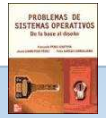
Grupo ARCOS
Universidad Carlos III de Madrid

Lección 6
Aspectos avanzados

Diseño de Sistemas Operativos
Grado en Ingeniería Informática

Lecturas recomendadas

Base



1. Carretero 2007:
   1. Cap.9

Recomendada

1. Tanenbaum 2006(en):
   1. Cap.5
2. Stallings 2005:
   1. Parte siete
3. Silberschatz 2006:
   1. Cap. 14 y 15

A recordar…

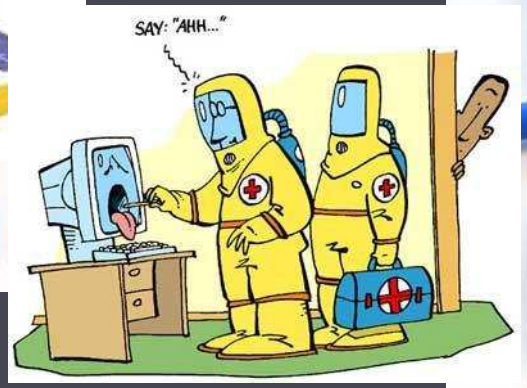
1. Estudiar la teoría asociada.
   - Estudiar el material asociado a la bibliografía: las transparencias solo no son suficiente.
   - Crear cuestiones con sus respuestas y justificación.
2. Repasar lo visto en clase.
   - Realizar el cuaderno de prácticas progresivamente.
3. Ejercitar las competencias.
   - Realizar las prácticas progresivamente.
   - Realizar todos los ejercicios posibles.

Contenidos

- Seguridad en sistemas operativos
- Tolerancia a fallos en SSOO
- Sistemas distribuidos
- Sistemas de tiempo real

Seguridad y Protección en Sistemas Operativos

Diseño de Sistemas Operativos. Aspectos Avanzados



Seguridad en SS.OO.
Conceptos

- Protección, Seguridad y Salvaguarda:
  - Son tres conceptos diferentes y, a menudo, confundidos.
  - Veamos si gráficamente los distinguimos mejor:
    - Protección (*protection*): limitador de velocidad =>

    - Seguridad (*security*): alarma para el coche

    - Salvaguarda (safety): airbags =>

Seguridad en SS.OO.
Conceptos

- ¿Protección o Seguridad?
  - Protección (*protection*):
    - Consiste en evitar que se haga un uso indebido de los recursos que están dentro del ámbito del sistema operativo.
    - Ejemplo:
      - La memoria virtual sirve de mecanismo de protección porque evita el acceso desde un proceso a la memoria usada por otro proceso.
  - Seguridad (*security*):
    - Es necesario comprobar que los recursos sólo se usan por aquellos usuarios que tienen derechos de acceso a los mismos.
    - Ejemplo:
      - Hay que identificarse y autentificarse mediante usuario y clave antes de acceder a una sesión en un sistema operativo.

Seguridad en SS.OO.
Conceptos

- ¿Seguridad o Salvaguarda?
  - Seguridad (*security*):
    - Es necesario comprobar que los recursos sólo se usan por aquellos usuarios que tienen derechos de acceso a los mismos.
    - Ejemplo:
      - Hay que identificarse y autentificarse mediante usuario y clave antes de acceder a una sesión en un sistema operativo.
  - Salvaguarda (*safety*):
    - El funcionamiento nominal, mal funcionamiento o no funcionamiento del sistema informático no debe producir mayores consecuencias que aquellas identificadas como previstas.
    - Ejemplo:
      - Un mal funcionamiento de un determinado sistema informático no debe causar daños a humanos, a menos que esté previsto y en qué medida.

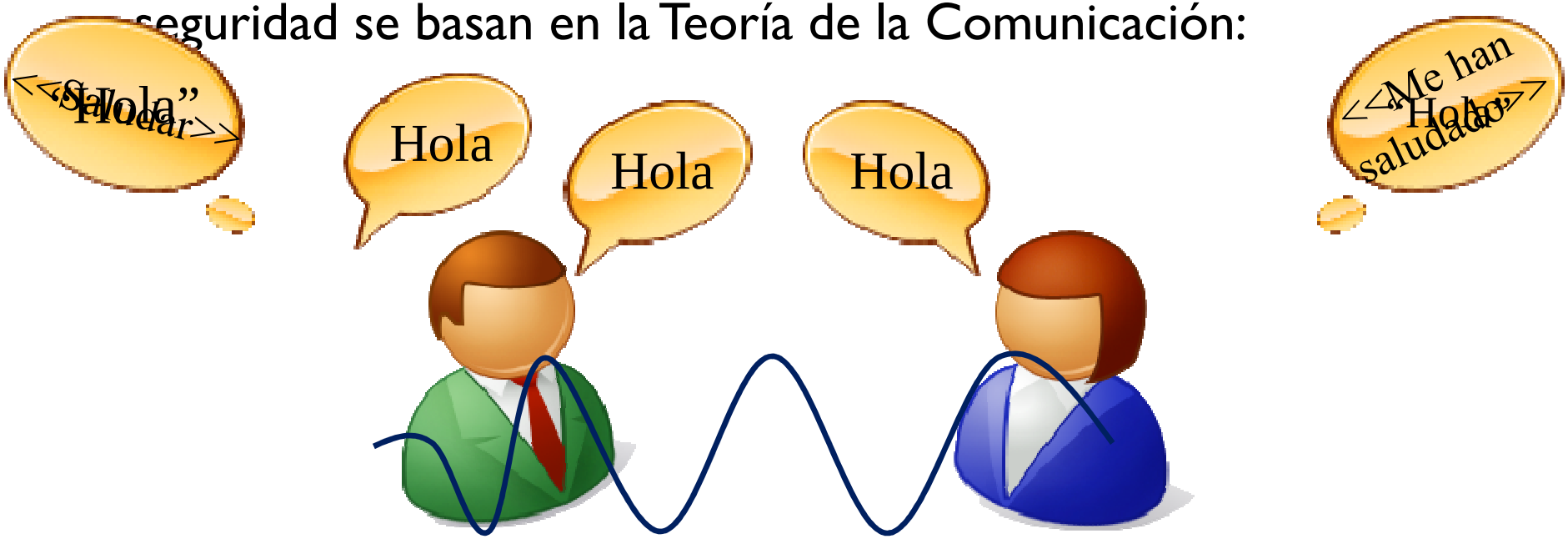
Seguridad en SS.OO.
Conceptos

- Seguridad: ¿Física o Lógica?
  - Seguridad física:
    - El acceso autorizado a los recursos se garantiza por medio de mecanismos físicos de acceso seguro.
    - Ejemplo:
      - Los controles armados de seguridad, puertas blindadas, etc.
  - Seguridad lógica:
    - El acceso autorizado a los recursos se garantiza por medio de mecanismos lógicos de acceso seguro.
    - Ejemplo:
      - Solicitud de usuario y clave, dispositivos biométricos, usos de cortafuegos, etc.
  - … Mejor ambas ☺

Seguridad en SS.OO.

Requisitos de Seguridad y Teoría de la Comunicación

- Tanto los requisitos de seguridad como las amenazas contra la seguridad se basan en la Teoría de la Comunicación:



Emisor
- Fase Codificación
- Fase Conceptual
- Fase Emisión

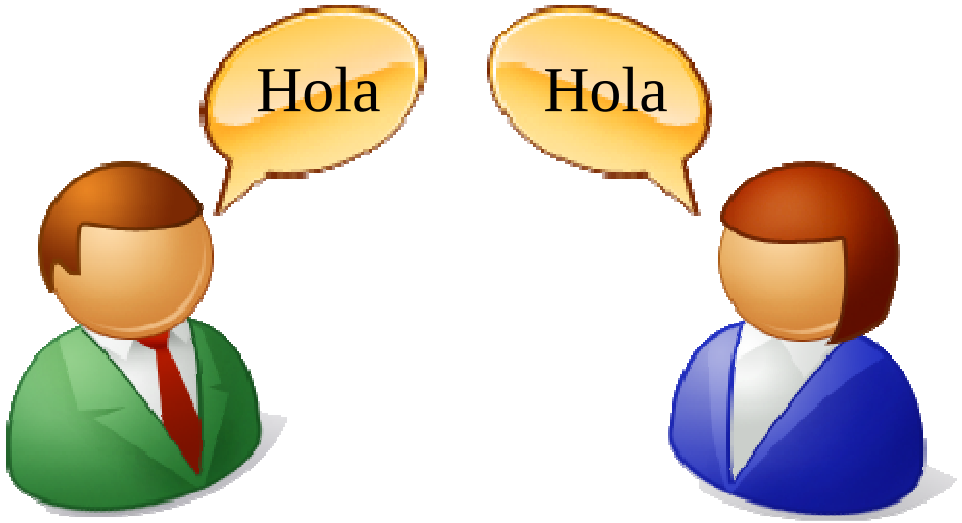
- Fase Transmisión

Receptor
- Fase Descodificación
- Fase Recepción
- Fase Conceptual

Seguridad en SS.OO.

Requisitos de Seguridad y Teoría de la Comunicación

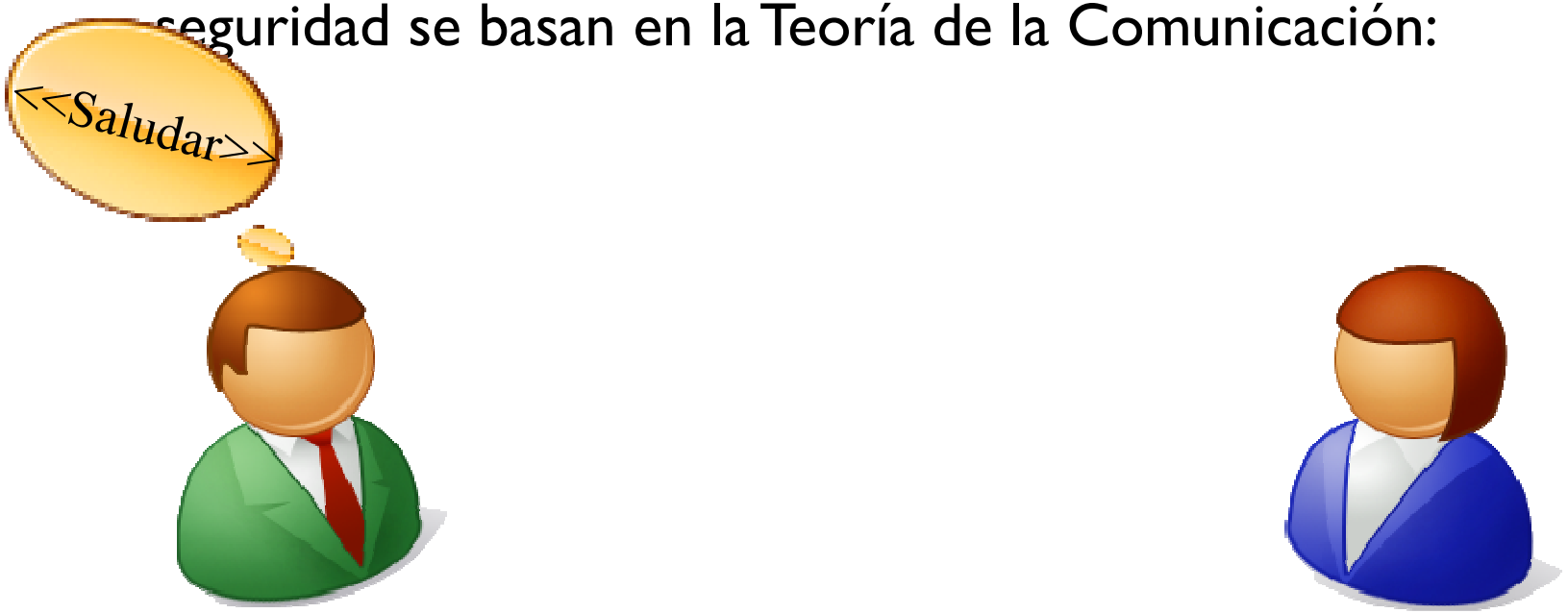
- Tanto los requisitos de seguridad como las amenazas contra la seguridad se basan en la Teoría de la Comunicación:



Seguridad en SS.OO.

Requisitos de Seguridad y Teoría de la Comunicación

- Tanto los requisitos de seguridad como las amenazas contra la seguridad se basan en la Teoría de la Comunicación:



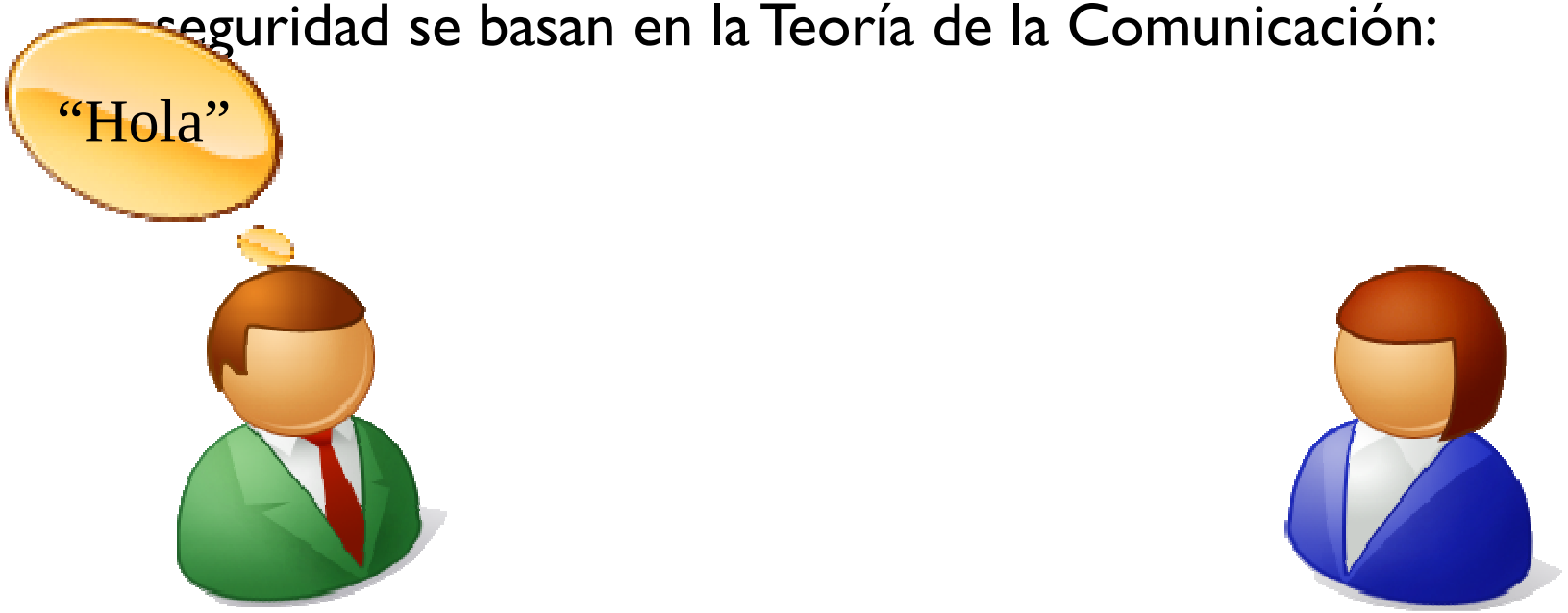
Emisor

- Fase Conceptual

Seguridad en SS.OO.

Requisitos de Seguridad y Teoría de la Comunicación

- Tanto los requisitos de seguridad como las amenazas contra la seguridad se basan en la Teoría de la Comunicación:



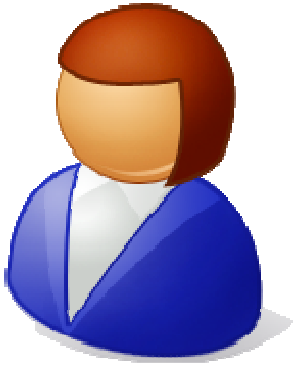
Emisor
- Fase Codificación

- Fase Conceptual

Seguridad en SS.OO.

Requisitos de Seguridad y Teoría de la Comunicación

- Tanto los requisitos de seguridad como las amenazas contra la seguridad se basan en la Teoría de la Comunicación:

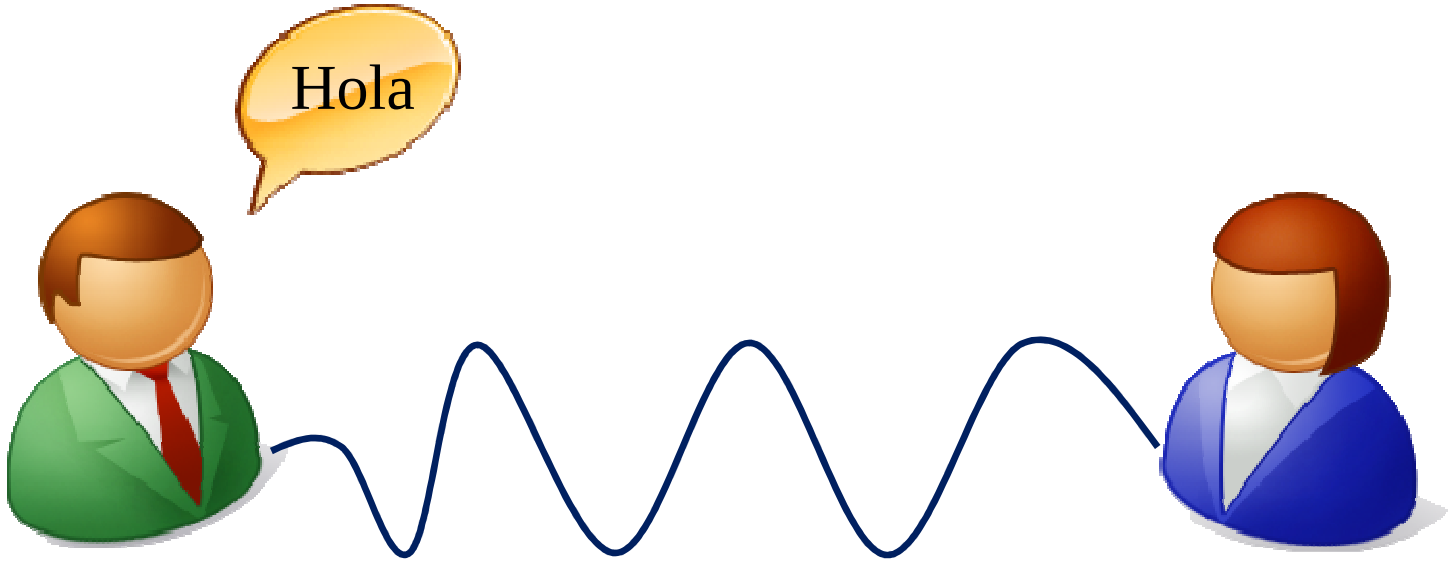


Emisor
- Fase Codificación
- Fase Conceptual
- Fase Emisión

Seguridad en SS.OO.

Requisitos de Seguridad y Teoría de la Comunicación

- Tanto los requisitos de seguridad como las amenazas contra la seguridad se basan en la Teoría de la Comunicación:



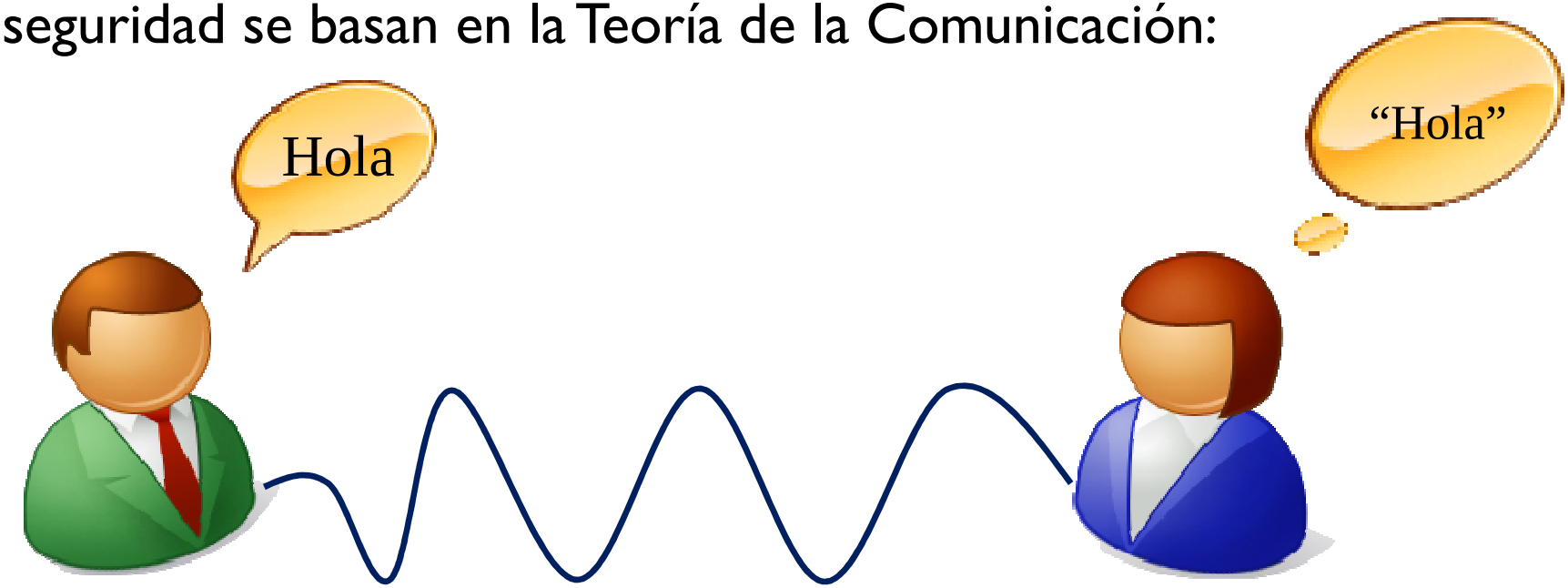
Emisor
- Fase Codificación
- Fase Conceptual
- Fase Emisión

- Fase Transmisión

Seguridad en SS.OO.

Requisitos de Seguridad y Teoría de la Comunicación

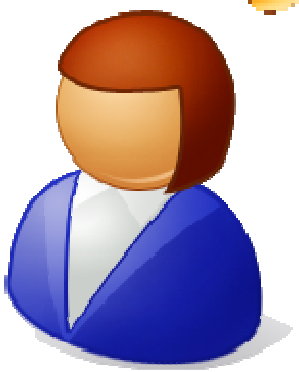
- Tanto los requisitos de seguridad como las amenazas contra la seguridad se basan en la Teoría de la Comunicación:



Seguridad en SS.OO.

Requisitos de Seguridad y Teoría de la Comunicación

- Tanto los requisitos de seguridad como las amenazas contra la seguridad se basan en la Teoría de la Comunicación:



Emisor
- Fase Codificación
- Fase Conceptual
- Fase Emisión

Fase Transmisión

Receptor
- Fase Descodificación
- Fase Recepción
- Fase Conceptual

Seguridad en SS.OO.
Requisitos de Seguridad

- Los requisitos de seguridad son tres:
  - Confidencialidad (o Secreto)
  - Integridad
  - Disponibilidad
    - Si bien algunos autores (Stallings) añaden "*Autenticidad*", la mayoría considera que está incluida como parte del "*Confidencialidad*".

Seguridad en SS.OO.
Requisitos de Seguridad: Confidencialidad

- Los requisitos de seguridad: Confidencialidad (o Secreto)
  - Exige que la información de un sistema informático sea accesible para lectura solamente por partes autorizadas.
  - Este tipo de acceso incluye impresión, visualización y otras formas de revelación, incluyendo el simple revelado de la existencia de un objeto.
  - Requiere, entre otros, que el sistema informático sea capaz de verificar la identidad de un usuario ("Autenticidad").

Seguridad en SS.OO.
Requisitos de Seguridad: Integridad

- Los requisitos de seguridad: Integridad
  - Exige que los elementos de un sistema informático puedan ser modificados sólo por partes autorizadas.
  - La modificación incluye escritura, cambio, cambio de estado, borrado y creación.

Seguridad en SS.OO.
Requisitos de Seguridad: Disponibilidad

- Los requisitos de seguridad: Disponibilidad
  - Exige que los elementos de un sistema informático estén disponibles para partes autorizadas.

Seguridad en SS.OO.
Amenazas contra la Seguridad

- Contra los requisitos de seguridad existen una serie de Amenazas (threads).
- Todos los elementos de un Sistema Informático están amenazados:
- ¡¡Incluso las Personas!!

Elemento	Confidencialidad	Integridad	Disponibilidad
Hardware			Robo o sobrecarga de equipos, eliminando el servicio
Software	Realización de copias no autorizadas del software	Alteración de un programa en funcionamiento haciéndolo fallar durante la ejecución o haciendo que realice alguna tarea no pretendida	Eliminación de programas denegando el acceso a los usuarios
Datos	Lecturas de datos no autorizadas. Revelación de datos ocultos de manera indirecta por análisis de datos estadísticos.	Modificación de archivos existentes o invención de nuevos	Eliminación de archivos, denegando el acceso a los usuarios
Líneas de Comunicación	Lectura de mensajes. Observación de la muestra de tráfico de mensajes.	Mensajes modificados, retardados, reordenados o duplicados. Invención de mensajes falsos.	Destrucción o eliminación de mensajes. Las líneas de comunicación o redes se hacen no disponibles.

Seguridad en SS.OO.

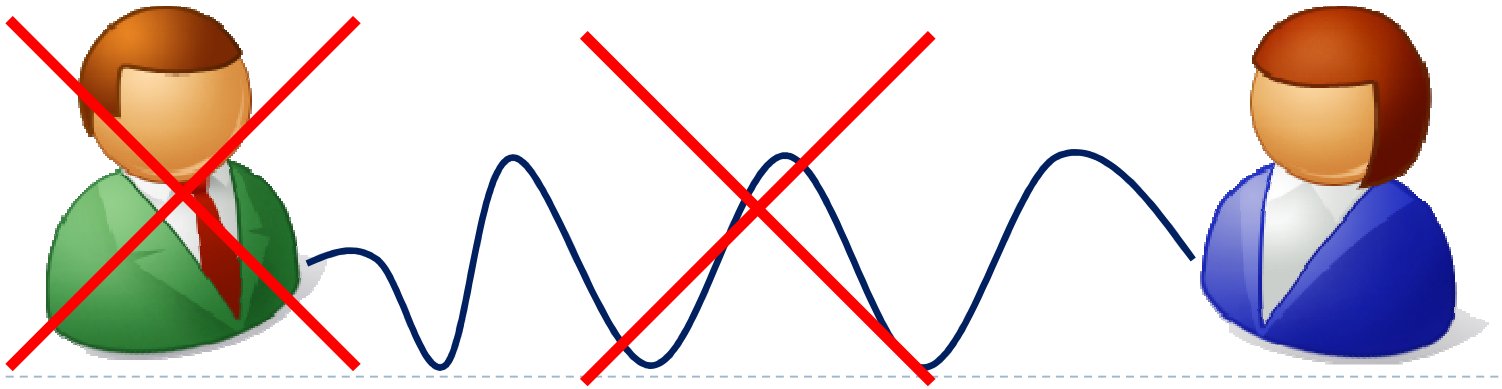
Amenazas contra la Seguridad

- Las amenazas existentes se categorizan en los siguientes cuatro tipos:
  - Interrupción
  - Intercepción
  - Modificación
  - Invención

Seguridad en SS.OO.

Amenazas contra la Seguridad: Interrupción

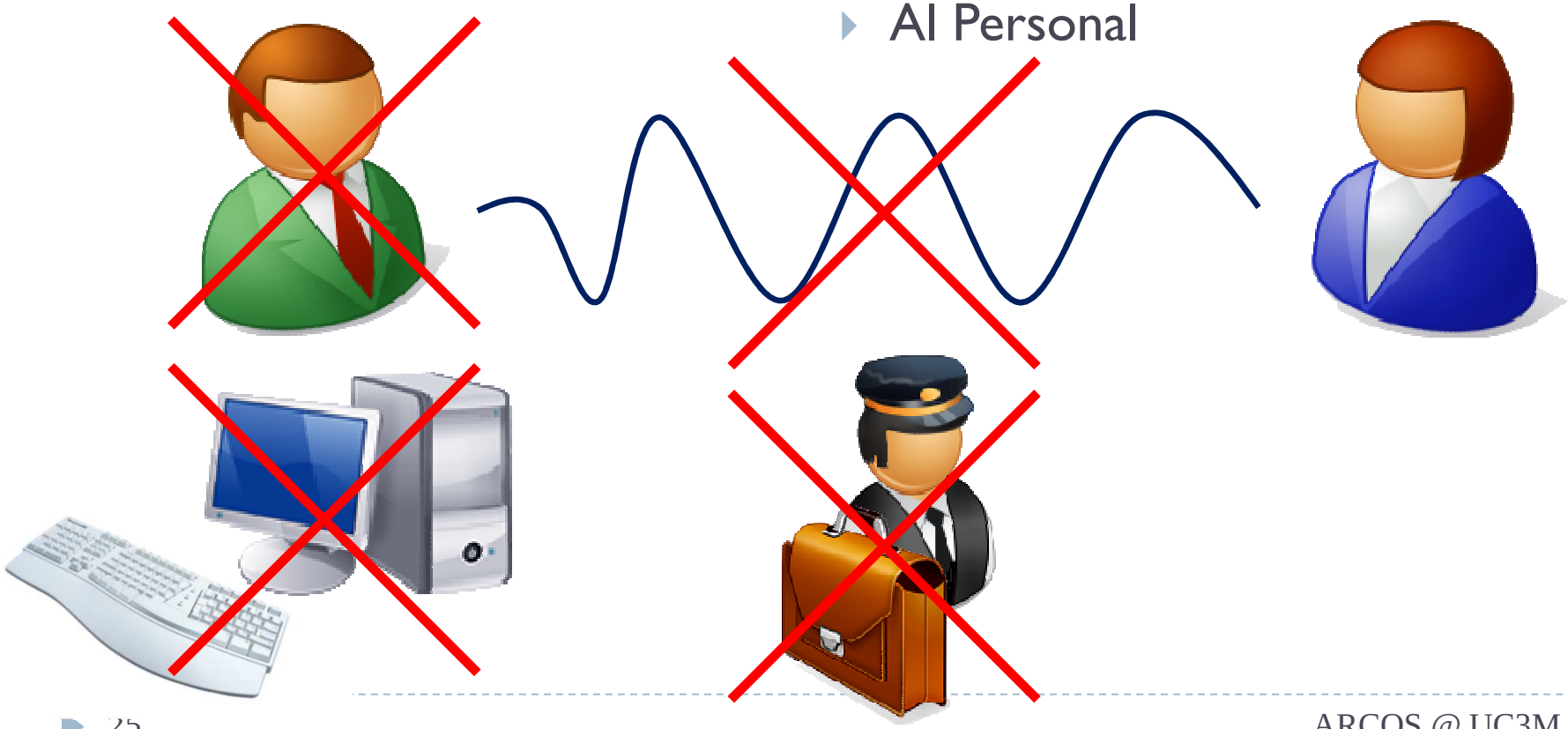
- Se destruye un elemento del sistema o se hace inasequible o inútil.
- Este es un ataque contra la **disponibilidad**.
- Ejemplos:
  - La destrucción de una pieza de hardware (como un disco duro), el corte de una línea de comunicaciones o la inutilización del sistema de gestión de archivos.



Seguridad en SS.OO.

Amenazas contra la Seguridad: Interrupción

- Puede afectar al Emisor:
  - Al Personal
  - Al Hardware
- O puede afectar al Canal de Transmisión:
  - Al Hardware
  - Al Personal



Seguridad en SS.OO.

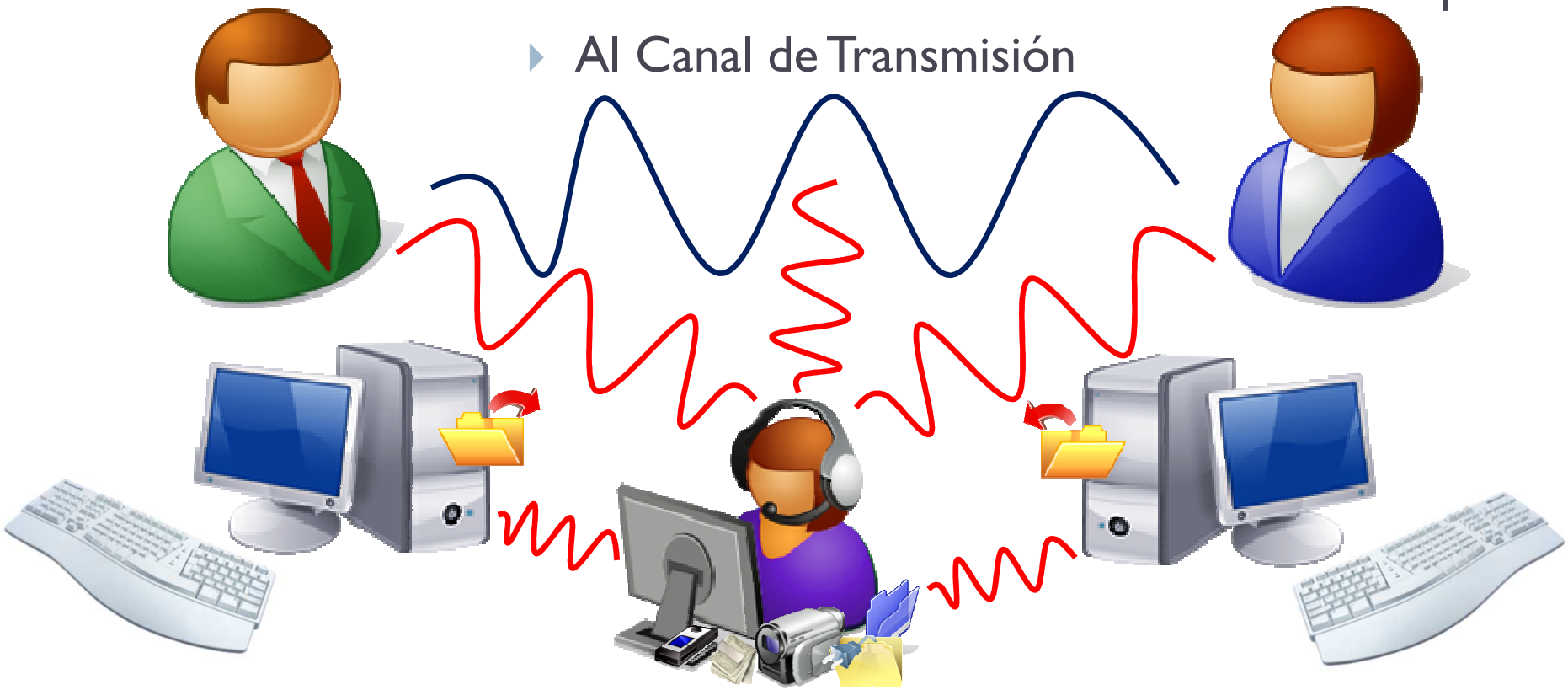
Amenazas contra la Seguridad: Intercepción

- Una parte no autorizada consigue acceder a un elemento.
- Este es un ataque contra el **confidencialidad**.
- La parte no autorizada puede ser una persona, un programa o un ordenador.
- Ejemplos:
  - La intervención de las conexiones telefónicas para capturar datos de una red y la copia ilícita de archivos o programas.

Seguridad en SS.OO.

Amenazas contra la Seguridad: Interrupción

- Puede afectar a todas las partes:
  - Y tanto al Personal como al Hardware
    - Al Emisor
    - Al Canal de Transmisión
    - Al Receptor



Seguridad en SS.OO.

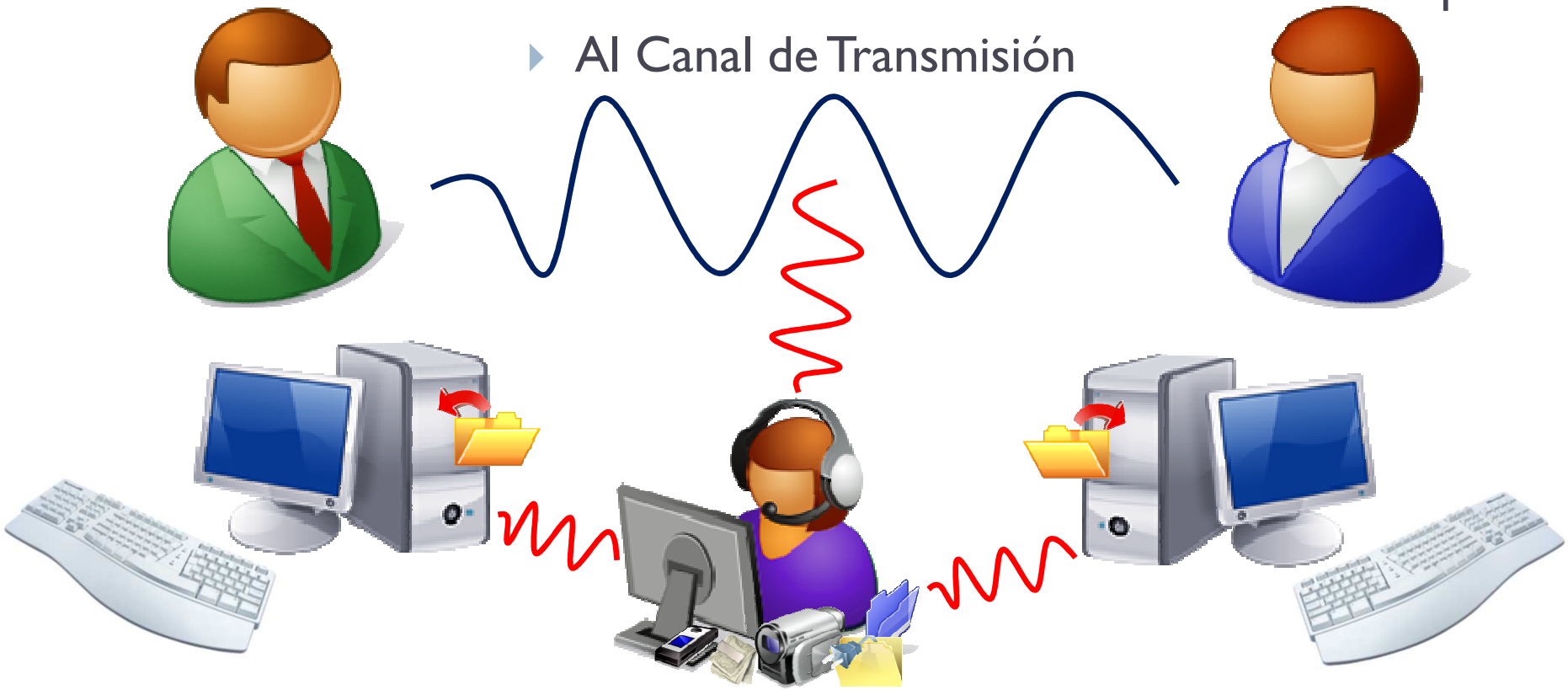
Amenazas contra la Seguridad: Modificación

- Una parte no autorizada no sólo consigue acceder, sino que falsifica un elemento.
- Este es un ataque a la **integridad**.
- Ejemplos:
  - El cambio de valores en un archivo de datos, la alteración de un programa para que se comporte de manera diferente y la modificación del contenido de los mensajes transmitidos en una red.

Seguridad en SS.OO.

Amenazas contra la Seguridad: Modificación

- Puede afectar a todas las partes:
  - Aunque sólo cuando son Hardware
  - Al Emisor
  - Al Canal de Transmisión
  - Al Receptor



Seguridad en SS.OO.

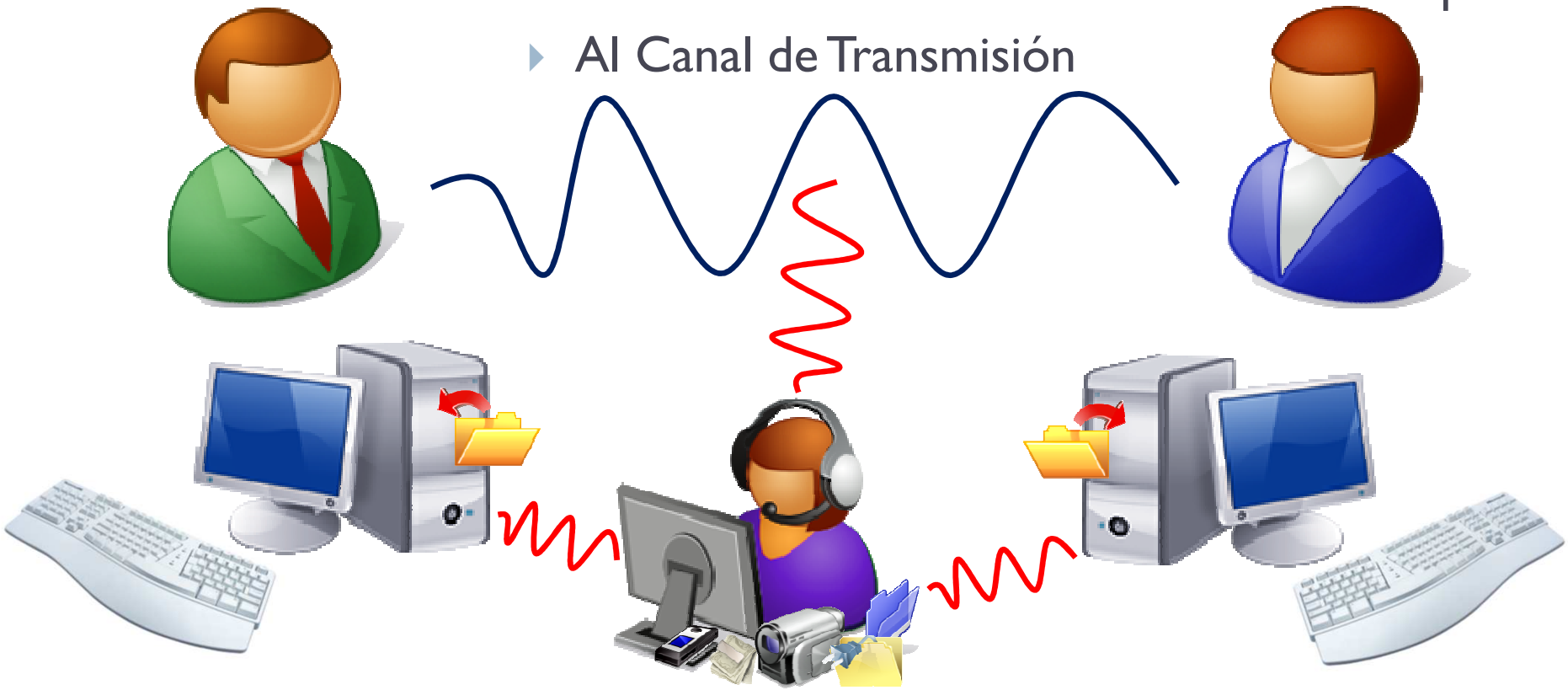
Amenazas contra la Seguridad: Invención

- Una parte no autorizada inserta objetos falsos en el sistema.
- Este es también un ataque a la **integridad.**
- Ejemplo:
  - La inserción de mensajes falsos en una red o la adición de registros a un archivo.

Seguridad en SS.OO.

Amenazas contra la Seguridad: Invención

- Puede afectar a todas las partes:
  - Aunque sólo cuando son Hardware
    - Al Emisor
    - Al Canal de Transmisión
    - Al Receptor



Seguridad en SS.OO.
Los dos principios básicos en seguridad

- Existen dos principios básicos en seguridad que afectan al diseño de un SS.OO.:
  - El principio de necesidad de saber (need-to-know):
    - Sólo debe revelarse la información a quien tenga la necesidad de su conocimiento, en el grado que tenga necesidad de su conocimiento y en función de la autorización que disponga, sólo hasta el nivel necesario para que pueda realizar su trabajo.
  - El principio de diseño intrínseco de la seguridad:
    - La seguridad debe diseñarse como parte intrínseca del sistema, desde la base del mismo y como requisito fundamental, no debe tomarse como un añadido posterior.

Seguridad en SS.OO.
Los principios de diseño de seguridad

- Todos los principios de Diseño de un SS.OO. referentes a la seguridad son los dos principales:
  - El principio de necesidad de saber (need-to-know).
  - La seguridad debe diseñarse como parte intrínseca del sistema.
- Y cinco complementarios:
  - Mínimo privilegio
  - Ahorro de mecanismos
  - Aceptación
  - Mediación total
  - Diseño abierto

Seguridad en SS.OO.

Los principios de diseño de seguridad

- Mínimo privilegio:
  - Todos los programas y usuarios del sistema deben operar utilizando el menor conjunto de privilegios necesarios para completar la labor.
  - Los derechos de acceso deben adquirirse sólo por permiso explícito.
  - El valor por omisión deberían ser "sin acceso".

Seguridad en SS.OO.

Los principios de diseño de seguridad

- Ahorro de mecanismos:
  - Los mecanismos de seguridad deben ser tan pequeños y simples como sea posible, ayudando en su verificación.
  - Esta exigencia suele suponer que deben ser una parte integral del diseño, más que mecanismos añadidos a diseños existentes.

Seguridad en SS.OO.
Los principios de diseño de seguridad

- Aceptación:
  - Los mecanismos de seguridad no deben interferir excesivamente en el trabajo de los usuarios, mientras cumplen al mismo tiempo las necesidades de aquellos que autoricen el acceso.
  - Si los mecanismos no son fáciles de usar, probablemente no van a ser usados o lo serán de forma incorrecta.

Seguridad en SS.OO.

Los principios de diseño de seguridad

- Mediación total:
  - Cada acceso debe ser cotejado con la información de control de acceso.
  - Esto incluye también a aquellos accesos que suceden fuera de la operación normal, como la recuperación y el mantenimiento.

Seguridad en SS.OO.

Los principios de diseño de seguridad

- Diseño abierto:
  - La seguridad del sistema no debe depender de guardar en secreto el diseño de sus mecanismos.
  - De esta forma, los mecanismos podrán ser revisados por muchos expertos y los usuarios podrán, por tanto, depositar una alta confianza en ellos.

Seguridad en SS.OO.
Problemas de seguridad

- Los problemas de seguridad suelen tener su origen en:
  - Fallos del soporte del S.O. (hardware).
  - Fallos intrínsecos del propio S. O. (software)
  - Usuarios inexpertos o descuidados.
  - Acceso de personas/programas a usuarios específicos para los que no están autorizados.
    - Por ejemplo, se consigue acceso a cuentas de administradores y superusuarios.
  - Uso indebido de software malicioso (*malware*):
    - Virus
    - Gusanos
    - Spyware …

Seguridad en SS.OO.

Objetivos de seguridad del diseño de SS.OO.

- Los objetivos de seguridad del diseño de un sistema operativo se basan en tres aspectos:
  - Evitar la pérdida de datos.
  - Controlar la confidencialidad de los datos.
  - Controlar el acceso a los datos y recursos.

Seguridad en SS.OO.
Ejemplos: The Rainbow Series (I)

- Se refiere a seguridad lógica (*security*)
  - Fue la primera, sigue vigente y es la más respetada de la bibliotecas de seguridad de los Sistemas Informáticos.
  - Se llama así porque cada libro de la enciclopedia tiene un color asignado por el que se le suele conocer.
  - El libro naranja está directamente relacionado con la seguridad lógica.
  - Clasifica los sistemas en cuatro niveles principales y varios niveles secundarios.
  - La clasificación es tan estricta que, para los niveles superiores se desconocen qué SS.OO. están acreditados en ellos.

Seguridad en SS.OO.

Ejemplos: The Rainbow Series (II)

- Fuente: *Sistemas Operativos. Una visión aplicada. Jesús Carretero et ál. McGraw-Hill 2001.*

A	Plan de seguridad acreditado	Ax	A1 + Desarrollo con instalaciones y personal fiables
		A1	B3 + Sistema de seguridad con verificación formal
B	Sistema de seguridad obligatorio	B3	B2 + ACL para denegar acceso + registro y auditoría de violaciones de seguridad
		B2	B1 + Protección obligatoria para todo recurso
		B1	C2 + Protección obligatoria para todo objeto de usuario
C	Capacidad discrecional de controlar accesos	C2	C1 + Control de acceso individual
		C1	Control de acceso por dominio de seguridad
D	No existen medidas de seguridad		

Seguridad en SS.OO.
Ejemplos: DO-178B

- Se refiere a salvaguarda (safety)
  - Es la normativa de software para uso en aviación civil.
  - Es la más exigente y está extendida y aceptada casi a nivel mundial.
  - Clasifica la criticidad de los sistemas informáticos:
    - En función del impacto en vidas humanas y/o destrucción total o parcial del sistema en caso de mal funcionamiento o no funcionamiento del software.
    - En cinco niveles: del A (más restrictivo) al E (menos restrictivo).
    - Por ejemplo, entre sus requisitos más estrictos está requerir una cobertura total de requisitos y de una cobertura estructural (física) del código fuente (hasta nivel C) o hasta el código máquina (para niveles B y A).

Seguridad en SS.OO.

Otras normativas de salvaguarda (safety)

- Por sectores:
  - Europa:
    - Sector Nuclear: ISO/IEC 880
    - Sector Aeroespacial: ESA - ECSS
    - Sector Médico: ISO/IEC 601-x
    - Automoción: MISRA
  - USA:
    - Sector Nuclear: NRC
    - Sector Aeroespacial: NASA Estandards
    - Sector Médico: FDA
  - Comunes:
    - Aviación Civil (Eurocontrol): DO-178B
    - Sector Militar: ¿¿??

Seguridad en SS.OO.

Los principios de diseño de protección

- Un S. O. deber ofrecer protección en el siguiente rango de posibilidades:
  - Ninguna protección.
  - Aislamiento.
  - Compartición Todo/Nada.
  - Compartición por limitación de acceso.
  - Compartición por capacidades dinámicas.
  - Uso limitado de un objeto.

Seguridad en SS.OO.

Conceptos: Mecanismo o Política (I)

- Política de seguridad:
  - Identifica el nivel de seguridad requerido en el S. O.
  - ¿Qué (nivel de) seguridad hay que tener?
- Mecanismos de seguridad:
  - Identifica los medios físicos y lógicos para llevar a cabo la política de seguridad.
  - ¿Cómo vamos a implantar el nivel de seguridad requerido por la política?

Seguridad en SS.OO.
Conceptos: Mecanismo o Política (II)

- Es una relación interdependiente y flexible:
  - Una política puede implementarse con varios mecanismos diferentes
  - Un mecanismo debe tener la posibilidad de implementar diferentes políticas

Seguridad en SS.OO.
Matriz de Protección (I)

- La Matriz de Protección es un mecanismo de seguridad:
  - Cada elemento de la Matriz determina el acceso por parte de un usuario a un objeto identificando las acciones que puede realizar sobre dicho objeto (permisos)
  - La matriz es una terna:
    - {Objeto, Usuario, Derechos}
    - Donde
      - *Objeto*: puede ser cualquier elemento HW o SW del S.O.
      - *Usuario*: puede ser cualquier ente que quiera utilizar un *Objeto*, pudiendo ser un usuario del S.O., un proceso o algún componente del propio S. O.
      - *Derechos*: son los permisos que se asignan/gestionan sobre un determinado *Objeto* y que autorizan o deniegan las operaciones sobre el mismo.

Seguridad en SS.OO.
Matriz de Protección (II)

- Para reducir la complejidad de la matriz se utiliza el concepto de Dominio:
  - Un dominio de protección es un conjunto de pares:
    - {Objeto, Derechos}
    - Cada par identifica las operaciones (=Derechos=Permisos) que se pueden ejecutar sobre un Objeto.
    - Los propios *Dominios* también son *Objetos* a proteger, ya que un determinado usuario puede cambiar de un *Dominio* a otro.
  - La matriz de protección queda reducida a la relación entre dominios y objetos.
  - La matriz de protección así definida recibe, también, el nombre de matriz de acceso.

Seguridad en SS.OO.
Matriz de Protección (III)

- Ejemplo:

Objeto / Dominio	Arch1	Arch2	Modem	Impresora	Dom1	Dom2	Dom3
Dom1	rwx	r	rw	w		Switch	
Dom2	r		rw				Switch
Dom3		rwx		w			

Seguridad en SS.OO.
Matriz de Protección (IV)

- Implementaciones de la matriz de protección/acceso:
  - Esta matriz es una matriz dispersa: poco contenido y muchas celdas vacías.
  - Para hacer posible el almacenamiento y el acceso, se utilizan dos alternativas fundamentales:
    - Almacenar por columnas: Lista de Control de Acceso.
    - Almacenar por filas: Lista (Etiquetas) de Capacidades.
  - ...Y alguna combinación de las anteriores:
    - Mixta (Clave-permisos)
    - Take-Grant (grafo dirigido)

Seguridad en SS.OO.
Matriz de Protección (V)

- Listas de Control de Acceso:
  - Su acrónimo en inglés es ACL (Access Control List).
  - Presenta una lista por cada objeto que especifica qué operaciones puede hacer cada dominio sobre ese objeto.
    - Definir una lista por objeto con los pares {dominio, derechos}.
    - El objeto contiene la lista de derechos. El usuario no tiene relación con los derechos.
  - Por ejemplo:
    - Los SS.OO. basados en UNIX utilizan el siguiente esquema de ACL para acceso a ficheros:

Objeto	owner	group	others	ACL
fichero1.txt	rwx	r	rw	{fichero1.txt, rwx r- - rw-}
fichero2.txt	r		rw	{fichero2.txt, r - - - - - rw-}
fichero3.txt		rwx		{fichero3.txt, - - - rwx - - -}

Seguridad en SS.OO.
Matriz de Protección (VI)

- Listas de Control de Acceso (ACL):
  - Ventajas:
    - Permite localizar rápidamente los derechos de cada objeto.
  - Desventajas:
    - Es difícil localizar la información de un dominio concreto sobre todo si no se limita el número de dominios.
    - Los accesos implican recorrer las listas de todos los objetos.
      - Se soluciona con una sola comprobación al inicio.
      - Esto impide cambiar los derechos al vuelo.
    - Las listas deben ser dinámicas y su mantenimiento es costoso
  - SS.OO. que las usan:
    - UNIX, Linux, Minix, Windows (NT y sucesivos)

Seguridad en SS.OO.
Matriz de Protección (VII)

- Listas (Etiquetas) de Capacidades:
  - Su acrónimo en inglés es CL (Capabilities List).
  - Presenta una lista por dominio que especifica qué operaciones se pueden hacer sobre un objeto cuando se pertenece a ese dominio.
    - Definir una lista por dominio con los pares {objeto, derechos}.
    - El dominio (usuario) contiene la lista de capacidades (derechos) que puede ejecutar sobre el objeto.

Seguridad en SS.OO.
Matriz de Protección (VIII)

- Listas (Etiquetas) de Capacidades (CL):
  - Las listas son objetos en si que los procesos acceden directamente.
  - El proceso tiene fácil acceso a las capacidades.
  - Por ejemplo:

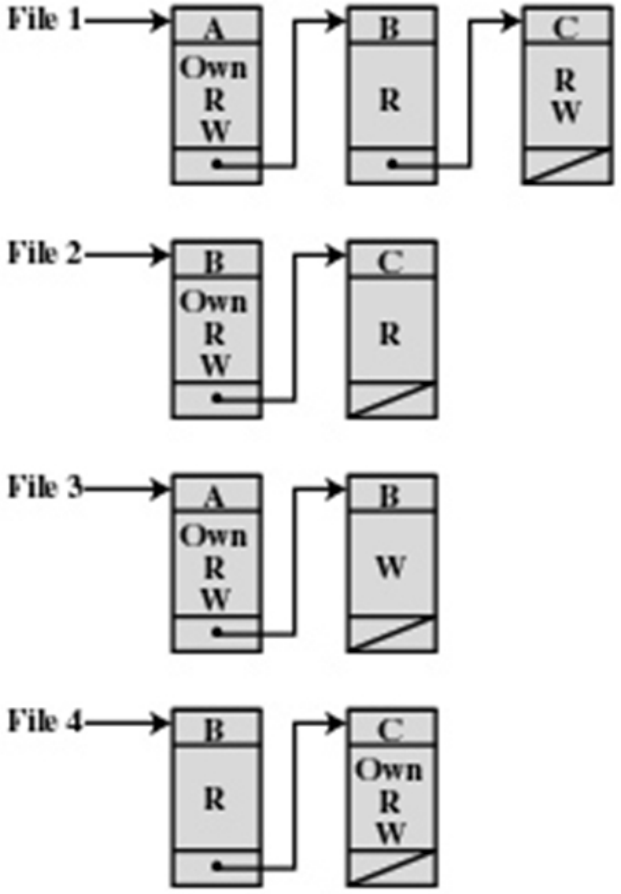
Capacidad Id.	Tipo	Derechos	Objeto	Objeto
0	Archivo	rw-	Rw	datos
1	Archivo	rwx		programa

Seguridad en SS.OO.
Matriz de Protección (IX)

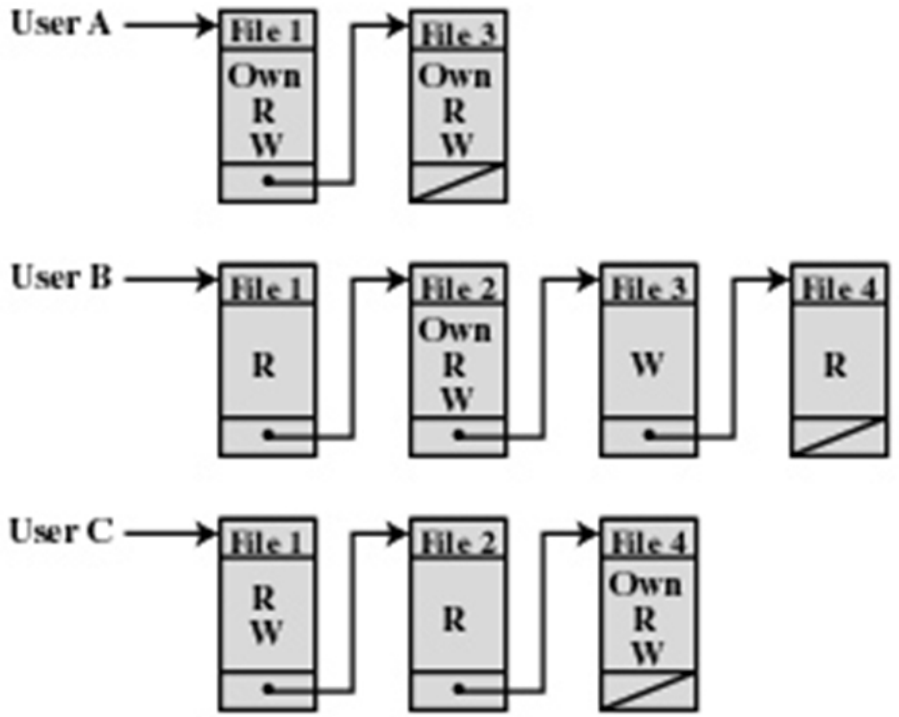
- Listas (Etiquetas) de Capacidades (CL):
  - Ventajas:
    - Se pueden realizar consultas más frecuentes sin perder rendimiento.
    - Así se pueden hacer modificaciones efectivas al vuelo.
  - Desventajas:
    - Problemas de seguridad al modificar el proceso su lista.
    - Revocación de derechos para un objeto, hay que ver todas las listas.
  - SS.OO. que las usan:
    - VMS y OS 360

Seguridad en SS.OO.
Matriz de Protección (X)

Listas de Control de Acceso (ACL)



Listas (Etiquetas) de Capacidades (CL)



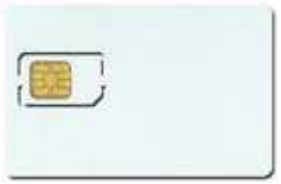
Fuente de las gráficas: *Sistemas Operativos. William Stallings*

Seguridad
Autentificación de usuarios

- Procesos de autentificación:
  - Pedir información que solo conoce el usuario (contraseña)

  - Determinar características físicas del usuario (biometría).

  - Pedir un objeto que posee el usuario (tarjetas)



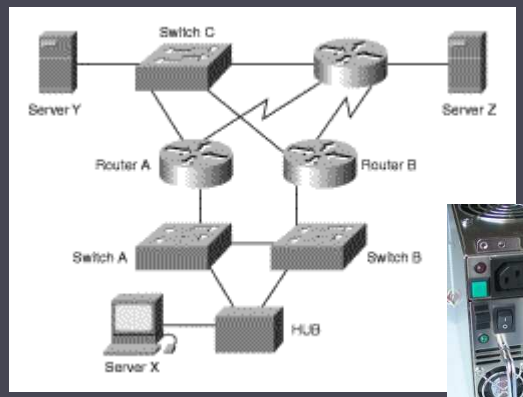
  - Combinación de las anteriores

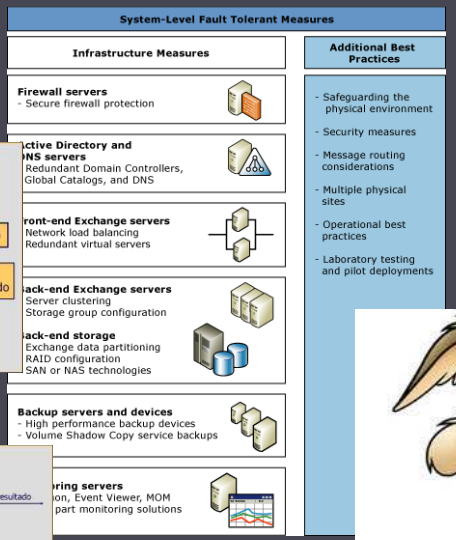
Seguridad
Autentificación de usuarios

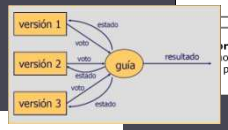
- Palabras clave o contraseñas:
  - Asignación de claves (usuario o computadora).
  - Longitud y formato de las claves.
  - Almacenamiento de las claves.
  - Duración de las claves.

Sistemas Operativos Tolerantes a Fallos

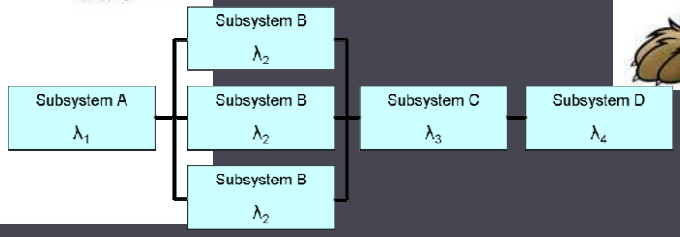
Diseño de Sistemas Operativos. Aspectos Avanzados







$$\mathrm{MTBF} = \int_0^\infty t f(t)\, dt$$



SS.OO. Tolerantes a fallos.
Conceptos (I).

- Definiciones:
  - La tolerancia a fallos es la capacidad de un sistema de conservar la máxima capacidad operativa y los máximos datos posibles en caso de fallo de alguno de sus elementos.
  - La capacidad introducida (*built-in capability*) en un sistema para proporcionar una ejecución correcta continuada en presencia de un número limitado de faltas/defectos (faults) del HW o del SW.
- El objetivo es mantener el sistema operativo de forma consistente a pesar de los fallos que puedan existir.

Tolerancia a fallos.

Sistemas típicos tolerantes a fallos

- Sistemas con tolerancia a fallos suelen ser sistemas
  - Con una larga vida útil prevista
  - Sistemas de difícil mantenimiento
  - Sistemas de alta disponibilidad
  - Aplicaciones críticas
- Por ejemplo:
  - Sistemas bancarios, aviónica, telemedicina, satélites, cohetes, sistemas militares, sistemas nucleares, etc.

SS.OO. Tolerantes a fallos.
Conceptos (II).

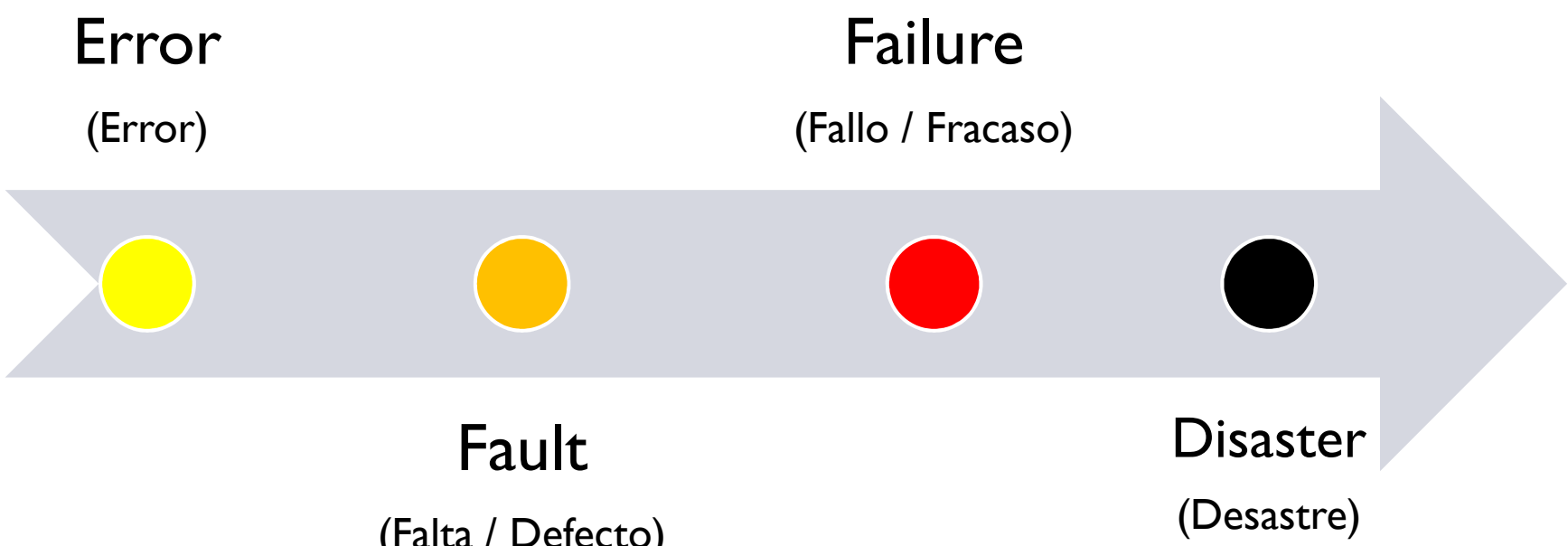
- Comportamiento anómalo:
  - Comportamiento del sistema que es inconsistente con los requisitos especificados.
- Contingencia, Azar o Peligro (*Hazard*):
  - Conjunto de condiciones de un sistema (estado) que junto con condiciones del entorno pueden conducir a un accidente.
- Error (*Error*):
  - Con respecto al software, es una equivocación o malentendido en los requisitos, el diseño y/o el código.

SS.OO. Tolerantes a fallos.
Conceptos (III).

- Fallo o Fracaso (*Failure*)
  - La inhabilidad del sistema (o componente) para llevar a cabo la función requerida dentro de los límites especificados.
- Avería, Falta o Defecto (*Fault*)
  - Manifestación de un error proveniente del software.
    - Por manifestación se entiende que es externa, visible o se hace presente de alguna manera.
    - Es una definición extensible al hardware y al sistema.
- Relación entre Fallo (*Failure*) y Falta (*Fault*)
  - Un fallo o fracaso se puede producir cuando se encuentra un defecto o falta.

SS.OO. Tolerantes a fallos.

Conceptos (IV).



SS.OO. Tolerantes a fallos.
Conceptos (V).

- Fiabilidad (*Reliability*):
  - Probabilidad de que un sistema desarrolle correctamente las funciones que se le requieren durante un intervalo de tiempo.
- Disponibilidad (*Availability*):
  - Probabilidad de que un sistema esté disponible para desarrollar la función que se le requiere en un instante dado.
- Mantenibilidad (*Maintainability*):
  - Capacidad de ser mantenido de un Sistema.
- Confiabilidad (*Dependability*):
  - *Dependability* = RAM + *Safety ¿+ Integrity?*

SS.OO. Tolerantes a fallos.
Conceptos (VI).

- La estabilidad:
  - Es un aspecto importante de la tolerancia a fallos directamente relacionado con los sistemas de tiempo real.
  - Se dice que es estable si, en los casos en los que es imposible cumplir todos los plazos de ejecución de las tareas, el sistema cumplirá los plazos de las tareas más críticas y de mayor prioridad, incluso si no se cumplen los de alguna tarea menos crítica.

Tolerancia a fallos.

Prevención de fallos vs Tolerancia a fallos

- Prevención de fallos (*Fault avoidance*)
  - Evitar que se introduzcan fallos en el sistema antes que entre en funcionamiento.
  - Se utilizan en la fase de desarrollo del sistema.
    - Evitar fallos.
    - Eliminar fallos.
- Tolerancia a fallos (*Fault tolerance*)
  - Conseguir que el sistema continúe funcionando aunque se produzcan fallos.
  - Se utilizan en la etapa de funcionamiento del sistema.
  - Es necesario saber los posibles tipos de fallos, es decir, anticiparse a los fallos.

Tolerancia a fallos.

Determinación de prevención y tolerancia a fallos

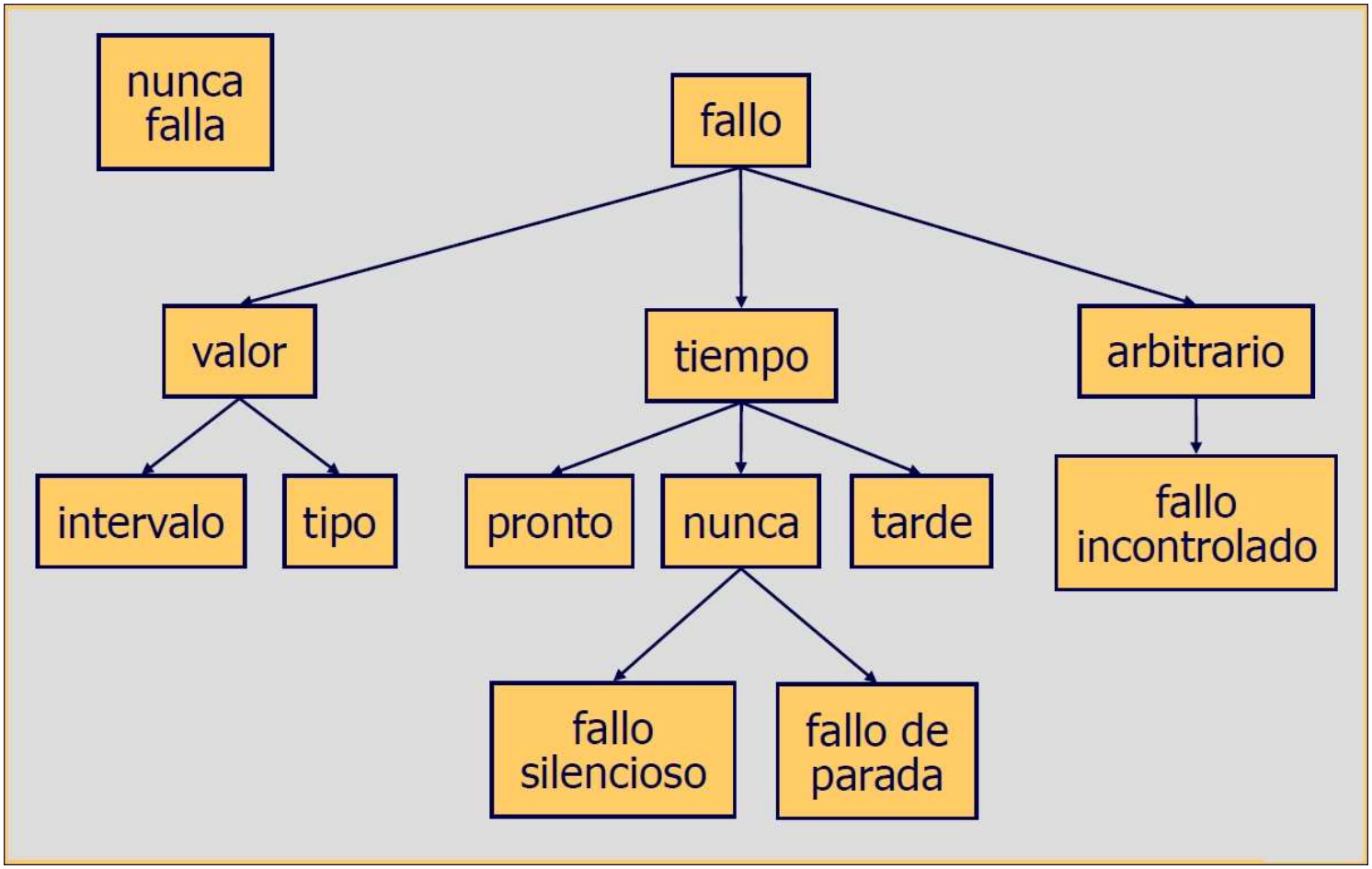
- *Hazard Analysis* (HA):
  - Análisis de posibles puntos de fallo del sistema.
- *Fault Tree Analysis* (FTA):
  - Establecimiento de puntos de fallo simples y descomposición del árbol de fallos.
- *Fault Modes and Effect Critical Analysis* (FMECA):
  - Determinación de los modos y submodos de funcionamiento degradado en función de los potenciales fallos y análisis de su criticidad.
  - Establecimiento del *Mean Time Between Failures* (MTBF) desde elementos hasta la totalidad del sistema.

$$\mathrm{MTBF} = \int_0^\infty t f(t)\, dt$$

- *Life Cycle Cost Analysis* (LCCA):
  - Análisis del coste del Ciclo de Vida del Sistema en función de su mantenibilidad y los puntos anteriores.

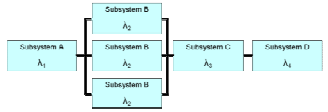
Tolerancia a fallos.

Análisis de Modos de Fallo



Fuente de la gráfica: *Ingeniería Informática. Universidad de Vigo*

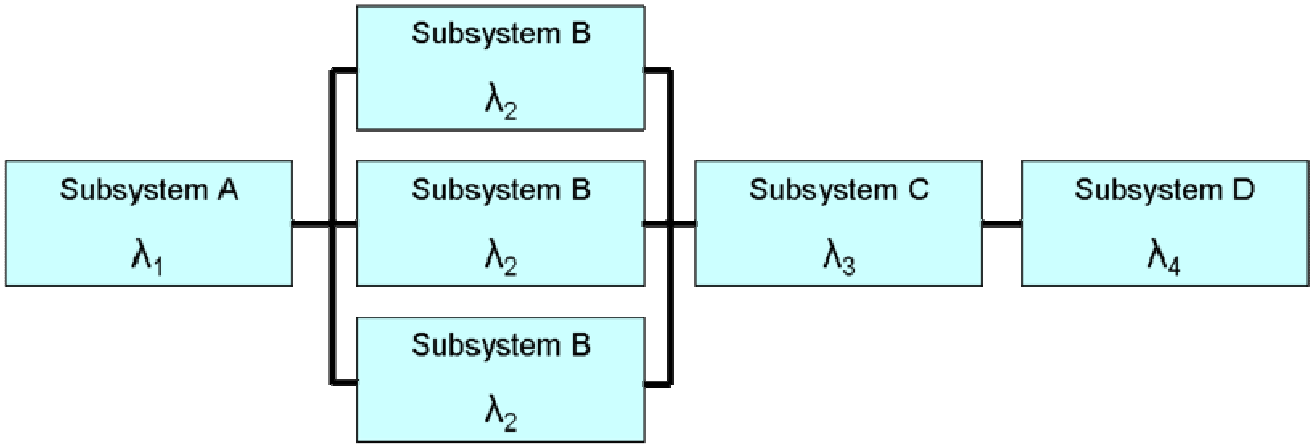
Tolerancia a fallos.



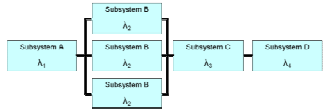
Técnica de tolerancia a fallos: Redundancia (I).

- Definición: disposición de una o más medidas adicionales, normalmente idénticas, para proporcionar tolerancia a fallos.
  - Fuente: Serie UNE-EN-5012

- Se utilizan componentes adicionales para detectar los fallos, enmascararlos y recuperar el comportamiento correcto del sistema.



Tolerancia a fallos.

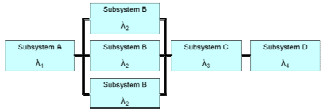


Técnica de tolerancia a fallos: Redundancia (II).

- Ventajas
  - Es la técnica más simple, extendida y probada.
  - Se aplica igualmente a los niveles de Sistema, Software y Hardware.
  - La redundancia del Software en Sistemas no críticos se puede reducir a tener copias del mismo Software sobre los elementos redundantes Hardware.

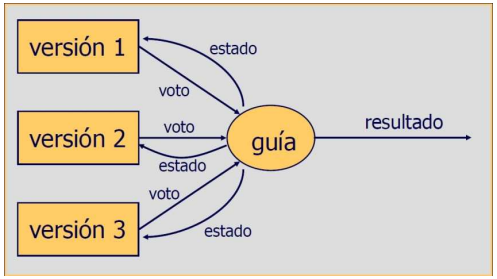
- Desventajas
  - El empleo de redundancia aumenta la complejidad del sistema y puede introducir fallos adicionales si no se gestiona de forma correcta.
  - Es difícil incluir redundancia del Software, a menos que se utilicen soportes HW también redundantes.

Tolerancia a fallos.

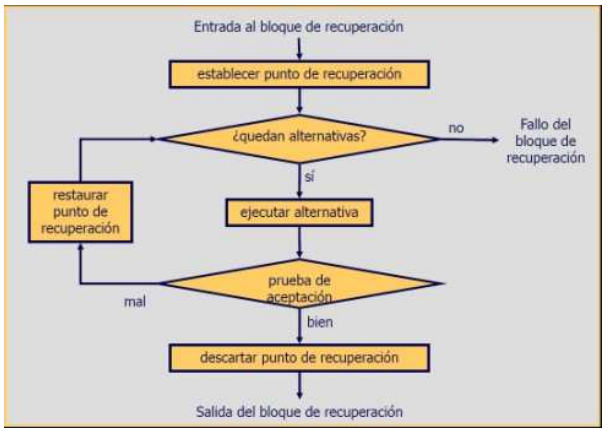


Técnica de tolerancia a fallos: Redundancia SW.

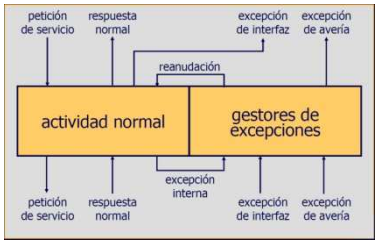
- Redundancia estática:
  - Programación de Múltiples versiones del Software



- Redundancia dinámica:
  - Bloques de recuperación
    - *Fault Detection, Isolation and Recovery (FDIR) technique*



- Gestión de Excepciones



Fuente de las gráficas: *Ingeniería Informática. Universidad de Vigo*

Tolerancia a fallos.

Algunos datos experimentales: Linux (I).

- Se experimenta inyectando errores en el Kernel de Linux directamente lo que permite conocer las razones internas de las caídas.

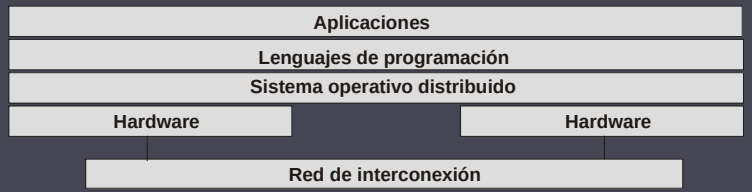
  - Fuentes:
    - Original:
      - W. Gu, Z. Kalbarczyk, R.K. Iyer, Z. Yang, "Characterization of Linux Kernel Behavior under Error", en Dependable Systems and Networks, 2003. (http://courses.ece.uiuc.edu/ece442/Notes_03/lecture_16_spr2003.pdf).
    - Resumido y presentado por:
      - José Antonio Gómez Hernández (Universidad de Granada)

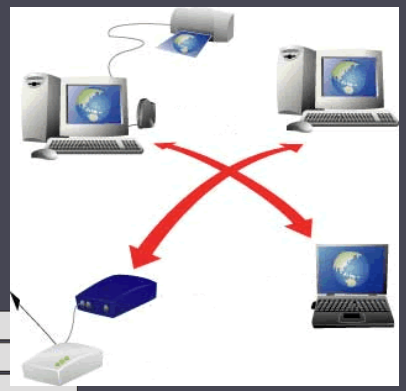
Tolerancia a fallos.

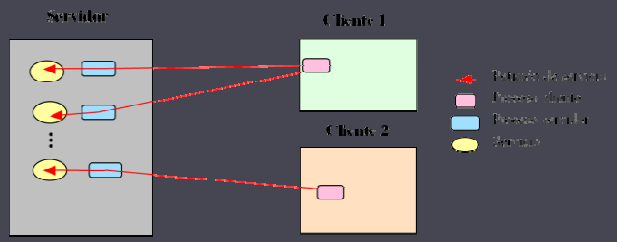
Algunos datos experimentales: Linux (II).

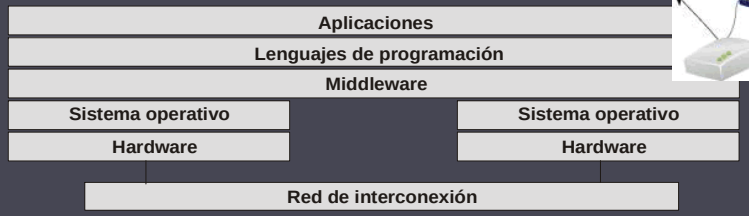
- 95% de las caídas del kernel de Linux se deben a:
  - Incapacidad para manejar un puntero kernel NULL.
  - Incapaz de manejar solicitudes de páginas kernel.
  - Código de operación inválido.
  - Falta de protección general
- Menos del 10% de las caídas están asociadas con propagación de faltas.
- El 40% de las latencias de caída están dentro de 100 ciclos (el 20% por encima de los100.000 ciclos.
- Nueve errores del kernel provocaron caídas que necesitaron reformatear el sistema de archivos; restablecer el sistema llevó sobre una hora.

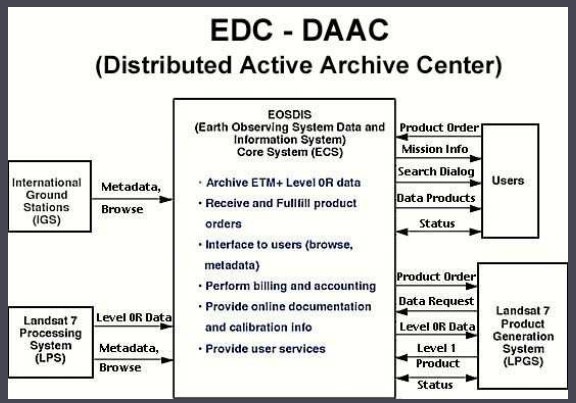
Diseño de Sistemas Operativos. Aspectos Avanzados

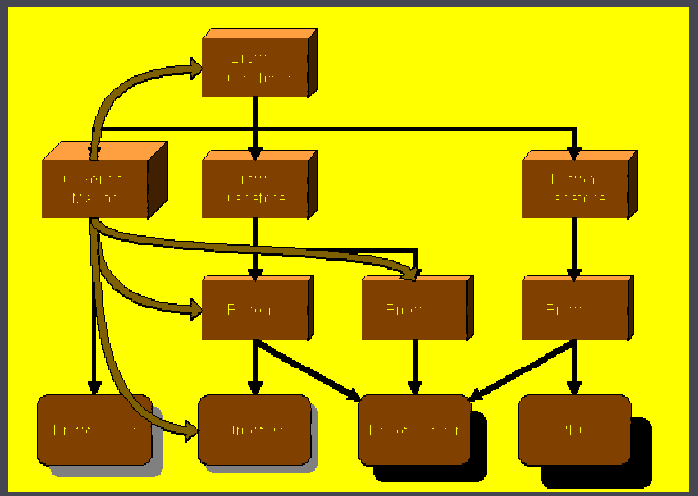












SS.OO. Distribuidos.

Conceptos

- Definición general y formal:
  - Sistema compuesto por recursos de computación físicamente separados (*distribuidos*) y que se encuentran conectados a través de una red de comunicaciones.

- Otra definición menos general y formal:
  - Un sistema distribuido es aquél en el que no puedes trabajar con tu máquina por el fallo de otra máquina que ni siquiera sabías que existía (Leslie Lamport)

SS.OO. Distribuidos.

Conceptos

- Definición desde un punto de vista **físico**:
  - Es un conjunto de procesadores, posiblemente heterogéneos, sin memoria ni reloj común y que se encuentran conectados a través de una red de interconexión.
- Definición desde un punto de vista **lógico**:
  - Es un conjunto de procesos que ejecutan en una o más computadoras, que colaboran y se comunican entre ellos mediante el intercambio de mensajes.

SS.OO. Distribuidos.
Características

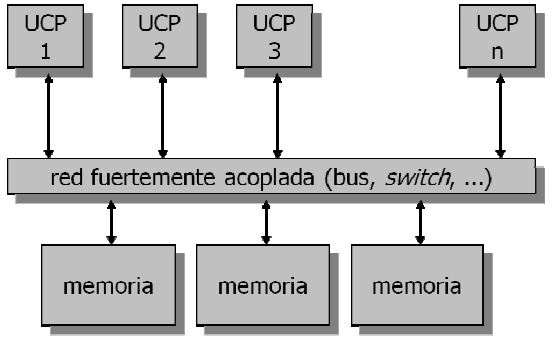
- Los SS.OO. Distribuidos presentan las siguientes características positivas (ventajas):
  - *Compartición de recursos*: HW, SW y Datos.
  - *Escalabilidad*: tiene capacidad de crecimiento a base de añadir más HW y el SW necesario para gestionarlo.
  - *Alto rendimiento*: debido al uso de múltiples procesadores.
  - *Fiabilidad y disponibilidad*: debido al alto número de procesadores y a la replicación consistente de la información.
  - *Buena relación coste/rendimiento*: debido al uso de muchos procesadores de bajo coste individual en lugar de usar sistemas concentrados de alto coste de adquisición y mantenimiento.
  - *Otros*: concurrencia, velocidad, …

SS.OO. Distribuidos.
Características

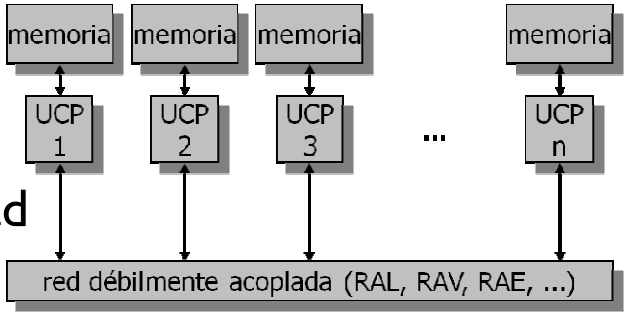
- Los SS.OO. Distribuidos presentan las siguientes características negativas (desventajas):
  - Alta complejidad y dependencia en la interconexión:
    - Mayor coste de las comunicaciones.
    - Mecanismos específicos para aumentar la fiabilidad de las comunicaciones frente a pérdida de mensajes.
    - Potencial saturación de las comunicaciones.
  - Menor grado de seguridad (*security*) en las comunicaciones.
  - Software de gestión del S. O. más complejo.
  - Balanceo de la carga de trabajo y almacenamiento que puede incurrir en:
    - Una carga de trabajo en cada nodo no adecuada a su potencia.
    - Una inconsistencia en las réplicas de la información.

SS.OO. Distribuidos o SS.OO. Paralelos

- SS.OO. Distribuidos
  - Objetivo:
    - Compartir recursos de manera colaborativa.
  - Entornos de trabajo:
    - Redes de ordenadores



- SS.OO. Paralelos
  - Objetivo:
    - Alto rendimiento (*speed-up*) y alta productividad
  - Entornos de trabajo:
    - *Máquinas paralelas en arquitecturas dedicadas*: con multiprocesadores o multicomputadores.
    - *Clusters*: agrupaciones de estaciones de trabajo en red trabajando como un multicomputador.



SS.OO. Distribuidos.
Protocolos de comunicación

- Los SS.OO. Distribuidos dependen de forma clave de los protocolos de comunicación:
  - Un protocolo de comunicación define un conjunto de reglas e instrucciones que gobiernan el intercambio de mensajes entre procesos.
  - Los protocolos se estructuran en pilas de protocolos por niveles:
    - El protocolo más extendido es el OSI de ISO.
    - Para los niveles inferiores de dicha torre, se suele usar TCP/IP.
  - Sobre estos protocolos se montan los protocolos de comunicación entre procesos

SS.OO. Distribuidos.

Protocolos de comunicación: la torre OSI

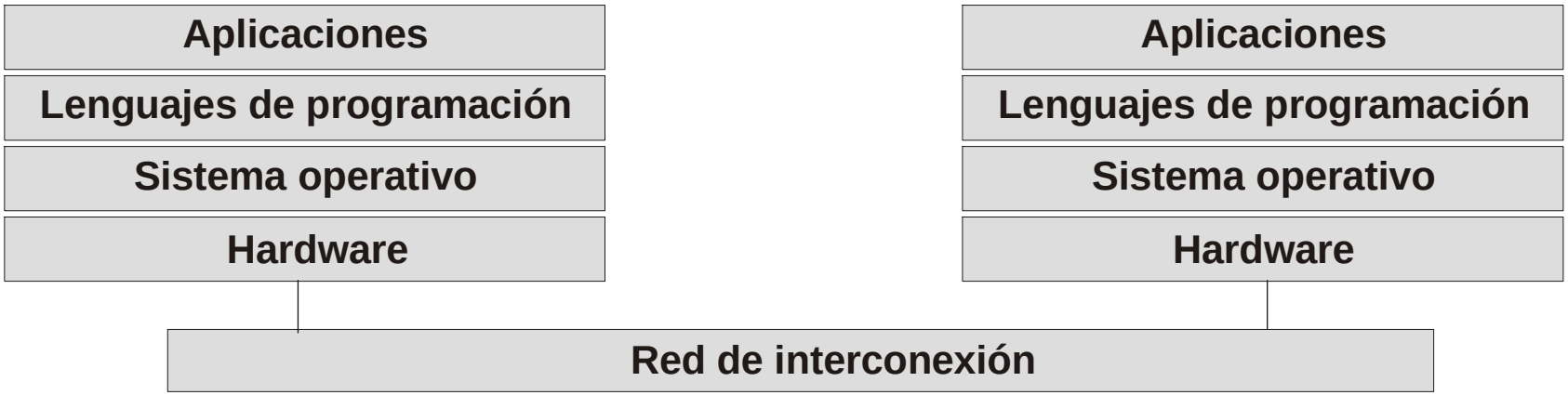
Nivel	Definición
1 Físico	Se ocupa de la transmisión de un flujo no estructurado de bits por el enlace físico; incluye parámetros tales como el nivel de voltaje de la señal y la duración de los bits; trata con características mecánicas, eléctricas y procesales para establecer, mantener y desactivar el enlace tísico.
2 Enlace de Datos	Ofrece una transferencia fiable de datos a través del enlace tísico; envía bloques de datos (tramas) con la sincronización, el control de errores y de flujo necesarios.
3 Red	Ofrece a los niveles superiores independencia de las tecnologías de transmisión de datos y conmutación empleadas para conectar los sistemas; responsable de establecer, mantener y finalizar las conexiones.
4 Transporte	Ofrece transferencia fiable y transparente de datos entre los puntos finales; ofrece recuperación de errores y control de flujo.
5 Sesión	Proporciona la estructura de control para la comunicación entre las aplicaciones; establece, gestiona y finaliza las conexiones (sesiones) entre las aplicaciones cooperativas.
6 Presentación	Lleva a cabo transformaciones de datos para proporcionar una interfaz estándar de aplicaciones y ofrecer servicios de comunicación comunes; por ejemplo: cifrado, compresión de textos y reformateado.
7 Aplicación	Ofrece servicios a los usuarios del entorno OSI; por ejemplo: servidor de transacciones, servicio de transferencia de archivos, gestión de la red.

SS.OO. Distribuidos.
Protocolos de comunicación entre procesos

- Existen dos enfoques diferentes para el paso de mensajes:
  - Mecanismos de bajo nivel
    - El programador debe encargarse de establecer los protocolos y de gestionarlos.
    - Como son los mensajes POSIX, los Mailslots de Win32 y los Sockets.
  - Mecanismos de alto nivel
    - Existen abstracciones que facilitan la programación y aíslan al programador de los aspectos de bajo nivel.
    - Como son las RPC (llamadas a procedimientos remotos), RMI de Java o la invocación a métodos remotos de CORBA.

SS.OO. Distribuidos

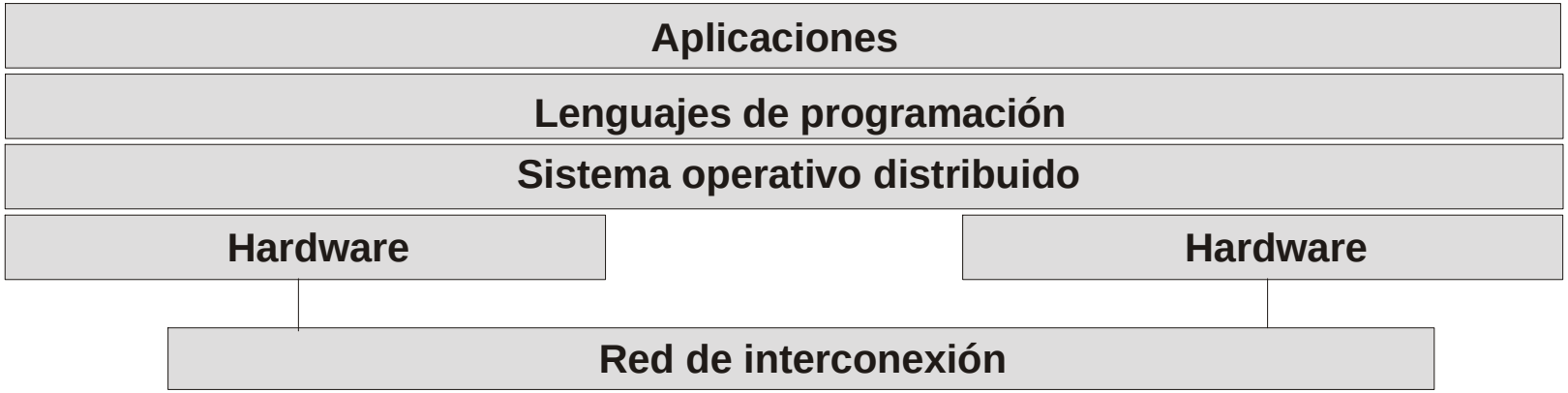
Sistema operativo en red



- El usuario ve un conjunto de máquinas independientes
  - No hay transparencia
- Se debe acceder de forma explícita a los recursos de otras máquinas
- Difíciles de utilizar para desarrollar aplicaciones distribuidas

SS.OO. Distribuidos

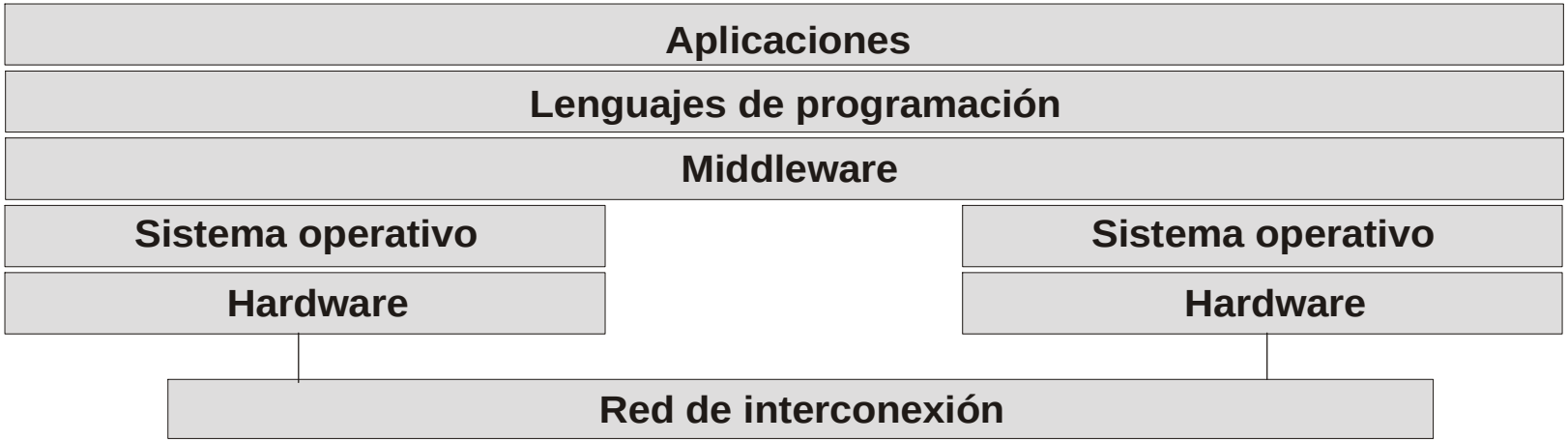
Sistema operativo distribuido



- Se comporta como un SO único
  - Hace creer a los usuarios que trabajan con un único sistema centralizado
  - Distribución. Transparencia
- Se construyen normalmente como micronúcleos que ofrecen servicios básicos de comunicación (Mach, Amoeba, Chorus).
- Todos los computadores deben ejecutar el mismo SOD

SS.OO. Distribuidos

Middleware y entornos distribuidos



- Servicios y protocolos estandarizados: Sistemas abiertos
- Ofrecen servicios no incluidos en el SO (servicios de ficheros distribuidos, servicios de nombres, ...)
- Facilitan el desarrollo de aplicaciones distribuidas
- Independientes del HW y del SO subyacente.
- DCE, CORBA, DCOM, WebOS, Globus, .NET

SS.OO. Distribuidos

Paradigmas de computación distribuida

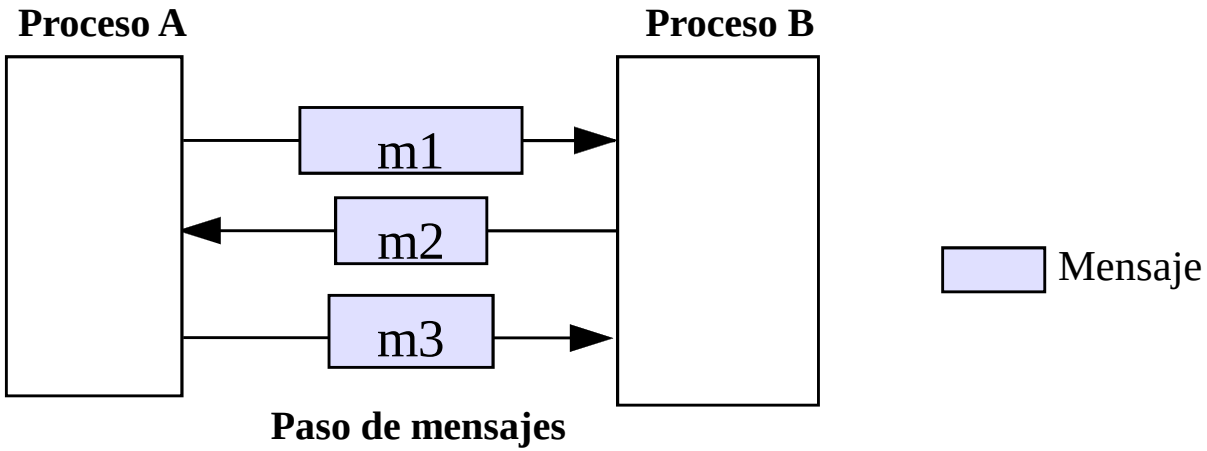
- Paso de mensajes
- Llamadas a procedimientos remotos
- Invocación de métodos remotos

- Cliente-servidor
- Peer-to-peer

SS.OO. Distribuidos

Paso de mensajes

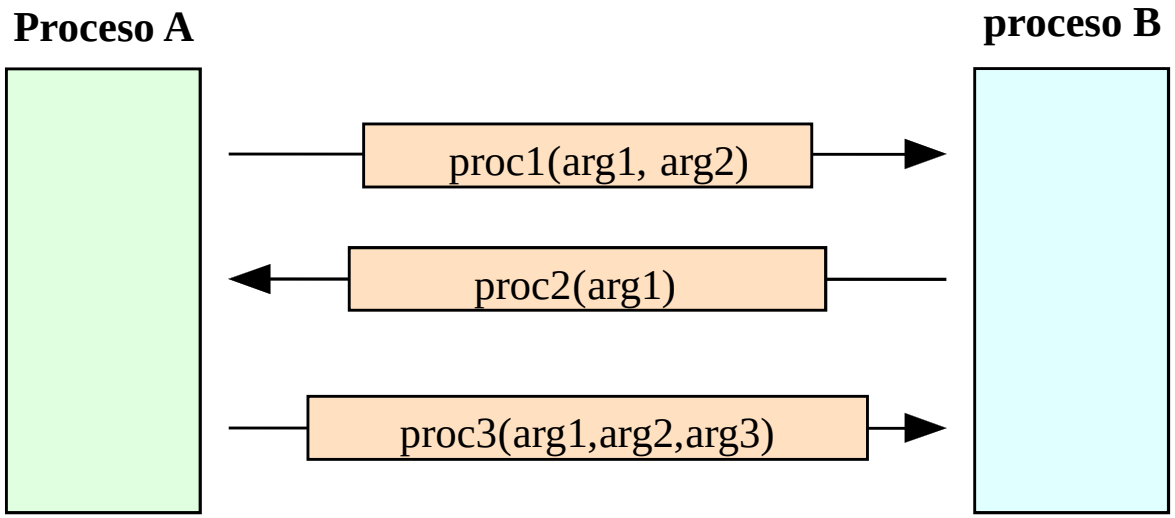
- Paradigma fundamental para aplicaciones distribuidas.
- Un proceso envía un mensaje de solicitud.
- El mensaje llega al receptor, el cual procesa la solicitud y devuelve un mensaje en respuesta.
- Esta respuesta puede originar posteriores solicitudes por parte del emisor.



SS.OO. Distribuidos

Llamadas a procedimientos remotos

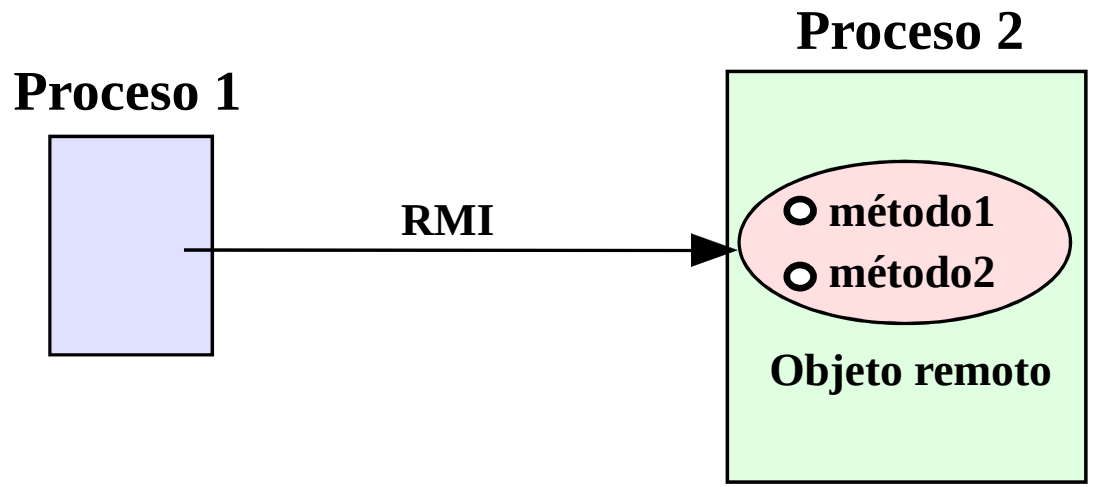
- Objetivo: hacer que el software distribuido se programe igual que una aplicación no distribuida.
- *Conceptualmente* igual que la invocación de un procedimiento local.



SS.OO. Distribuidos

Invocación de métodos remotos

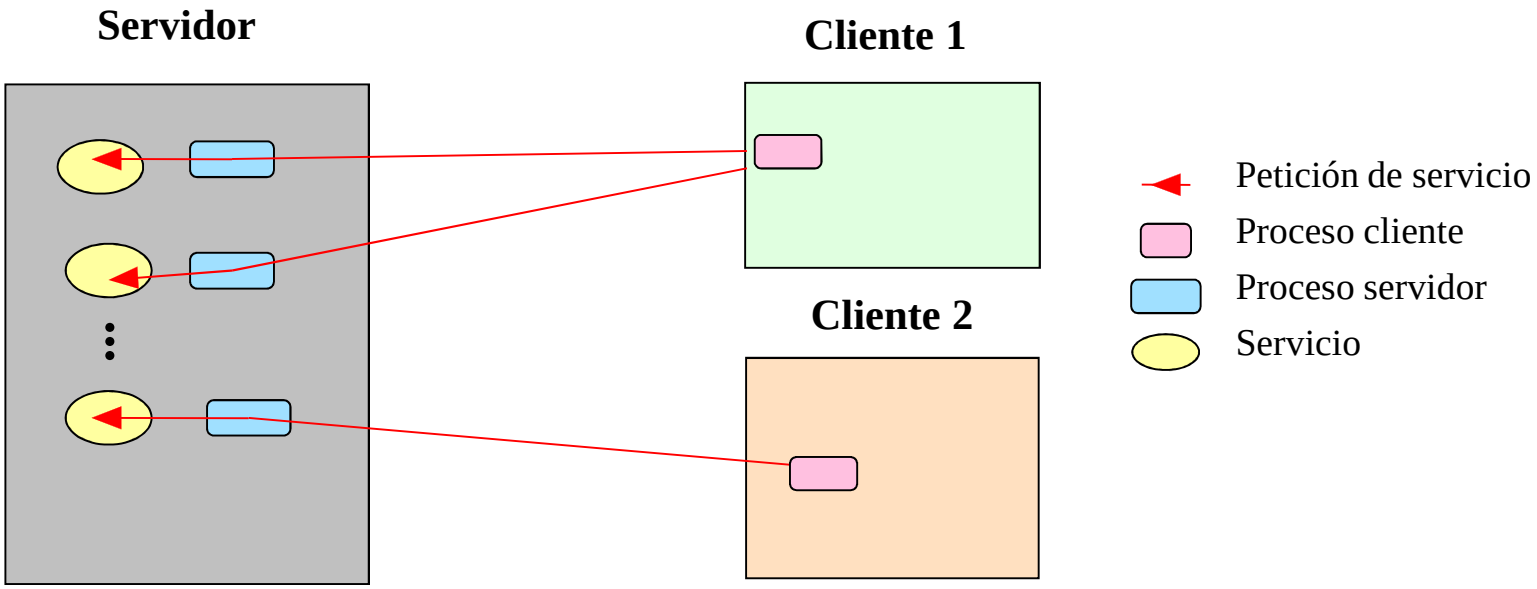
- Modelo equivalente a las llamadas a procedimientos remotos.
- Proceso invoca un método local de otro proceso.
- Ejemplos: CORBA, RMI de Java, *Microsoft* COM, DCOM, *Java Beans*, .NET Remoting



SS.OO. Distribuidos

Cliente-Servidor

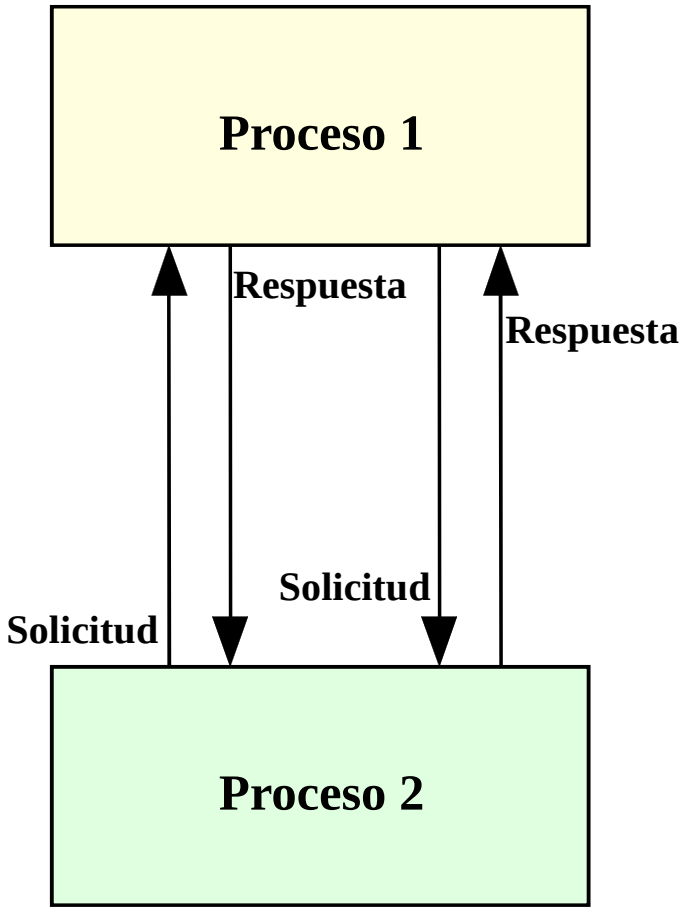
- Servidor
  - Ofrece servicio
  - Elemento pasivo: espera la llegada de peticiones
- Cliente
  - Solicita el servicio. Elemento activo



SS.OO. Distribuidos

Peer-to-Peer

- Los procesos participantes tienen el mismo papel:
  - Cliente y servidor
- Los recursos y los servicios son intercambiados entre los computadores.
- Ejemplos:
  - Napster → intercambio de ficheros.
  - Gnutella
  - …



SS.OO. Distribuidos.
Enfoque actual y futuro

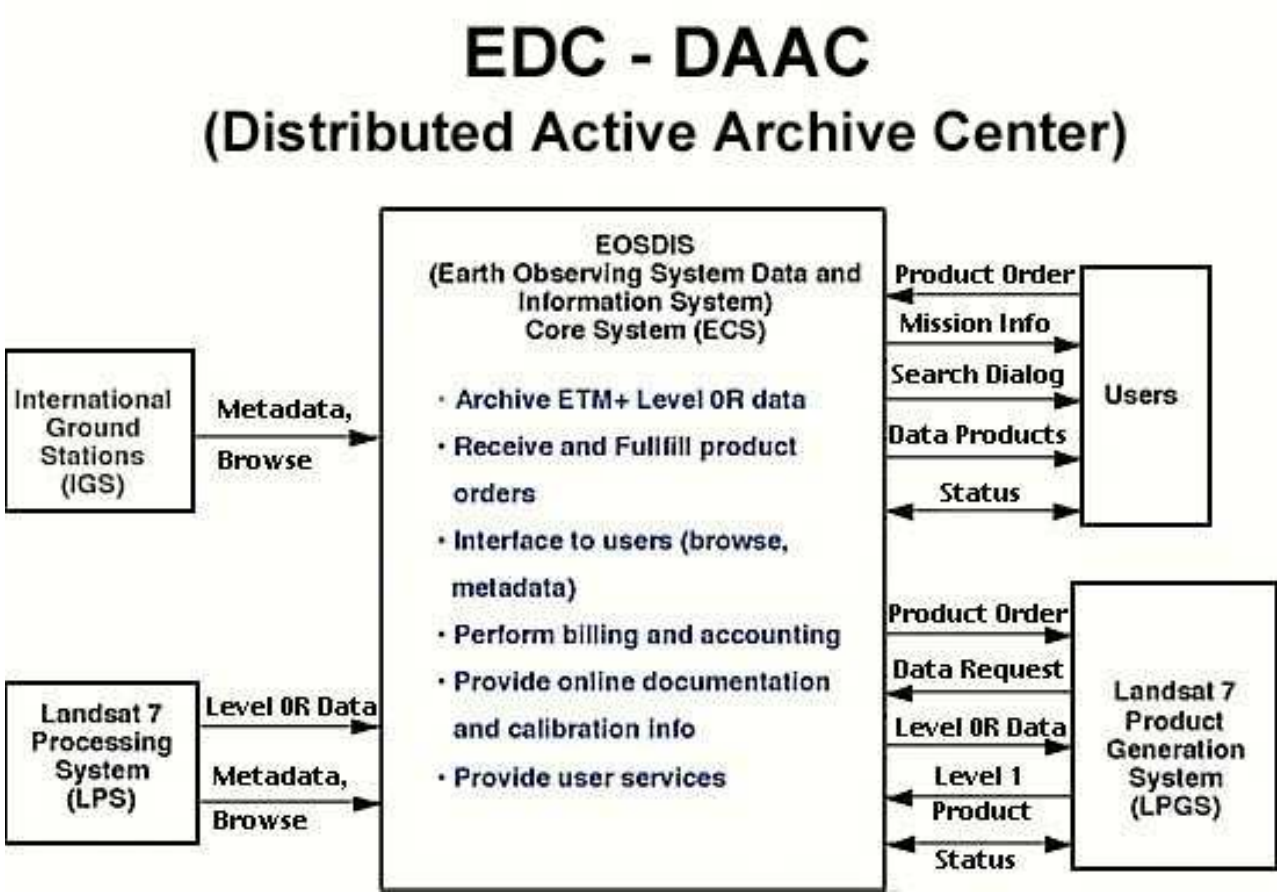
- Actualmente existen dos tendencias muy fuertes y diferenciadas en el proceso distribuido:
  - Middelware y Data Distribution System (DDS)
  - The Cloud (la Nube)
  - Grid Computing
  - Sistemas MultiAgentes (MAS)
- Todos son enfoques complementarios y que actúan sobre diferentes áreas de aplicación.

SS.OO. Distribuidos.
Enfoque actual y futuro

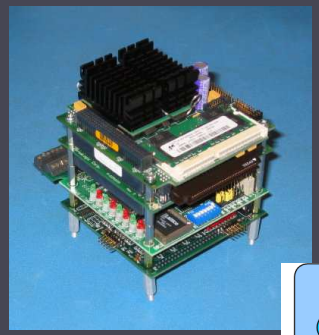
- Middelware y Data Distribution System (DDS):
  - Utilizan mecanismos de Publicador / Suscriptor basados en la filosofía de Cliente/Servidor:
  - Se están utilizando en entornos industriales para el control de redes de sensores y actuadores, como por ejemplo en automoción.
  - También están teniendo un amplio uso en entornos militares (de donde proviene) y espaciales (ver siguiente transparencia)
  - Plantea problemas significativos de salvaguarda (safety) más por los entornos en los que se usa que por la propia tecnología.

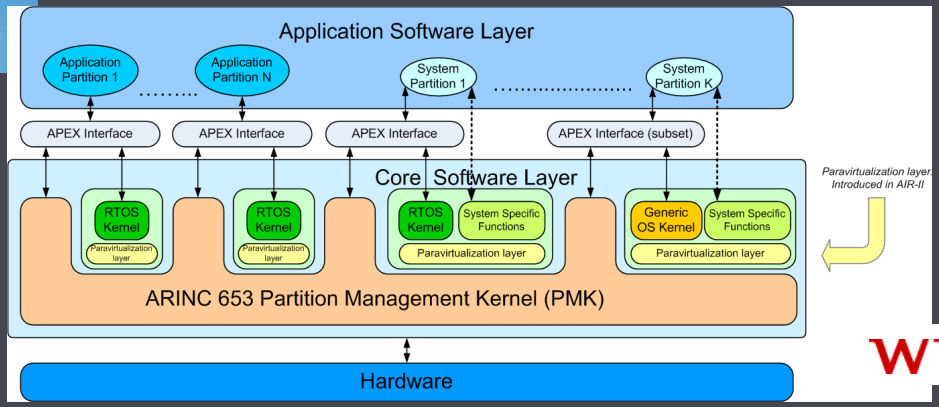
SS.OO. Distribuidos.
Enfoque actual y futuro. Ejemplo DDS

- Landsat 7. Science Data Users Handbook.
- http://landsathandbook.gsfc.nasa.gov/program/prog_sect1_3.html

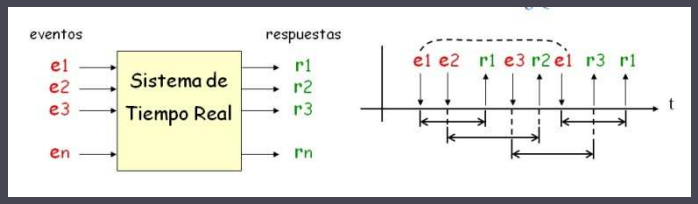


Diseño de Sistemas Operativos. Aspectos Avanzados





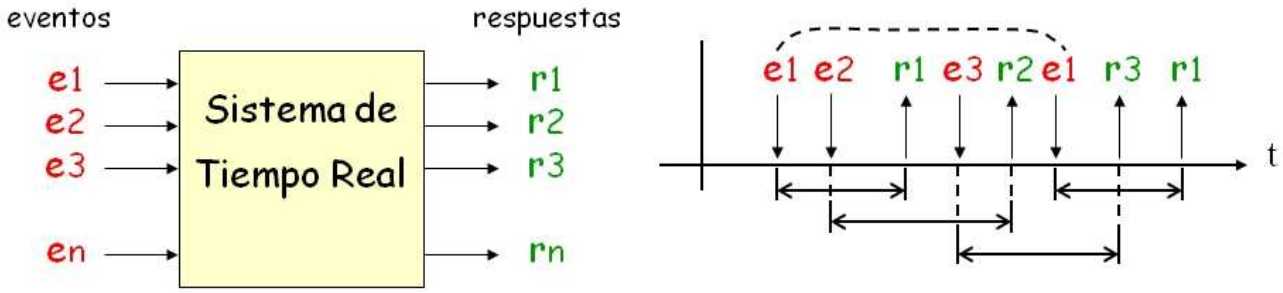
WIND RIVER



ARINC

SS.OO. en Tiempo Real.
Conceptos

- Definición de Procesamiento en Tiempo Real:
  - Tipo de procesamiento en el que la exactitud del sistema no depende sólo del resultado lógico de un cálculo sino también del instante en que se produzca el resultado.



- Ejemplos:
  - Un sistema informático que controla la inyección de un motor o el ABS de los frenos.

SS.OO. en Tiempo Real.
Conceptos

- Características:
  - Algunos de los procesos deben ejecutarse en tiempo real, pudiendo llegar a ser todos.
  - Estos procesos intentan controlar o reaccionar ante sucesos que tienen lugar en el mundo exterior.
  - Se asigna un plazo de tiempo a cada proceso que, según la Teoría del Muestreo, tenga una frecuencia dos veces mayor que el suceso que pretende controlar.
  - Los procesos en tiempo real también reciben el nombre de *tareas:*
    - *Unidad básica (indivisible) de trabajo desde un punto establecido de un programa de control.*

SS.OO. en Tiempo Real.
Conceptos

- Clasificación por importancia:
  - Hard-real time:
    - Traducido como tiempo real **rígido, estricto** o **duro**.
    - Indica que el período de tiempo debe cumplirse obligatoriamente.
    - Suelen ser períodos de tiempo muy pequeños.
    - Suelen ser tareas críticas para el sistema que controlan.
  - Soft-real time:
    - Traducido como tiempo real **flexible, ligero** o **suave**.
    - Indica que el período de tiempo debería cumplirse, aunque no es obligatorio.
    - Suelen ser períodos de tiempo más grandes que en hard-real time.
    - Suelen ser tareas no críticas o *menos críticas* para el sistema que controlan.

SS.OO. en Tiempo Real.
Conceptos

- Clasificación por importancia:
  - Firm-real time:
    - Traducido como tiempo real **no críticos firmes**.
    - Indica que los resultados obtenidos fuera de plazo son inútiles pero el sistema no se colapsa.
    - Dependiendo de las fuentes:
      - Algunas clasificaciones se consideran como una tercera categoría.
      - Otras consideran simplemente que es un tiempo compartido con validez (calidad) de datos no garantizada.

SS.OO. en Tiempo Real.
Conceptos

- Clasificación de las tareas en función de su periodicidad:
  - Tareas de tiempo real periódicas:
    - Indica que existe un período de tiempo conocido, establecido y que no se altera para la interacción con el mundo exterior.
  - Tareas de tiempo real aperiódicas:
    - Deben comenzar o terminar en un plazo o bien puede tener una restricción tanto para el comienzo como para la finalización
- En ambos casos, es suficiente con cumplir el Teorema del Muestreo para garantizar el cumplimiento de sus requisitos.

SS.OO. en Tiempo Real.
Características (I)

- Un S.O. en Tiempo Real suele presentar las siguientes características:
  - Cambios de contexto rápidos.
  - Pequeño tamaño (con una mínima funcionalidad asociada).
  - Capacidad de responder rápidamente a interrupciones externas.
  - Multitarea con herramientas de comunicación entre procesos, como semáforos, señales y sucesos.
  - Uso de archivos secuenciales especiales que puedan acumular datos a alta velocidad.

SS.OO. en Tiempo Real.
Características (II)

- Un S.O. en Tiempo Real suele presentar las siguientes características:
  - Planificación apropiativa (expulsiva) basada en prioridades.
  - Reducción de los intervalos en los que están inhabilitadas las interrupciones.
  - Primitivas para demorar tareas durante un tiempo fijo y para detenerlas y reanudarlas.
  - Alarmas especiales y temporizadores.

SS.OO. en Tiempo Real.

Ejemplos de clasificación de Sistemas en Tiempo Real

- Según las propiedades del Sistema controlado:
  - Sistemas críticos y sistemas acríticos
  - Sistemas con parada segura y sistemas con degradación aceptable
- Según las propiedades del sistema de tiempo real
  - Sistemas con tiempo de respuesta garantizado y sistemas que hacen lo que pueden
  - Sistemas con recursos adecuados y sistemas con recursos inadecuados
  - Sistemas dirigidos por tiempo y sistemas dirigidos por eventos
- Según la instalación del sistema de tiempo real
  - Sistemas convencionales y empotrados
  - Sistemas centralizados y distribuidos

SS.OO. en Tiempo Real.
Requisitos

- Un sistema en Tiempo Real presenta y cumple con los siguientes cinco requisitos:
  - Determinismo
  - Sensibilidad
  - Control del usuario
  - Fiabilidad
  - Tolerancia a fallos

SS.OO. en Tiempo Real.
Requisitos. Determinismo

- Es determinista cuando realiza las operaciones en instantes fijos y predeterminados o en intervalos de tiempo predeterminados.

- El determinismo se verá influenciado por:
  - La competición de los procesos por los recursos y por el tiempo del procesador.
  - Las solicitudes de servicio de los procesos que vienen dictadas por sucesos y temporizaciones externas.

- Luego, el determinismo realmente depende de:
  - La velocidad con la que pueda responder a las interrupciones.
  - La capacidad del sistema para gestionar todas las peticiones en el tiempo exigido.

SS.OO. en Tiempo Real.
Requisitos. Sensibilidad

- La sensibilidad se refiere a cuánto tiempo tarda un sistema operativo en dar servicio a la interrupción, después de reconocerla.

- Las características de la sensibilidad incluyen:
  - La cantidad de tiempo necesario para iniciar la gestión de la interrupción y comenzar la ejecución de su rutina de tratamiento.
  - La cantidad de tiempo necesario para ejecutar dicha rutina.
  - El efecto del anidamiento de interrupciones.

- El determinismo y la sensibilidad forman conjuntamente el tiempo de respuesta a sucesos externos.

SS.OO. en Tiempo Real.
Requisitos. Control del Usuario

- El control del usuario es generalmente mucho mayor en un sistema operativo de tiempo real que en un sistema operativo ordinario:
  - En un sistema operativo de propósito general, el usuario no tiene control sobre la planificación del sistema operativo o únicamente puede proporcionar directrices a grandes rasgos, tales como agrupar a los usuarios en más de una clase de prioridad.
  - En un sistema de tiempo real resulta esencial permitir al usuario un control preciso sobre la prioridad de las tareas:
    - El usuario debe poder distinguir entre tareas rígidas y flexibles
    - El usuario debe poder especificar prioridades relativas dentro de cada clase.
    - El usuario debe poder especificar características tales como el uso de paginación o intercambio de procesos, qué procesos deben estar siempre residentes en memoria principal, los algoritmos de transferencia de disco que se emplearán, qué factores disponen a los procesos en varias bandas de prioridad, etc.

SS.OO. en Tiempo Real.
Requisitos. Fiabilidad

- La fiabilidad es crítica en sistemas de tiempo real por que responde y controla sucesos en tiempo real.

- Las pérdidas o degradaciones del rendimiento pueden tener consecuencias catastróficas.

SS.OO. en Tiempo Real.
Requisitos. Tolerancia a fallos

- La tolerancia a fallos es vital para permitir el cumplimiento de la planificación del sistema en Tiempo Real.

RT-SS.OO. en Sistemas Empotrados.

Sistemas empotrados

- Los SS.OO. en Tiempo Real suelen utilizarse en Sistemas Empotrados.
- Un Sistema Empotrado es un sistema de propósito específico, que forma parte de un sistema más complejo (típicamente industrial) y cuyo SW y HW suelen ser desarrollados expresamente.
- También reciben el nombre de **sistemas embebidos** o **sistemas embarcados**.

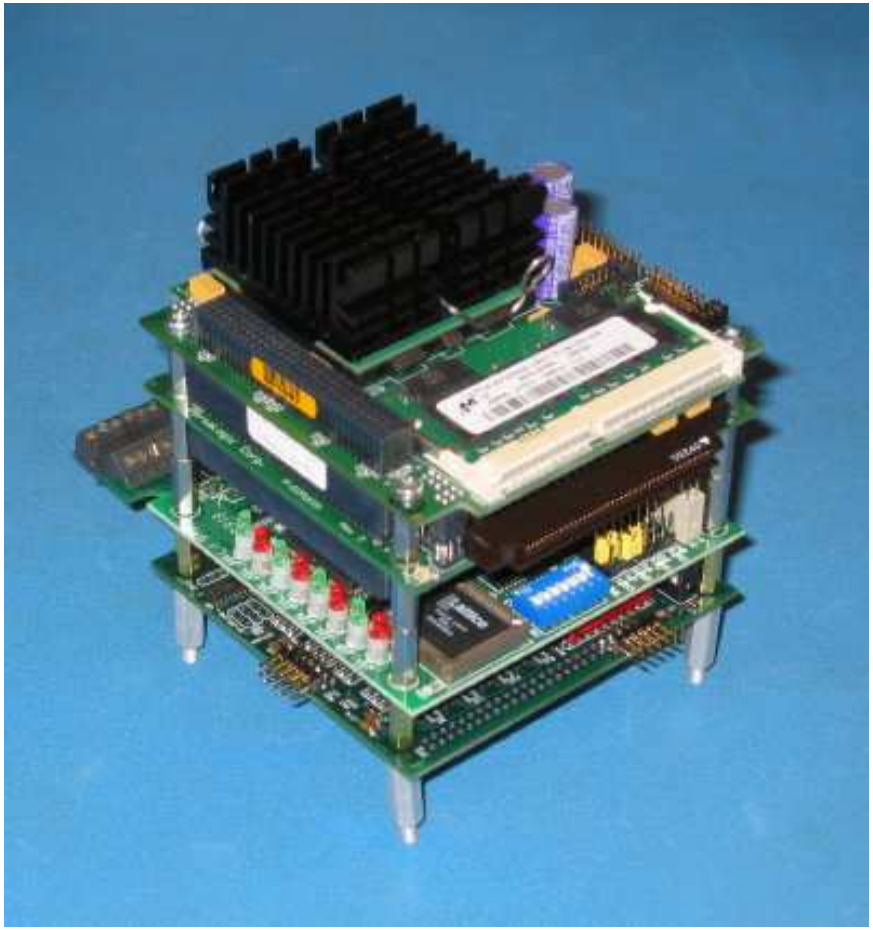
RT-SS.OO. en Sistemas Empotrados.

Sistemas empotrados. Características.

- Se diseñan para cumplir una tarea concreta
- No son de propósito general.
- A menudo tienen requisitos de tiempo real.
- No suelen ser visibles ni accesibles desde el exterior del sistema.
- No siempre son dispositivos separados
  - A menudo se integran dentro del dispositivo que controlan.
- Suelen disponer de hardware limitado.
  - CPU + memoria RAM + memoria FLASH + elementos E/S
- El software suele crearse a la vez que el dispositivo y no se suele cambiar.
- A este software específico se le suele denominar Firmware

RT-SS.OO. en Sistemas Empotrados.

Ejemplo de una placa PC/104.



Sistemas RT Certificados.

Para uso en Sistemas Críticos.

- Los SS.OO. certificados son SS. OO. :
  - En Tiempo Real, tanto Hard-RT como Soft-RT
  - Se suelen usar en Sistemas Críticos.
  - Se suelen usar en Sistemas Embarcados.
  - Se encuentran certificados por Normativa Internacional muy estricta en cuanto a RAMS.
  - Su uso está condicionado a disponer de un HW también certificado (con respecto a una normativa equivalente) que soporte sus características.

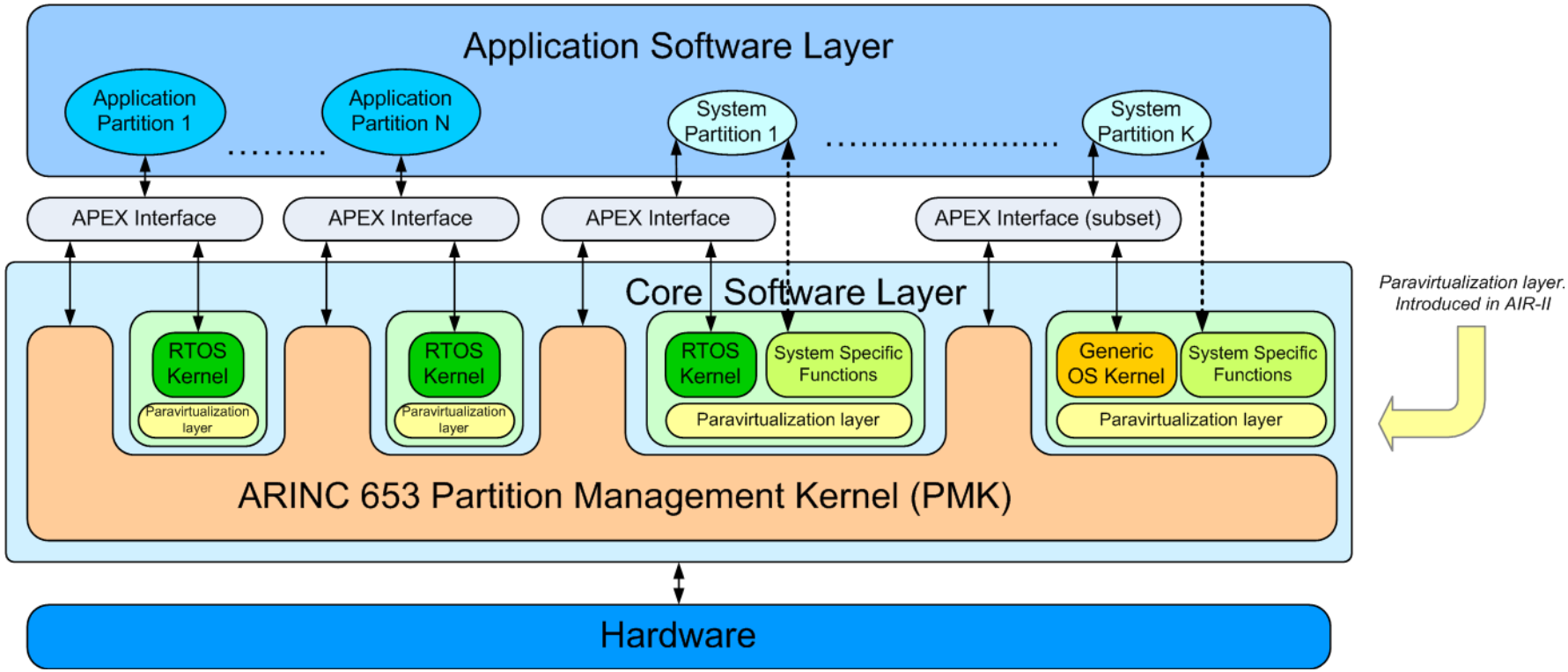
Sistemas RT Certificados.

Para uso en Sistemas Críticos.

- Algunos de los SS.OO. certificados son:
  - VxWorks AE653 2.2.1.1 WIND RIVER
  - RTOS Lynx LynxOS FROM LYNUXWORKS
  - ARINC 429
  - ARINC 653 ARINC

Sistemas RT Certificados.

Ejemplo de una arquitectura ARINC 653.

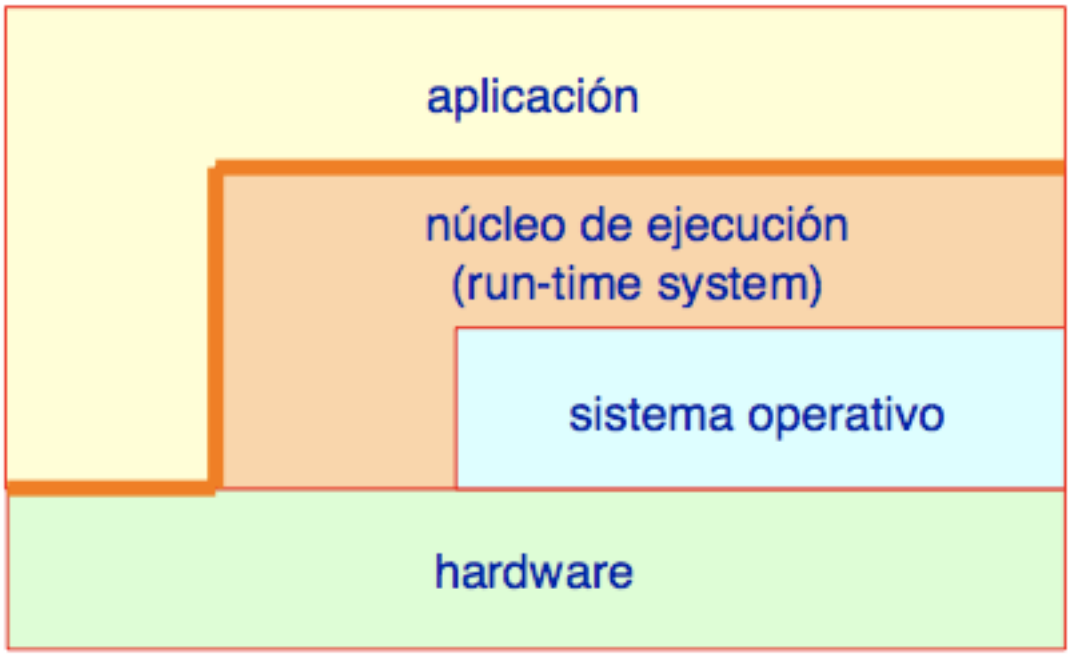


Tiempo real
Entorno de ejecución

- El firmware de un sistema empotrado suele estar compuesto por:
  - Aplicación: el programa a ejecutar
    - Suele ser un único programa por dispositivo.
  - Núcleo de ejecución (*run-time*)
    - Conjunto de bibliotecas con la funcionalidad que requiere el programa.
    - Suele estar asociado con un lenguaje concreto.
  - Sistemas operativo
    - Gestor del hardware sobre el que se apoya el núcleo de ejecución.

Tiempo real
Entorno de ejecución

- El *Firmware* puede prescindir del nucleo de ejecución, del sistema operativo e incluso de ambos (algo raro).



Tiempo real
Sistemas certificados

- Son aquellos que una entidad externa certifica que cumplen con todas los requisitos.
  - Es un requisito imprescindible para sistemas de tiempo real criticos.
  - Lo más complicado es certificar los requisitos temporales
- Solo se pueden certificar sistemas completos (HW+SW)
  - No existen programas o sistemas operativos certificados solo sistemas completos certificados.

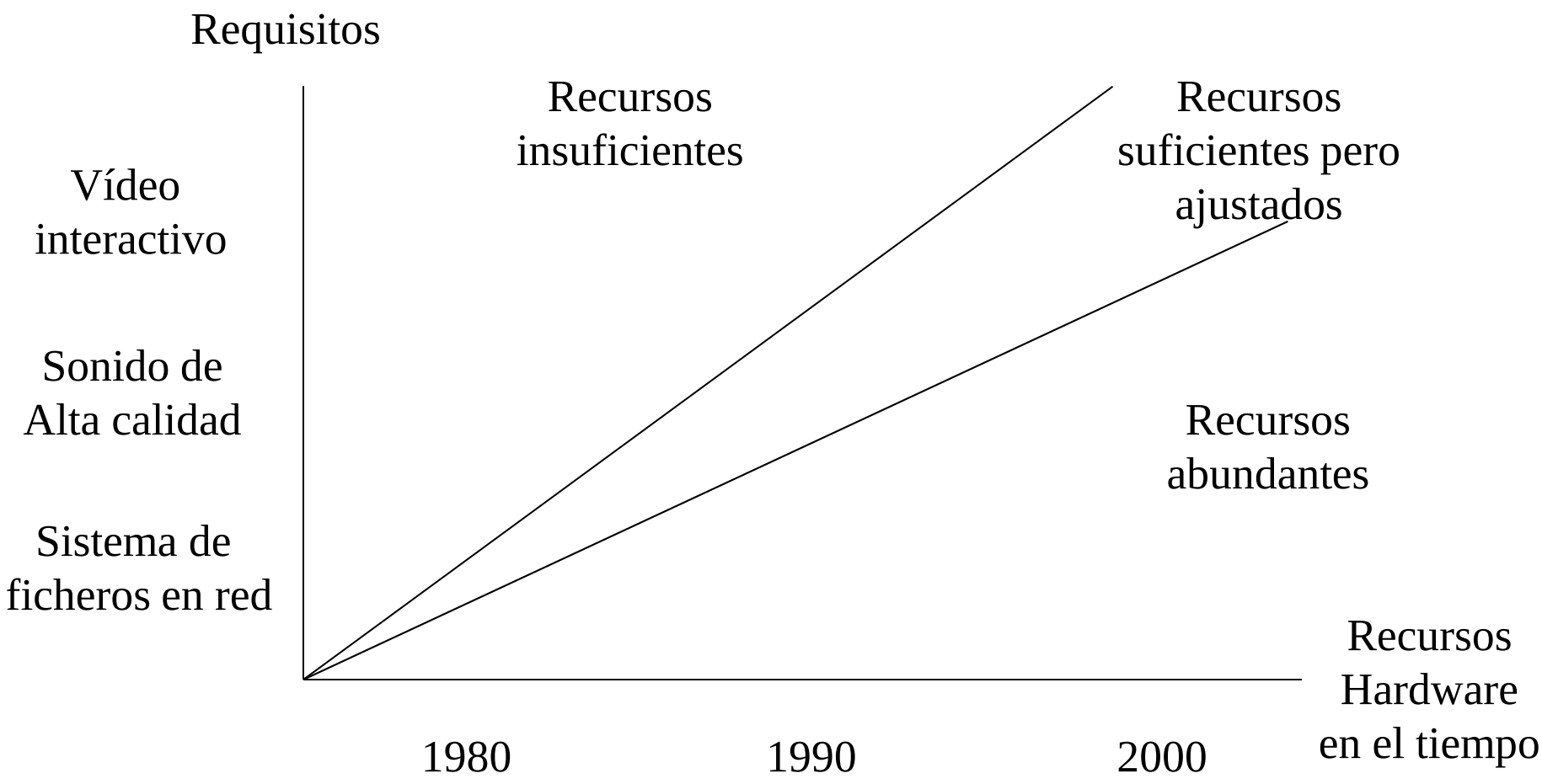
NOTA: Casi todos los sistemas certificados no llevan sistema operativo (muy pocos SSOO han participado de un sistema certificado).

Tiempo real
Desarrollo de sistemas con calidad de servicio

- Son aquellos preparados para realizar servicios cumpliendo un cierta calidad de servicio.
- Para ello utilizan un gestor de recursos que permita reservar el uso de un determinado recurso por parte de un determinado servicio.
- Para poder hacerlo deben disponer de módulos tanto fuera como dentro del kernel del S.O.
- Además todos los recursos involucrados deben permitir reservar su uso por anticipado.

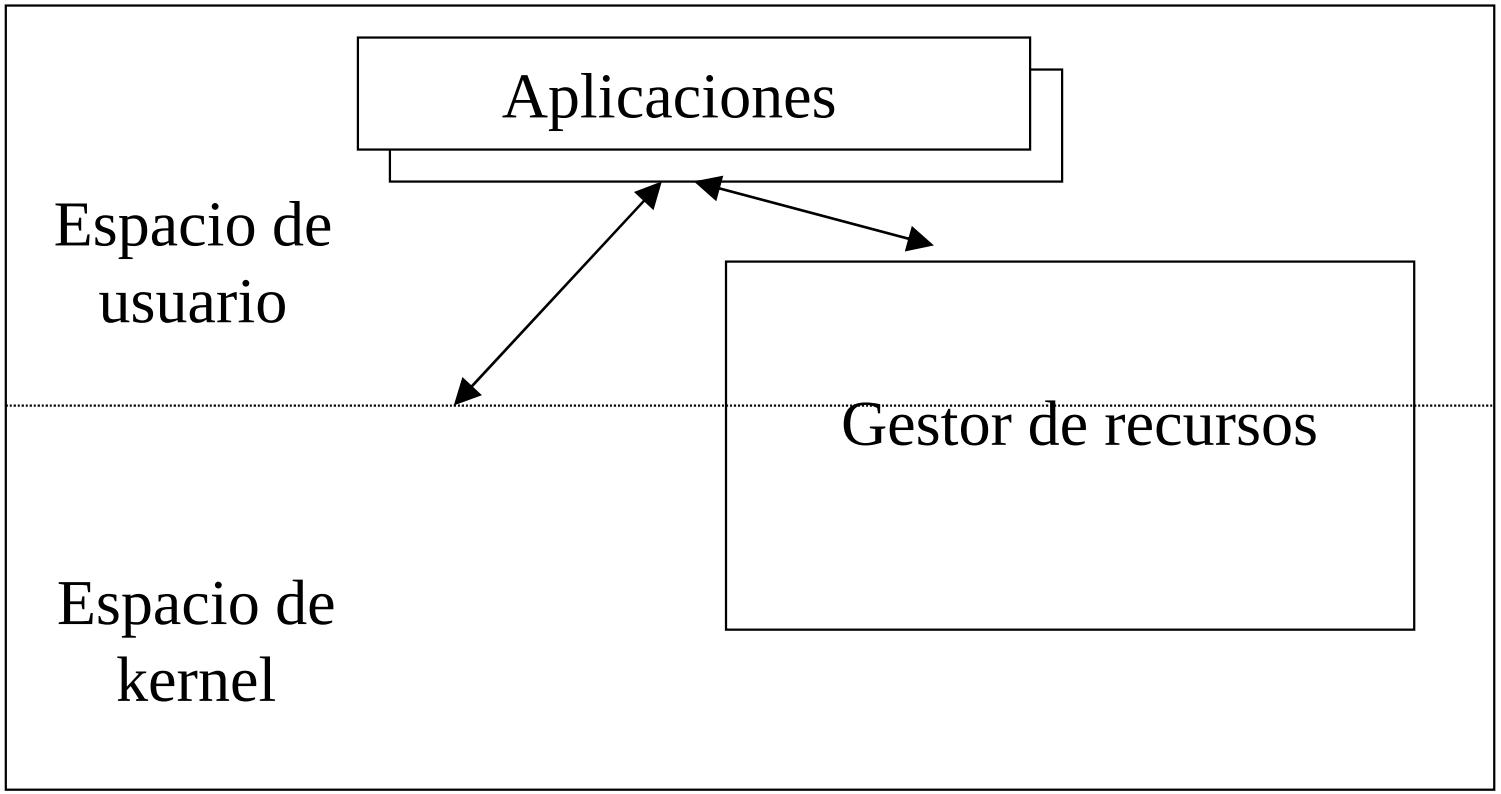
Tiempo real

Gestión de recursos con calidad de servicio



Tiempo real

Gestión de sistemas con calidad de servicio



Lección 6
Aspectos avanzados

Grupo ARCOS

Diseño de Sistemas Operativos

Grado en Ingeniería Informática

Universidad Carlos III de Madrid