atributo: son los componentes elementales de un objeto.

1.3 Características

Las características principales de la arquitectura son por una parte la reusabilidad de los modelos de federados entre diferentes simulaciones y aplicaciones, y por otra la capacidad de establecer comunicaciones entre plataformas heterogéneas de forma colaborativa.

Bajo estas dos principales propiedades podemos definir una serie de puntos que satisface un sistema HLA:

- Independencia de la plataforma.
- Especificación clara de la interfaz: identifica como deben interaccionar los federados con la federación y el RTI.
- Gestión de conexión y agrupaciones entre federados.
- Definición de estructuras de datos (objetos e interacciones).
- Gestión de propiedad para las estructuras declaradas como objetos.
- Gestión de la distribución de datos en una misma federación o entre federaciones.
- Gestión de políticas de tiempos.
- Control de tiempos y eventos.

1.4 Arquitectura y componentes

El estándar establece cuatro pilares fundamentales:

- La descripción de la especificación que debe cumplir el RTI y los federados.
- Las reglas de la federación, que especifica cómo interactúan los elementos de la simulación y sus responsabilidades.
- *Object Model Template* (OMT): Proporciona una plantilla común para documentar información clave, esto documentos describirán los elementos compartidos esenciales para la simulación distribuida.
- Prácticas recomendadas.

1.5 Reglas de la federación

Las reglas de la federación describen las responsabilidades de los federados, en sus relaciones con el RTI. Existen diez reglas. Cinco relacionadas con la federación y cinco con los federados.

Federación:

1. Las federaciones deben tener un documento llamando *Federation Object Model* (FOM), especificado de acuerdo con el OMT. En el que se describe los objetos e interacciones que un federado puede transmitir por la federación. El FOM representa el vocabulario común de la federación.
2. En la federación, todas las representaciones de objetos expresadas en el FOM deben estar en los federados, y no en el RTI. Los servicios del RTI son genéricos, para poder mantener la interoperabilidad del sistema.
3. Durante la ejecución de la federación, todos los intercambios de datos FOM deben realizarse a través del RTI.
4. Durante la ejecución de la federación, los federados deben interactuar con el RTI de acuerdo con la especificación impuesta por HLA.
5. Durante la ejecución de la federación, un atributo de una instancia de un objeto solo puede ser propiedad de un único federado para un tiempo dado.

Federado:

6. Los federados deben tener un *Simulation Object Model* (SOM), especificado de acuerdo con el OMT. En él se especificará sobre que objetos e interacciones se quiere publicar y subscribir.
7. Los federados deben actualizar y recibir los atributos de los objetos o enviar y recibir las interacciones especificadas en su SOM. La federación espera que el federado reaccione tratando correctamente los datos ofrecidos para el intercambio en el SOM.
8. Los federados deben transferir y/o aceptar la propiedad de los atributos dinámicamente durante la ejecución de la federación, de acuerdo con su especificación de SOM.
9. Los federados pueden modificar durante la ejecución las condiciones de los objetos e interacciones especificados en el SOM.
10. Los federados deben gestionar el tiempo local de forma que permita coordinar el intercambio de datos con otros miembros de la federación.

2. Proceso de comunicación

2.1 Componentes

La comunicación entre federados se realiza a través de un sistema de embajadores que gestionan el envió y recepción de eventos. Cada federado posee dos embajadores para establecer una comunicación bidireccional con el RTI:

El envío de datos del federado hacia el RTI se realiza a través del embajador del RTI (*RTIAmbassador*). Con este embajador realizamos todas las acciones que impliquen peticiones a los servicios del RTI, como por ejemplo peticiones de creación de federación, peticiones de avance de tiempo, actualizaciones de atributos, etc. Desde

el punto de vista práctico (basado en Portico) las peticiones sobre el servicio RTI se realizan como llamadas a métodos sobre un objeto del tipo *RTIAmbassador*.

La recepción de eventos se realiza a través del embajador del federado (*FederateAmbassador*). La interfaz que implementa este embajador comprende todas las funciones necesarias para recibir cualquier envío a través del RTI, como nuevos valores de actualizaciones de atributos, nuevas interacciones enviadas por otros federados, etc. Una vez recibidos los datos deben ser tratados según las reglas especificadas en HLA. Desde el punto de vista práctico (basado en Portico) la recepción se realiza a través de la implementación de una interfaz al estilo del patrón *observer*.

2.2 Fases

Una forma para comprender el proceso de comunicación consiste describir cada una de las fases que interviene y los servicios que en ella se realizan, estos se pueden dividir en seis categorías:

- Gestión de la federación. Control y ejecución.
- Gestión de declaraciones. Definición de datos publicados y subscritos.
- Gestión de objetos. Intercambio de objetos e interacciones.
- Gestión de propiedad. Intercambio de propietario de atributos.
- Gestión de tiempo. Control del orden de los mensajes.
- Gestión de distribución de datos.

2.3.1 Gestión de la federación

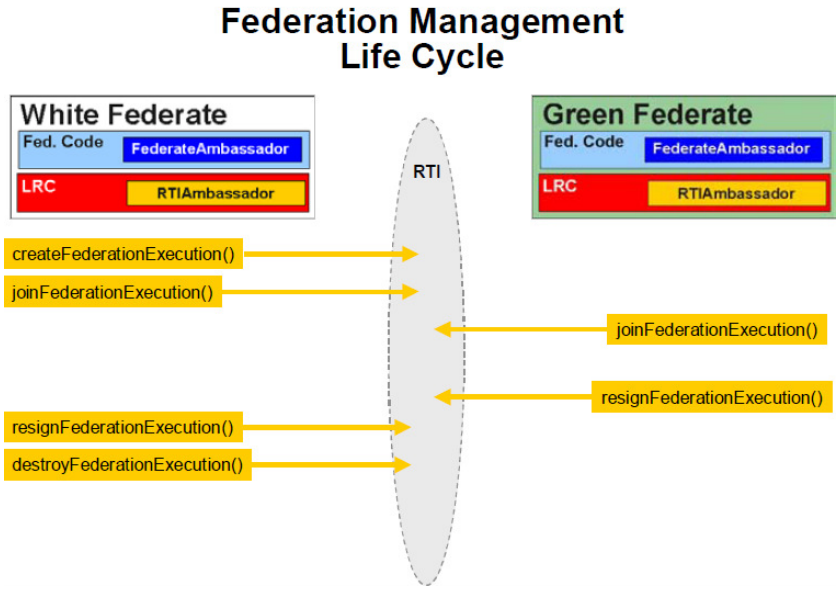
La gestión de la federación incluye principalmente la actividad de control de la ejecución de la federación. Una descripción apropiada de sus tareas son la proporciona por el propio manual de DMSO:

- Creación: “Jugamos un juego”.
- Unión de federados a la federación: “Yo quiero jugar”.
- Salvar y recuperar: “Salvemos nuestro estado, para poder recuperar la partida”.
- Control sobre los puntos de sincronismo: “Esperad, sincronicemos”.
- Desconectar federados de la federación: “Ahora quiero dejar el juego”.
- Destruir la federación: “Acabemos con el juego”.

Como introducción a la arquitectura HLA solo describiremos algunas de las funciones que consideramos clave y su esquema de comunicación:

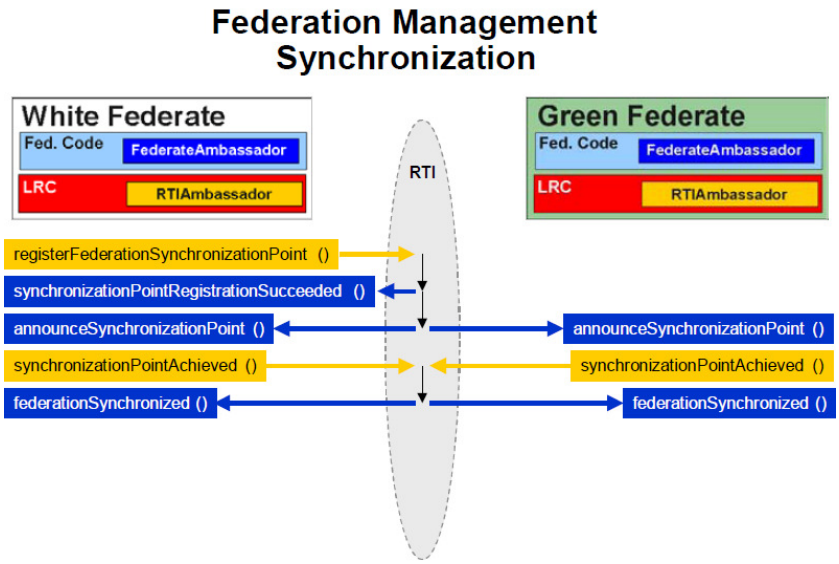
- *createFederationExecution*: Si la federación no existe, el proceso de ejecución del RTI crea un nuevo proceso de ejecución para la federación asociado al nombre pasado como argumento.
- *joinFederationExecution*: Asocia un federado a una ejecución de federación existente. Necesita además establecer el embajador del federado sobre el que actuarán las respuestas.

- *tick*: El RTI ejecuta una gran cantidad de carga de proceso, por lo que en determinados momentos necesita obtener el control del sistema de procesamiento.
- *regignFederationExecution*: Desconecta un federado de la federación.
- *destroyFederationExecution*: Termina la ejecución de una federación determinada.



Funciones para sincronización: Establece puntos de sincronización entre los federados.

- registerFederationSynchronizationPoint: Registra el nombre de un punto de sincronización
- synchonizationPointArchived: Establece el estado de preparado para sincronización.



Funciones para salvar y restaurar el estado.

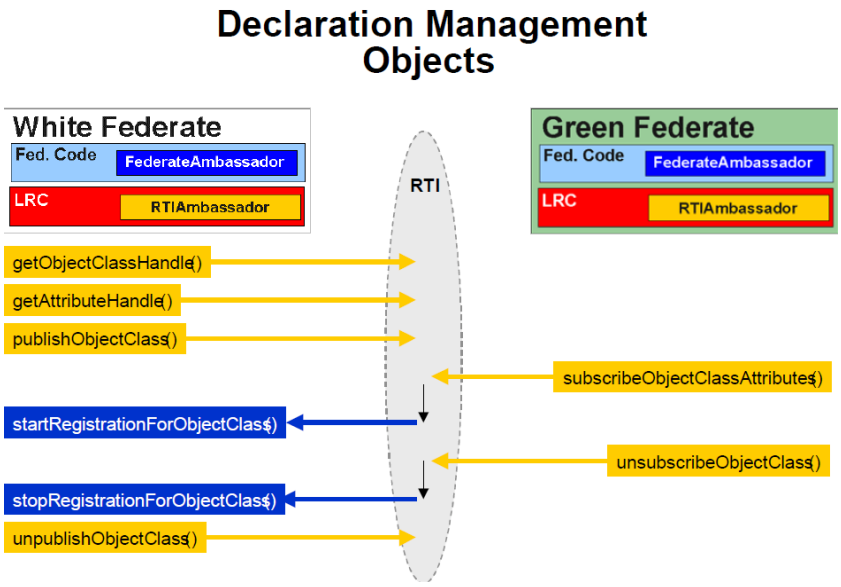
2.3.2 Gestión de declaraciones

Los federados deben especificar exactamente sobre que objetos e interacciones van a publicar o subscribir. Asociando las siguientes tareas:

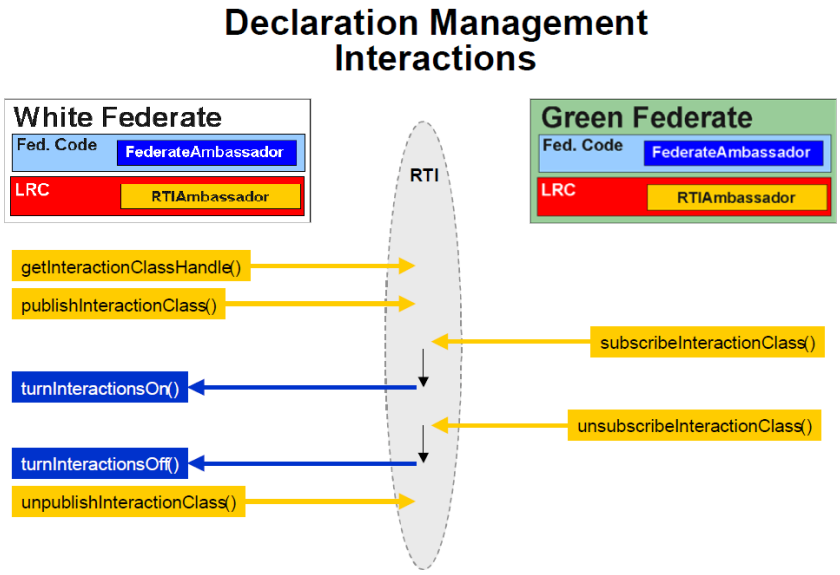
- Publicar: “Aquí está la información que compartiré”.
- Subscribir: “Yo lo quiero conocer todo de él”.
- Control: “Alguien me pregunta sobre que conozco”.

Para cada una de ellas el estándar describe las siguientes funciones:

- *getObjectClassHandle*: Obtiene el identificador de control de la clase para la estructura de datos tipo objeto.
- getAttributeHandle: Obtiene el identificador de control de los atributos especificados.
- publichObjectClass: Se define el federado como pretendiente a publicar sobre una clase “objeto”.
- subscribeObjectClassAttributes: Se define el federado como pretendiente a recibir las actualizaciones de unos atributos determinados.



- *getInteractionClassHandle*: Obtiene el identificador de control de la clase para la estructura de datos tipo interacción. No existe *getParameterClassHandle* porque las interacciones se realizan en bloque.
- *publichInteractionClass:* Se define el federado como pretendiente a publicar un tipo de interacciones determinadas.
- *subscribeInteractionClass*: Se define el federado como pretendiente a recibir un tipo de interacciones determinadas.



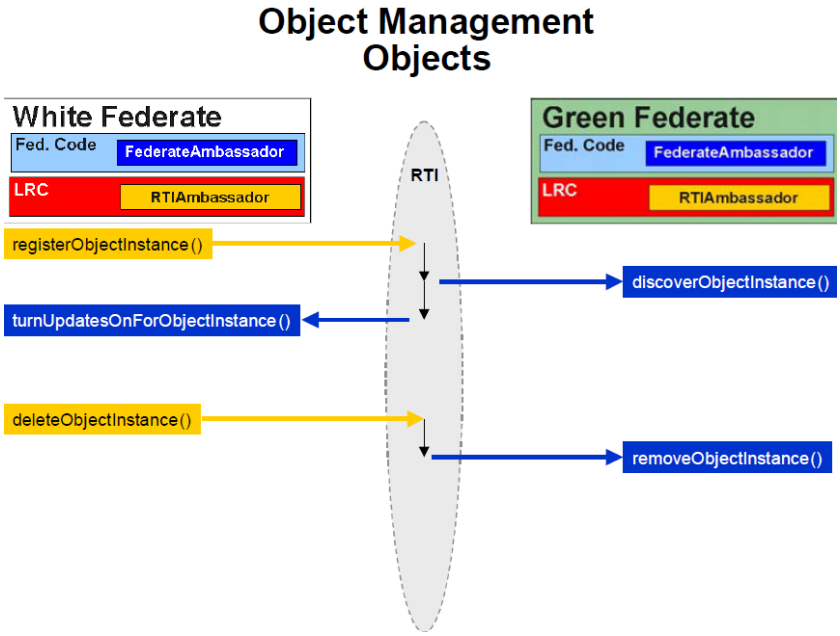
2.3.3 Gestión de objetos

La gestión de objetos incluye principalmente el registro de instancias y su actualización por parte de productor. Así como el descubrimiento de la instancia y su recepción de datos por parte de consumidor. Las tareas asociadas son:

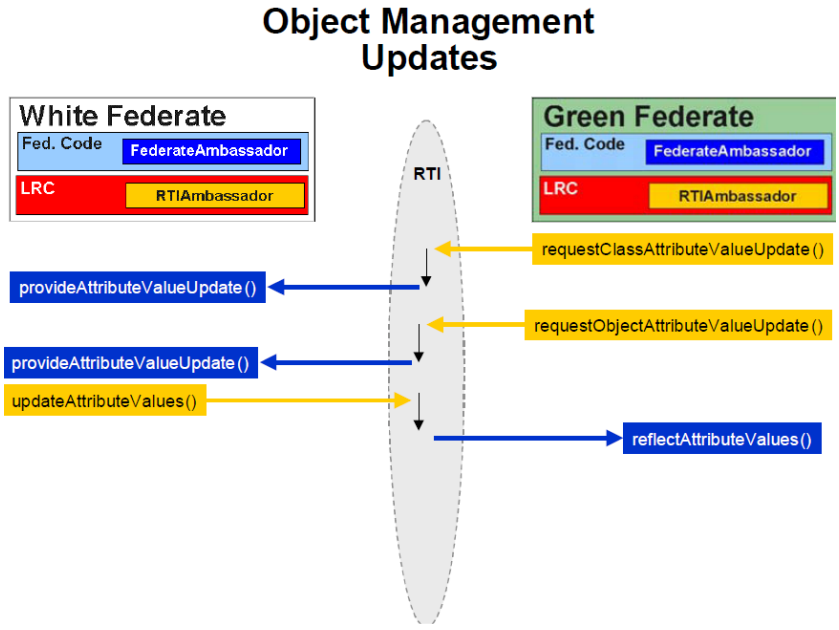
- Registrar un objeto: “Yo tengo un tanque nuevo”.
- Actualizar atributos: “Uno de mis aviones ha cambiado de dirección”.
- Enviar interacciones: “El vuelo 501 solicita permiso para aterrizar”.
- Eliminar objetos: “El camión esta fuera de servicio”.
- Cambio del protocolo de transporte.
- Cambio en el orden de recepción de eventos.

Para cada una de ellas el estándar describe las siguientes funciones:

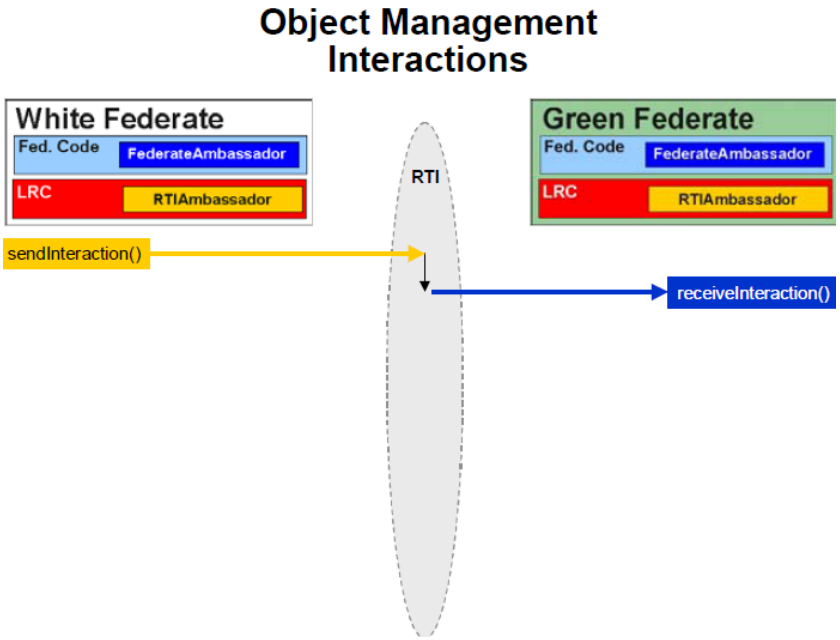
- *registerObjectInstance*: Registra una instancia de una clase del tipo de datos objeto sobre el federado local. Este tiene el control de las actualizaciones.
- *discoverObjectInstance*: Cada vez que se registra una nueva instancia por parte de algún federado el resto de federados subscritos a la clase del mismo tipo de objeto, reciben una llamada desde el RTI anunciando dicha instancia en el sistema.



- *provideAttributeValueUpdate*: Actualiza los valores de los atributos de una instancia determinada.
- *reflectAttributeValues*: Recibe por parte del RTI las actualizaciones de los valores de una instancia determinada.



- *sendInteraction*: Envía una interacción a través del RTI.



- *receiveInteraction*: Recibe por parte del RTI una interacción enviada por algún federado.

2.3.4 Gestión de propiedad

El RTI permite a los federados distribuir la responsabilidad de actualizar y eliminar instancias de objetos, con unas pequeñas restricciones. Tiene asociando las siguientes tareas:

- Ofrecer: “Alguien quiere simular el efecto del radar”.
- Adquirir: “Gracias, yo acepto la responsabilidad de posicionar este tanque”.
- Consultar: “¿Quien está gestionando el fuel del camión?”.

2.3.5 Gestión de tiempo

El objetivo de la gestión de tiempo es proporcionar mecanismos que permitan establecer políticas y negociaciones sobre el avance del tiempo. Sus tareas asociadas son:

- Establecer una política: “Envíame los eventos en orden de tiempo lógico”.
- Preguntar el tiempo: “¿Qué hora es?”.
- Intervalos: “Te proporciono 20 minutos para realizar los cambios”.
- Avance de tiempo: “Mueve el tiempo actual a 5 segundo más”.
- Próximo evento: “Mueve hasta el próximo evento”.
- Vaciar cola.

En una federación *HLA* podemos encontrarnos presentes diferentes gestiones de políticas de tiempos. Cada federado puede elegir la que más le convenga de entre un rango de posibles alternativas. El seguimiento de una u otra política marcará el ritmo de funcionamiento y actualización de datos del federado.

Cada evento en una federación puede estar asociado a un instante de tiempo, de forma que cuando el tiempo local del federado llegue a esa marca de tiempo, el RTI le hará llegar los eventos generados por otros federados que están asociados a esa marca de tiempo.

La única restricción que un federado está obligado a cumplir es que el tiempo siempre debe avanzar hacia delante. Sin embargo, cada federado puede tener su propia visión del tiempo actual. A medida que vaya avanzando en su tiempo local y este tiempo con la marca de tiempo de los eventos generados el federado recibirá por tales eventos.

Según su política de gestión de tiempos, los federados pueden ser:

- Reguladores (*Regulating*): Asocian un instante de tiempo a los eventos que generan. Marcan el ritmo de avance de los federados regulados, los cuales irán avanzando su tiempo al ritmo que vayan recibiendo eventos con marca de tiempo.
- Regulados (*Constrained*): Son capaces de recibir eventos con marca de tiempo u ordenados en el tiempo. Su ritmo de avance respecto al tiempo viene determinado por los federados reguladores.
- Otros: No son reguladores ni regulados. Pueden generar y recibir mensajes con marca de tiempo, pero no toman conciencia de ella. Es decir, no ordenan los mensajes recibidos respecto al tiempo.

Por tanto, el tipo de política seguida por los federados determina si los mensajes o eventos serán ordenados respecto al tiempo o no.

Una de los aspectos más sobresalientes del *HLA* es la capacidad de sincronización de sus federados respecto a un tiempo determinado. Haremos uso de ella cuando necesitemos que todos nuestros federados inicien su trabajo al mismo tiempo respecto a un punto de tiempo determinado. A partir de la unión en ese punto de tiempo todos empezarán sincronizados a partir de entonces. Para establecer una sincronización, la federación sigue una serie de etapas:

1. El federado interesado registra el punto de sincronización.
2. La federación anuncia ese punto a todos los federados.
3. El punto de sincronización se bloquea hasta que se cumpla cierta condición. En el momento que se cumpla esta condición todos los federados bloqueados en ese punto reemprenderán su marcha sincronizados.
4. Cada federado que llega al punto de sincronización lo notifica a la federación.
5. Cuando todos los federados han llegado al punto, la federación notifica a todos y cada uno de ellos que están totalmente sincronizados en ese punto.

2.3.6 Servicios de soporte

Otros servicios sobre la distribución de datos, permite crear, modificar, destruir, registrar y controlar regiones.