Sistemas Distribuidos de Tiempo Real

Apuntes: TEMA 1

Por: J. Javier Gutiérrez gutierjj@unican.es
http://www.ctr.unican.es/

Grupo de Computadores y Tiempo Real, Universidad de Cantabria

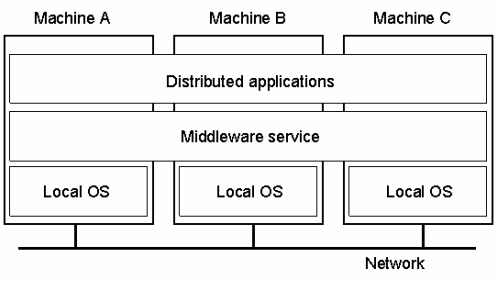
Sistemas distribuidos de tiempo real

PARTE I: Distribución y tiempo real

- **TEMA 1. Conceptos básicos de distribución y tiempo real**
- **TEMA 2. Comunicaciones de tiempo real**

Sistema distribuido

Conjunto de computadores que se presentan al usuario como un único sistema coherente (Tanenbaum)



Transparencia en el sistema distribuido

- Acceso: esconder diferencias en la representación de los datos y en el acceso a los recursos
- Localización: esconder dónde están los recursos
- Migración: esconder qué recursos pueden cambiar de sitio
- Relocalización: esconder el hecho de que un recurso se pueda mover de sitio mientras se está utilizando
- Sincronización y concurrencia: esconder que un recurso puede ser compartido por varios usuarios (consistencia)
- Replicación de recursos
- Fallo: esconder el fallo y el modo de recuperación
- Persistencia: esconder si un recurso está en memoria o en disco

Tiempo real y distribución

En un sistema de tiempo real lo importante es que los resultados (además de ser correctos) se produzcan a tiempo

A muy alto nivel, en la distribución en tiempo real normalmente se debe prescindir:

- de aspectos dinámicos como la migración y relocalización de código
- de esconder determinadas cosas como la localización de los servicios, o la persistencia

Es necesario evaluar temporalmente todo el software (incluido el de la red) y extraer los modelos para que puedan ser analizados:

- implementación del middleware con pautas más estrictas

Tiempo real y distribución (cont.)

Otros aspectos de la distribución que son menos relevantes o no resueltos en tiempo real son:

- El servicio de nombres se puede sustituir por un sistema más sencillo y normalmente cerrado
- Las réplicas van más dirigidas a la tolerancia a fallos. En tiempo real se utilizan otros mecanismos más sencillos y predecibles
  - control de sobrepaso del tiempo de ejecución
  - o simplificación de elementos en sistemas de seguridad crítica
- Aspectos como la seguridad no son tan importantes al tratarse de sistemas normalmente cerrados
  - empiezan a tener una importancia creciente
- Sistemas de ficheros distribuidos

Tiempo real y distribución (cont.)

Los aspectos importantes para tiempo real siguen siendo:

- modelo de concurrencia (software y hardware) y las políticas de planificación
- la sincronización que ahora se hace más compleja si se accede a recursos compartidos remotos

Aparecen nuevos aspectos:

- sincronización de relojes cuando sea necesario
- uso de redes de comunicaciones predecibles
- conceptos relativos a la calidad de servicio que cada vez se mezclan más con los de tiempo real

Conceptos del software de distribución

Sistema operativo distribuido:

- normalmente para sistemas fuertemente acoplados con procesadores homogéneos
- esconde y maneja el hardware

Sistema operativo de red:

- sistemas débilmente acoplados con procesadores heterogéneos conectados por LAN o WAN
- ofrece servicios locales a clientes remotos

Middleware de distribución:

- proporciona el servicio de distribución de forma transparente sobre sistema operativo local

Modelos de distribución

Cliente-Servidor:

- modelo tradicional con servicios remotos

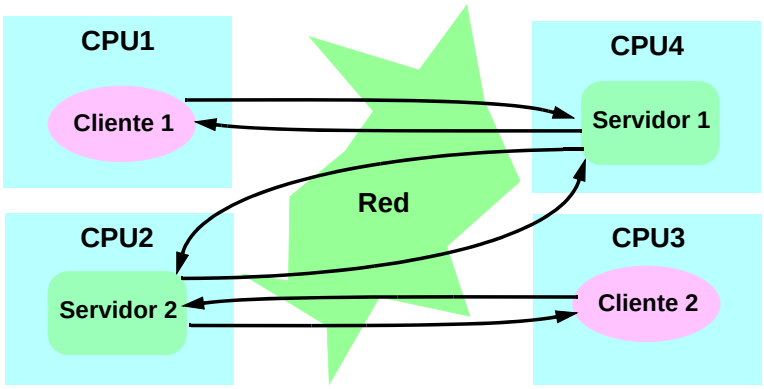
Modelo transaccional:

- transacciones formadas por actividades y eventos que cruzan por diferentes procesadores y redes

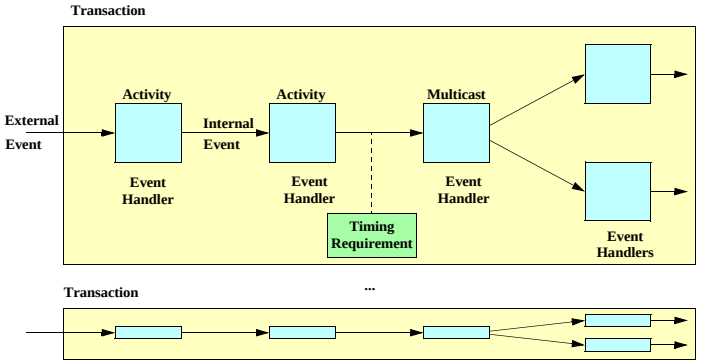
Servicios Web:

- oferta de servicios a través de web

Modelo cliente-servidor

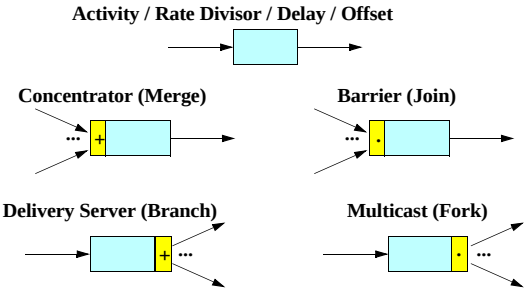


Modelo transaccional



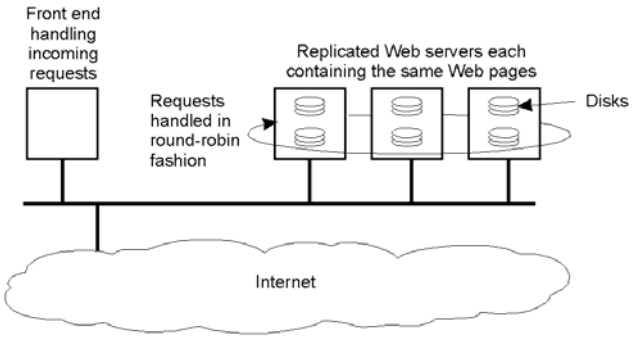
Modelo transaccional (cont.)

Combinaciones de eventos en el modelo transaccional



Servicios Web

Ejemplo de servicio web de distribución horizontal (Tanenbaum):



Mecanismos de distribución

Paso de mensajes:

- usa las comunicaciones tradicionales

RPC (Remote Procedure Call):

- procedimientos que se ofrecen como servicios remotos

Distribución de objetos:

- objetos que se pueden utilizar de forma remota

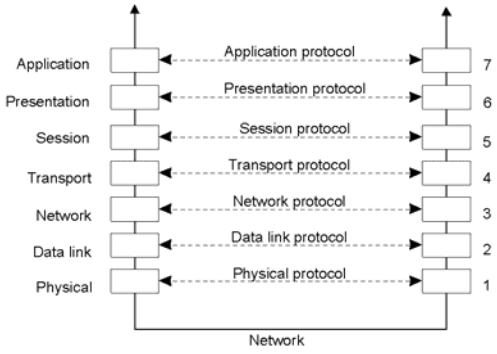
Distribución de componentes:

- aún en desarrollo, extiende el concepto de componente (elemento enchufable de forma transparente) a la distribución

Mecanismo Publicación/Suscripción

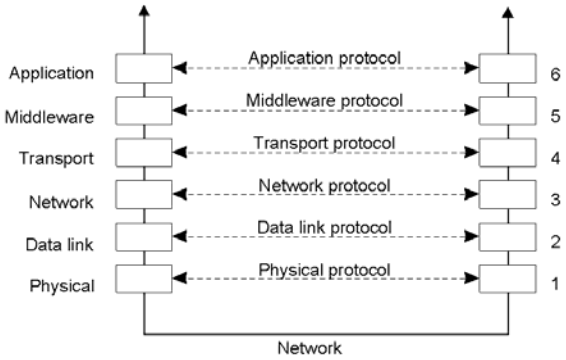
Paso de mensajes

Modelo de comunicaciones OSI (Tanenbaum):



Paso de mensajes (cont.)

Modificación del modelo OSI para añadir un protocolo middleware (Tanenbaum)



Paso de mensajes (cont.)

Las primitivas de paso de mensajes podrían operar sobre colas de mensajes con operaciones como:

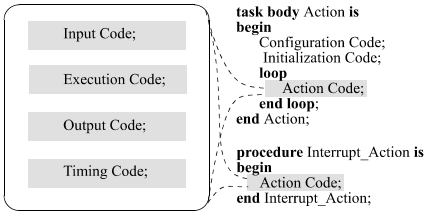
- Enviar con y sin espera
- Recibir con y sin espera
- Notificar

En tiempo real para sistemas planificados por prioridades fijas se pueden utilizar colas de prioridad

En cualquier caso la implementación debe ser cuidadosa de no utilizar mecanismos no predecibles (fuentes de inversión de prioridad no limitada)

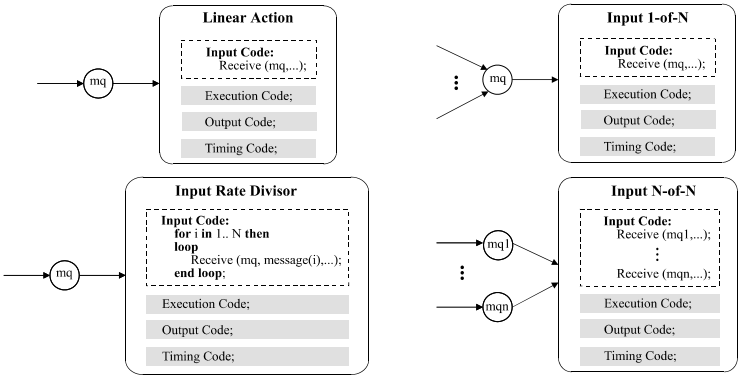
Transacciones, Ada y paso de mensajes

Modelo de acción periódica y de interrupción:



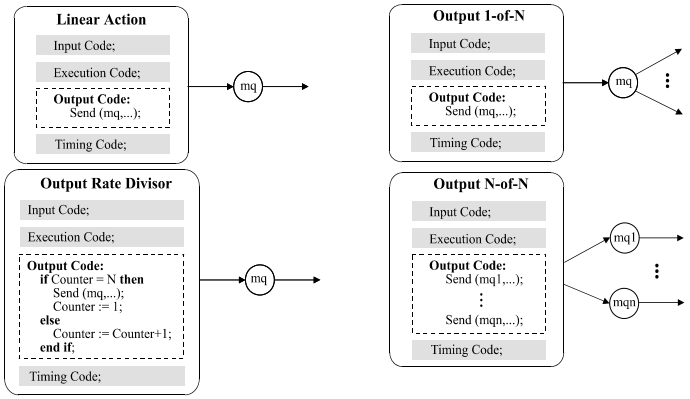
Transacciones, Ada y paso de mensajes (cont.)

Patrones de eventos a la entrada:



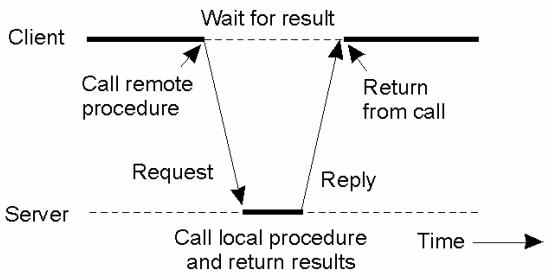
Transacciones, Ada y paso de mensajes (cont.)

Patrones de eventos a la salida:



RPC

Esquema temporal de llamada a procedimiento remoto (Tanenbaum):



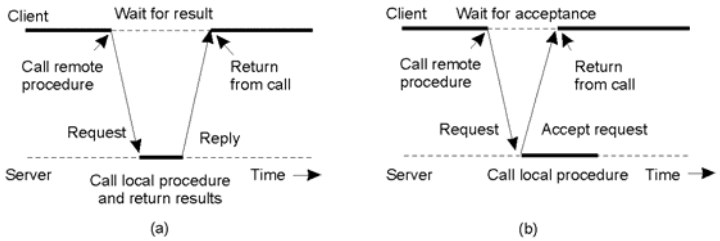
RPC (cont.)

Pasos en la llamada a un procedimiento remoto:

1. El cliente llama al *stub* del cliente
2. El *stub* del cliente construye el mensaje (*marshalling*)
3. Se envía el mensaje por la red
4. Se recibe el mensaje y se envía al *stub* de servidor
5. El *stub* de servidor desempaqueta los parámetros y llama al servidor (*unmarshalling*)
6. El servidor realiza el trabajo y devuelve el resultado al *stub*
7. El *stub* de servidor construye el mensaje de vuelta
8. Se envía el mensaje de vuelta al cliente
9. Se recibe el mensaje y se pasa al *stub* del cliente
10. El *stub* del cliente desempaqueta el resultado y se lo pasa al cliente

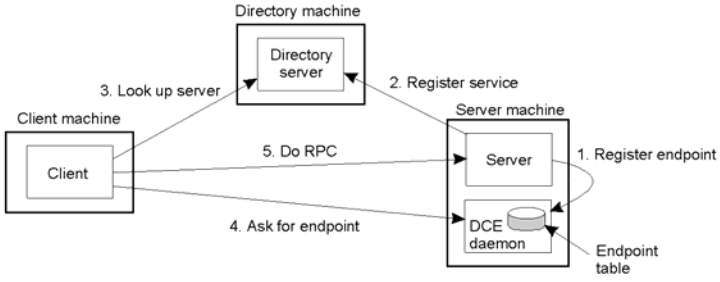
RPC asíncrono

El cliente no espera el retorno de la llamada al APC (Asynchronous Procedure Call, figura b Tanenbaum)



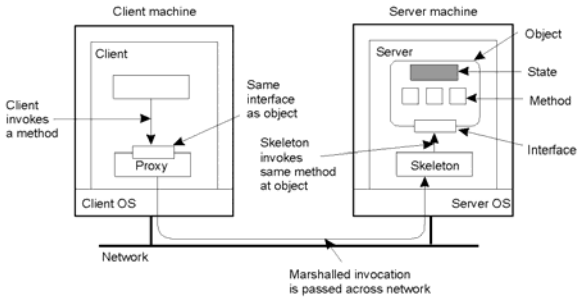
Servidor de nombres

Ejemplo de localización de servicio (Tanenbaum):



Distribución de objetos

Servicio sobre objeto remoto (Tanenbaum):



Publicación/Suscripción

En expansión a partir del DDS (Data Distribution Service) de la OMG:

- define un middleware que implementa un modelo de comunicaciones basado en el paradigma publicación-suscripción
- permite a un proceso en un entorno distribuido compartir datos independientemente de la localización física o la arquitectura del resto de procesos
- se puede ver como una variante avanzada del modelo cliente-servidor

Middleware de distribución

Anexo distribuido del lenguaje Ada 95 (anexo E, DSA, Distributed Systems Annex):

- no contempla tiempo real explícitamente, aunque se puede usar el anexo de tiempo real (anexo D)
- sólo para el lenguaje de programación Ada
- implementación del DSA por Ada-Core: GLADE
- existen otras implementaciones comerciales que se usan para tiempo real
- modificación de GLADE: RT-GLADE
- implementación de PolyORB debida a Ada-Core
  - middleware esquizofrénico que permite combinar la distribución en Ada con CORBA

Middleware de distribución (cont.)

CORBA (Common Object Request Broker Architecture):

- estándar del OMG
- estándar RT-CORBA para tiempo real
- multilenguaje: C, C++, Ada, Java
- ampliamente utilizado, aunque algunas implementaciones de tiempo real son más experimentales que comerciales
- ROFES, TAO, ORBit

Java RMI (Remote Method Invocation):

- tampoco contempla tiempo real
- sólo para lenguaje Java

Requisitos del middleware de distribución de tiempo real

A partir de los conceptos de RT-CORBA:

- ***Entidad planificable***:
  - se define la actividad como un concepto de diseño que utiliza los threads del sistema operativo subyacente para su implementación
  - opcionalmente, hay un servicio de planificación de prioridades fijas para ayudar a los programadores a planificar las actividades
- ***Interfaces para el control de prioridades de las entidades planificables***:
  - se define una prioridad universal con un esquema independiente de la plataforma
  - existen funciones de mapeado de estas prioridades "globales" a las nativas de cualquier planificador dado y viceversa

Requisitos del middleware de distribución de tiempo real (cont.)

- ***Mecanismos de propagación de prioridades del cliente al servidor***:
  - se define una política de propagación de prioridad para determinar la prioridad a la que el servidor maneja las peticiones de los clientes
  - dos modelos principales de propagación
  - ***Client_Propagated***: el servidor ejecuta a la prioridad del cliente
  - ***Server_Declared***: el servidor ejecuta a una prioridad determinada al crear el servicio

Requisitos del middleware de distribución de tiempo real (cont.)

- *Mecanismos para evitar la inversión de prioridad no acotada*:
  - interfaz de mutex
  - políticas para especificar y configurar los protocolos de comunicaciones
  - un *pool* de threads para manejar la ejecución de los servicios en el lado del servidor
  - políticas que permiten al cliente establecer múltiples conexiones y especificar el uso de conexiones no multiplexadas
  - mecanismos de transformación de prioridad en el lado del servidor para implementar protocolos de prioridad
- *Define "recursos" para el manejo de recursos*: threads, pool de threads, conexiones de transporte, y buffers de peticiones

Requisitos del middleware de distribución de tiempo real (cont.)

- *Mecanismos para la ubicación de recursos*:
  - interfaz de mutex para coordinar la contención
  - manejo de prioridades de los threads
  - API para el manejo del pool de threads, políticas de protocolo y la operación de enlace explícito para manejar conexiones de transporte
- *Otros requerimientos opcionales*:
  - timeouts
  - interfaz para la instalación de protocolos de transporte definidos por el usuario
  - interacción entre objetos de tiempo real y de no tiempo real
  - interfaces de tiempo de ejecución para las entidades planificables

Comunicaciones de tiempo real

El middleware de distribución de tiempo real debe estar soportado por un sistema operativo de tiempo real pero también por unas redes de comunicaciones de tiempo real

En general los mensajes tienen que competir por el uso del medio físico de comunicaciones:

- en tiempo real estricto es necesario disponer de mecanismos deterministas que permitan predecir el tiempo que puede estar un mensaje bloqueado

Los protocolos tradicionales asociados con Ethernet no soportan este tipo de comunicación puesto que encolan los mensajes en orden FIFO y el acceso al medio es CSMA/CD (Carrier Sense Multiple Access with Collision Detection)

Comunicaciones de tiempo real (cont.)

Problemas asociados a la planificabilidad de las redes:

- A diferencia de los procesadores, que tienen un único punto de acceso, las redes tienen múltiples puntos (uno por cada nudo físico conectado), por lo que se necesita un protocolo distribuido
- Los algoritmos expulsores son apropiados para los procesadores, pero la expulsión en las redes requeriría en principio la retransmisión del mensaje expulsado
- Además de los plazos impuestos a los procesos de aplicación, es necesario controlar los tiempos de disponibilidad de los datos en los buffers de transmisión (los datos se deben transmitir antes de colocar otros nuevos)

Comunicaciones de tiempo real (cont.)

Normalmente el problema de la expulsión se resuelve partiendo los mensajes en paquetes de una longitud máxima fija:

- se convierten en la unidad mínima no expulsable
- su tiempo de transmisión se contabiliza como un bloqueo

Entre otros protocolos de comunicaciones, se puede garantizar tiempo real con los siguientes esquemas de planificación en las redes:

- TDMA (Time Division Multiple Access)
- Protocolos de paso de testigo
- Protocolos basados en prioridades

TDMA

Es la extensión natural del ejecutivo cíclico de la planificación monoprocesadora

Características principales:

- cada nudo tiene un reloj sincronizado con los relojes de los demás nudos (es uno de sus mayores inconvenientes)
- durante un ciclo de comunicaciones cada nudo dispone de un *slot* en el que puede comunicar
- no existen colisiones de mensajes puesto que el hecho de que un nudo esté transmitiendo significa que los demás no lo hacen

TDMA (cont.)

Al igual que en el ejecutivo cíclico de los procesadores, la dificultad está en construir el esquema que hace que la aplicación sea planificable

Esta dificultad se incrementa exponencialmente con el número de nudos del sistema

Otra dificultad está en el tratamiento de mensajes esporádicos

- este protocolo requiere un plan cíclico de mensajes periódicos

Un ejemplo de este protocolo es el usado en MARS (MAintainable Real-time Systems) por Kopetz y otros (1985-1989).

Protocolos de paso de testigo

El acceso a la red se realiza cuando se posee un elemento especial llamado testigo:

- se va pasando como un mensaje especial entre los nudos
- los nudos sólo pueden transmitir los mensajes disponibles cuando están en posesión del testigo
- en los protocolos temporizados el nudo sólo puede disponer del testigo durante un tiempo máximo, de manera que hay un tiempo acotado de rotación del testigo
- en otras aproximaciones no temporizadas sólo se permite enviar un paquete cada vez que se posee el testigo
- no hay colisiones; sólo si se tiene el testigo se transmite
- hay mecanismos de recuperación del testigo si se pierde

Protocolos de paso de testigo (cont.)

Un ejemplo de uso de este protocolo es el FDDI (Fiber Distributed Data Interface):

- es un protocolo temporizado en el que los mensajes se agrupan en dos clases: síncronos y asíncronos
- los mensajes síncronos tienen restricciones de tiempo real y son los que se usan para definir el tiempo de posesión del testigo y por tanto el tiempo de rotación del testigo
- los mensajes asíncronos no tienen restricciones temporales, se envían si no hay mensajes síncronos que enviar, o si el testigo se ha recibido demasiado pronto (porque otros nodos no tienen nada que transmitir)

Protocolos basados en prioridades

Dadas las ventajas que en la planificación de procesadores tiene el uso de las prioridades fijas es lógico suponer que se utilice este mismo esquema para planificar las redes

Estos protocolos tienen normalmente dos fases:

- arbitrio de prioridad: cada nudo indica la prioridad del mensaje que desea transmitir
  - al nudo con el mensaje de mayor prioridad se le concede el uso de la red
- el envío del mensaje propiamente dicho

La ventaja en el uso de estos esquemas está en tener un número de prioridades suficiente para poder planificar los mensajes de una aplicación con diferentes prioridades

Protocolos basados en prioridades (cont.)

Existen protocolos como CAN (Controller Area Network) que resuelven el arbitrio de prioridad por hardware:

- el mensaje consta de 8 bytes precedidos por un identificador:
  - 11 bits en el CAN estándar 2.0 A
  - 29 bits en el CAN extendido 2.0 B
- al comienzo de la transmisión cada nudo va escribiendo los bits de su identificador (que es la prioridad del mensaje)
- el protocolo funciona como una puerta AND:
  - si un nudo escribe un 0 todos los demás leen un 0
  - el 0 es dominante

Protocolos basados en prioridades (cont.)

El protocolo CAN funciona como sigue:

- si un nudo transmite un 0 continúa con el siguiente bit
- si un nudo transmite un 1 y lee un 1 continúa con el siguiente bit
- si un nudo transmite un 1 y lee un 0 se retira hasta el siguiente turno de arbitrio de la prioridad

El menor valor del identificador indica la mayor prioridad

RT-EP

RT-EP (Real-Time Ethernet Protocol) es un protocolo de paso de testigo basado en prioridades:

- el testigo circula sobre un anillo lógico:
  - número de estaciones fijo
  - cada nudo conoce la dirección de su sucesor
  - un nudo arbitrario crea el testigo y se convierte en *token_master*
  - el *token_master* es dinámico
  - el mensaje se parte en paquetes
- la comunicación se realiza en dos etapas:
  - arbitrio de prioridad
  - transmisión del mensaje de datos

RT-EP (cont.)

- arbitrio de prioridad:
  - el testigo viaja por el anillo visitando todos los nudos
  - cada estación chequea y actualiza si uno de sus paquetes tiene mayor prioridad
  - el testigo se envía al nudo sucesor
  - el proceso se realiza hasta que el testigo llega al *token_master*
- transmisión del mensaje de datos:
  - el *token_master* envía un mensaje a la estación con el paquete de mayor prioridad que tiene permiso para enviar
  - se envía el paquete con los datos
  - la estación receptora se convierte en *token_master*

AFDX networks

ARINC-664 standard for networks

- Part 7, AFDX (Avionics Full Duplex Switched Ethernet)

AFDX characteristics:

- point to point full duplex Ethernet links (redundant)
- special purpose switches with preconfigured routing
- two main types of communication ports:
  - *Sampling Port*: the arriving message overwrites the current message stored in the buffer
  - *Queueing Port*: the arriving message is appended to a FIFO queue
- UDP/IP protocol is used for transmission
- traffic regulation is made in transmission via *Virtual Links*

Traffic regulation in AFDX

Each virtual link (VL) is characterized by two parameters:

- the largest Ethernet frame (Lmax): a value in bytes that can be transmitted on the VL (in the interval [64,1518])
- the Bandwidth Allocation Gap (BAG):
  - the minimum interval in milliseconds between Ethernet frames transmitted on the VL
  - a power of 2 value in the range [1,128]

Each virtual link has a FIFO queue for all the fragmented packets:

- the same VL can be shared by several ports, tasks or partitions
  - It can cause a poor schedulability of the system
- there is no way to prioritize messages on a VL

Sub-Virtual Links

A virtual link can be composed of a number of Sub-Virtual Links

Each Sub-VL has:

- a dedicated FIFO queue
- a round robin algorithm working over IP fragmented packets

AFDX switch

Transmissions to or from the switch are made using the full capacity of the physical link

Packets are delivered in a store and forward way:

- the hardware latency of the switch should be taken into account

Each output port of the switch has two FIFO queues (high and low priority) where packets should wait to be sent to the destination end system:

- priorities are selected on a VL basis