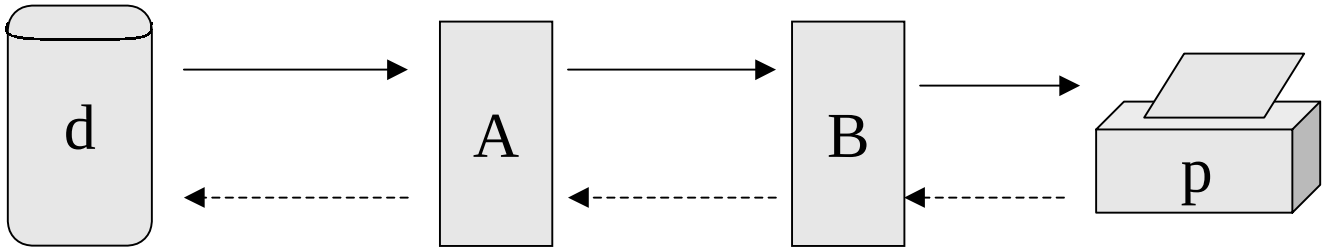
Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.1. Introducción

Tipos de servicios genéricos ofrecidos por un protocolo de comunicaciones:



DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.1. Introducción

Para imprimir un documento del disco d de A en la impresora p de B:

- Información 1 sentido. Comunicación 2 sentidos.
- Señales de control: Previo acuerdo del significado.

A todas las reglas, formatos y procedimientos preacordados entre A y B se le llama protocolo.

DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.1. Introducción

Tipos de acuerdos:

- Inicialización y terminación de un intercambio de datos.
- Sincronización de emisores y receptores.
- Detección y corrección de errores de transmisión.
- Dar formato y codificar datos.

Nótese que cada punto puede separarse en varios niveles de abstracción.

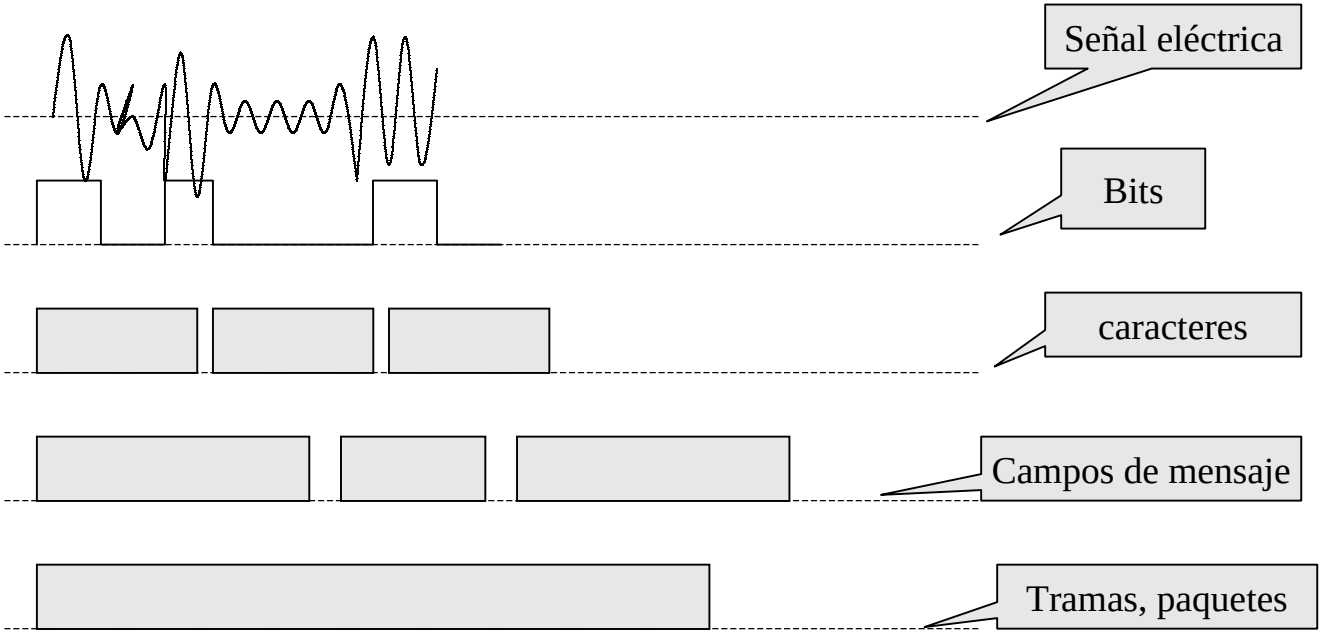
DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.1. Introducción

Formato:



DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.2. Los 5 Elementos de un Protocolo

La especificación de un protocolo consta de 5 partes distintivas. Toda especificación debe incluir:

1.- El **Servicio** que el P. (=protocolo) va a dar.

2.- Las **Suposiciones** sobre el entorno de ejecución del P.

3.- El **Vocabulario** de mensajes usado para implementar el P.

4.- La **Codificación** (formato) de cada mensaje del vocabulario

5.- Las **Reglas** (procedimientos) de intercambio de mensajes (consistencia).

francesc.burrull@upct.es

2.3. Ejemplo -Protocolo de Lynch (1968)-.
ESPECIFICACIÓN DEL SERVICIO:
Protocolo para transmitir ficheros de texto usando línea telefónica con protección contra errores de transmisión, suponiendo que son todos detectables. Transmisión de ficheros full-duplex. Uso del canal de B a A para reconocimientos, tanto positivos como negativos, del tráfico de A a B. Cada mensaje consta de 2 partes: Mensaje y Control (para el tráfico del canal opuesto).

2.3. Ejemplo -Protocolo de Lynch (1968)-.

SUPOSICIONES SOBRE EL ENTORNO:

El entorno consta de de 2 usuarios del servicio y un canal. Los usuarios hacen una petición de fichero y esperan que se complete. El canal introduce distorsión en los mensajes, pero no pierde, duplica, inserta ni reordena mensajes. Se supone un módulo de bajo nivel que recoje toda distorsión y cambia el mensaje por uno sin distorsión del tipo *err*.

2.3. Ejemplo -Protocolo de Lynch (1968)-.

VOCABULARIO DEL PROTOCOLO:

Se definen 3 tipos de mensaje:

•*ack*: mensaje combinado con reconocimiento positivo.

•*nack*: mensaje combinado con reconocimiento negativo

•*err*: mensaje combinado con error de transmisión.

V = { *ack*, *nack*, *err* }

francesc.burrull@upct.es

2.3. Ejemplo -Protocolo de Lynch (1968)-.

FORMATO DE LOS MENSAJES:

Cada mensaje consiste en un campo de control y uno de datos. Suponiendo datos de longitud fija 1 carácter:

```
enum control { ack, nack, err };

struct message {
        enum control            tag;
        unsigned char           data;
};
```

2.3. Ejemplo -Protocolo de Lynch (1968)-.

REGLAS DE PROCEDIMIENTO

De manera informal:

1.- Si la recepción previa no era errónea, el siguiente mensaje en el canal contrario transmitirá un *ack*. Si la recepción fue errónea, transmitirá *nack*.

2.- Si la recepción previa fue *nack* o *err*, retransmitir el último mensaje, sino preparar mensaje nuevo.

francesc.burrull@upct.es

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

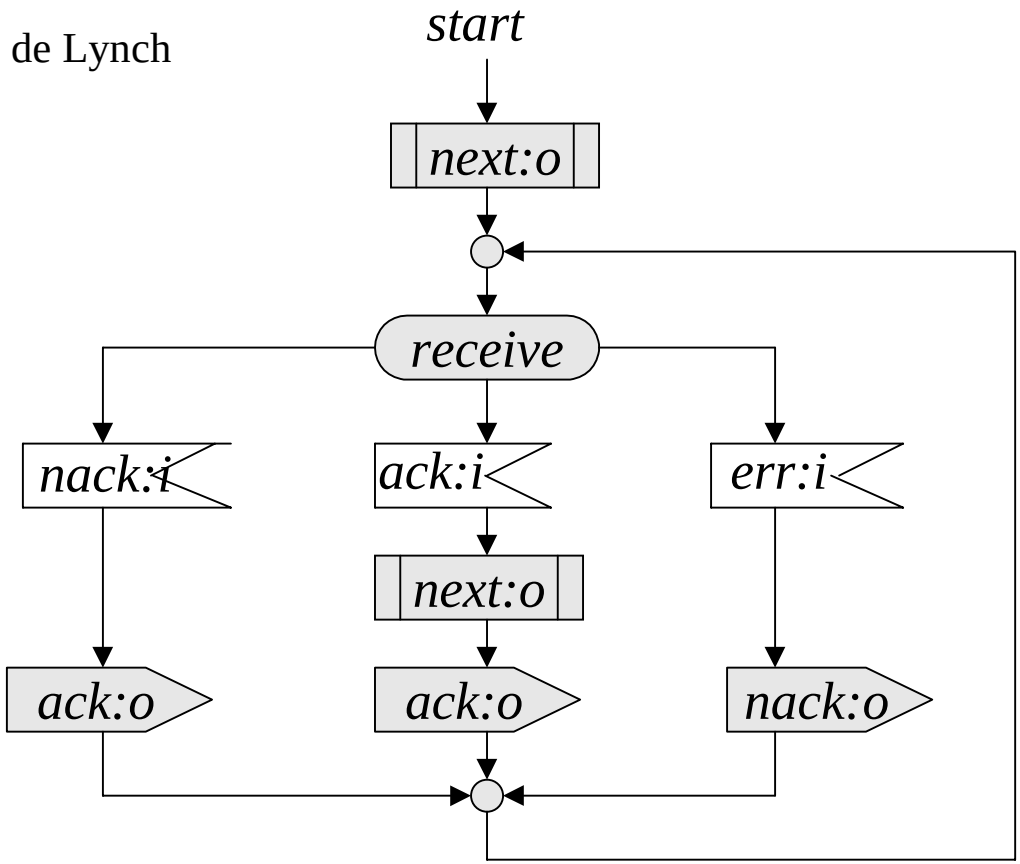
2.3. Ejemplo -Protocolo de Lynch (1968)-.
REGLAS DE PROCEDIMIENTO
De manera formal:
Hay muchos lenguajes, p.ej. Diagramas de flujo (subconjunto de SDL)

DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

Protocolo de Lynch



DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

2.3. Ejemplo -Protocolo de Lynch (1968)-.

ERRORES DE DISEÑO

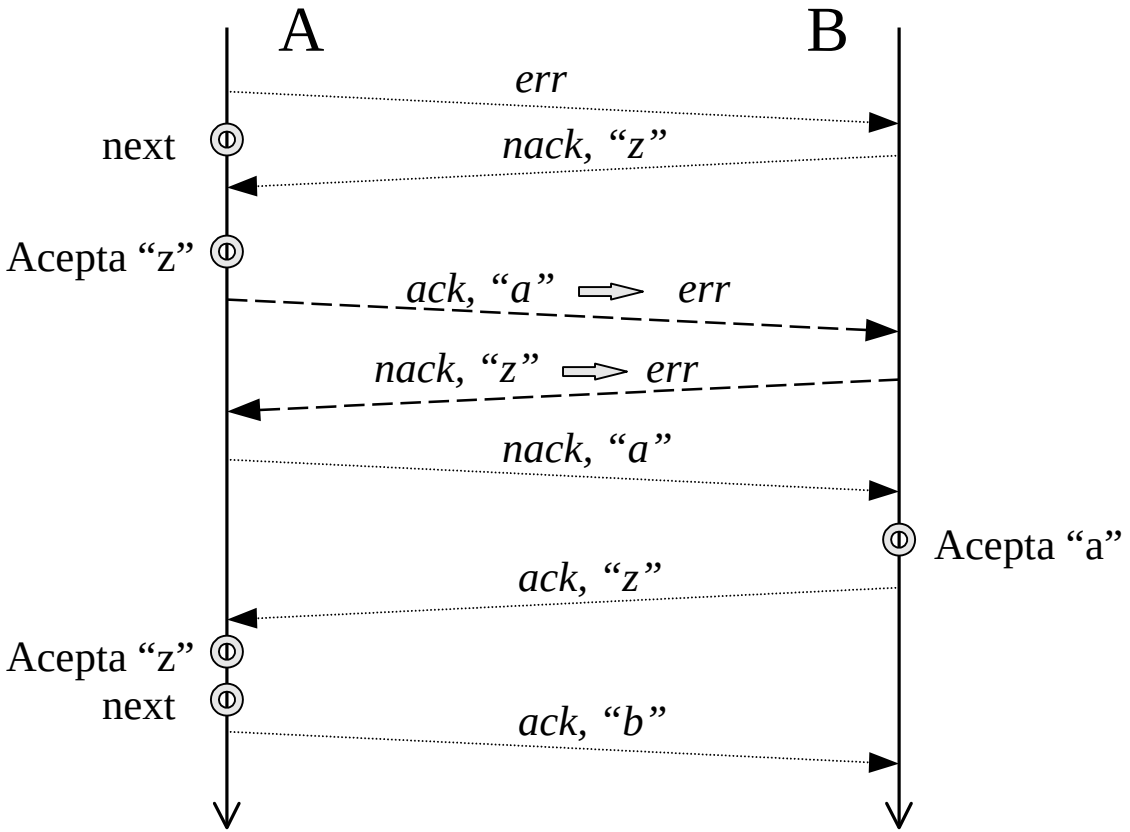
• Transmisión sólo si ambos transmiten ⇨ relleno

•No se ha especificado como iniciar ni terminar una transmisión de datos. ¿iniciar con falso *err*? ¿terminar?

•Deficiencia importante: No se ha especificado cuando hay que aceptar datos que se han recibido correctamente. ¿aceptar al recibir *ack* o *nack* y no aceptar al recibir *err*?

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios



◎ Evento interno

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

2.3. Ejemplo -Protocolo de Lynch (1968)-.

CONCLUSIONES

•A pesar de la sencillez del protocolo, es muy complicado descubrir el error.¡Podría no aparecer nunca!

•La descripción del protocolo es convincente, a pesar de no estar completamente especificado.

•Cualquier implementación permitiría errores en el intercambio de datos.

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.4. Servicio y Entorno
Servicio: Qué ofrece un protocolo.
Entorno: En qué se basa un protocolo.

EJEMPLO
protocolo que codifique caracteres en 7 bits y detecte errores añadiendo un bit de paridad.
Servicios: **codificación de datos** y **detección de errores**

DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

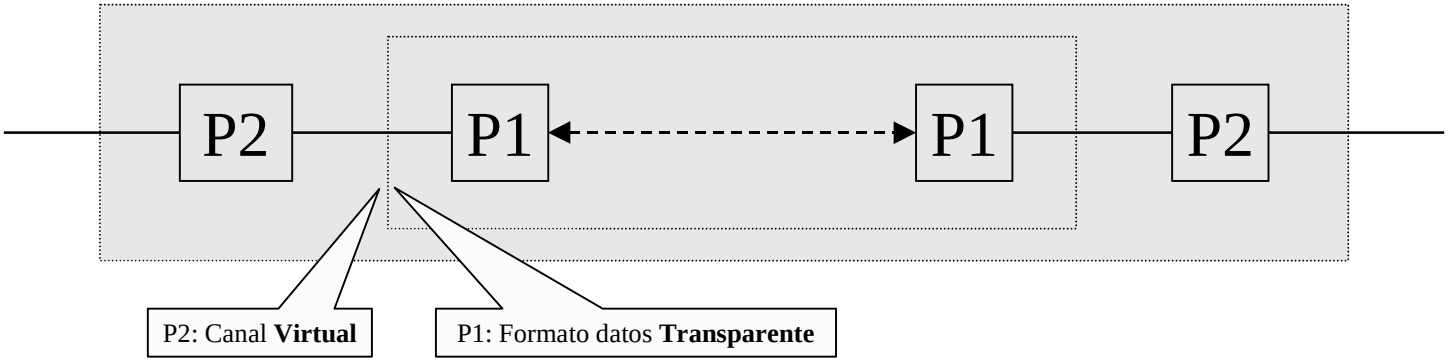
Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.4. Servicio y Entorno

Separación en:

- Módulo codificador/decodificador (P2)
- Módulo añadir/comprobar paridad (P1)



DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

2.4. Servicio y Entorno

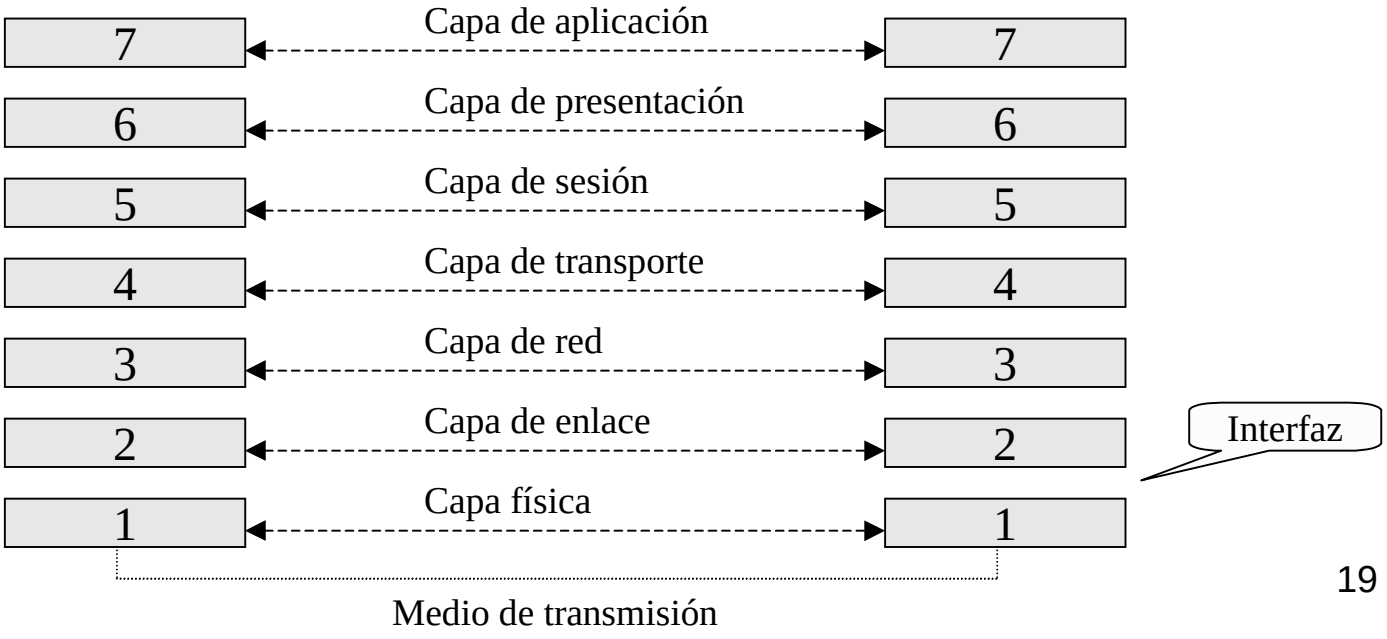
- Cada capa implementa un protocolo y ofrece un servicio
- P1: Datos 8 bits
- P2: Datos 7 bits
- El diseño jerarquizado ofrece 2 ventajas:
    - Las capas ayudan a clarificar la estructura del protocolo y las distintas funciones.
    - Si hay que modificar el protocolo, posiblemente la modificación afectará solo a alguna capa.

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.4. Servicio y Entorno

Modelo de Referencia OSI (Open Systems Interconnection) de la ISO (International Standards Organization)



DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

2.4. Servicio y Entorno

•Nivel Físico

Contiene todas las funciones relacionadas con la transmisión de bits sobre una conexión física.

- •Especifica el medio mecánico (cable, fibra, radioenlace, conectores -asignaciones de pines-, conmutadores, ...).
- • Especifica la codificación de bit.

Esconde estos detalles a capas superiores.

2.4. Servicio y Entorno

•Nivel de Enlace

Utiliza los servicios del nivel físico.

- •Corrección (tratamiento) de errores.
- •Multiplexación de flujos de datos.
- •Control de flujo
- •Unidad de datos: Trama

Convierte el canal físico en un canal fiable.

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.4. Servicio y Entorno

- Nivel de Red
  - Direccionamiento y encaminamiento
  - Evitar cuellos de botella
  - Reducción de la congestión
  - Iniciar y terminar conexiones

DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.4. Servicio y Entorno

- Nivel de Transporte
  - Interconecta procesos de usuario de manera transparente a través de una red.
  - Control de flujo extremo a extremo

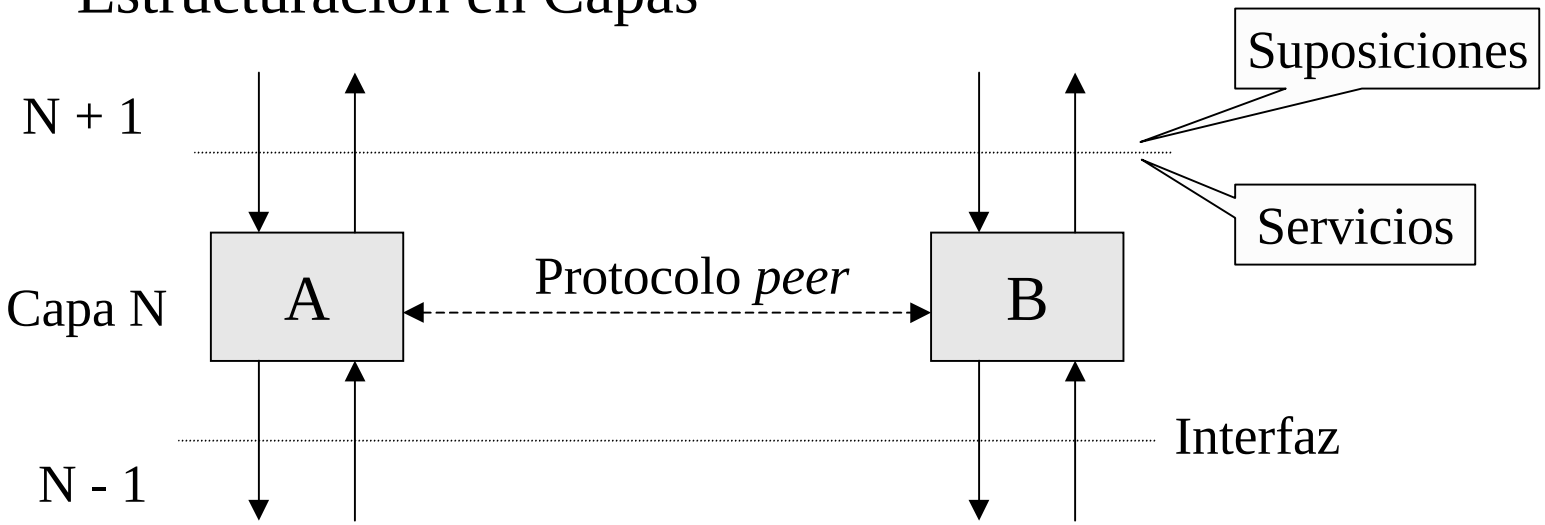
DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.4. Servicio y Entorno

- Estructuración en Capas



DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

2.4. Servicio y Entorno

En el método por capas:

•Una capa define un nivel de abstracción del protocolo. El número de capas depende del protocolo a diseñar.

•Un interfaz separa distintos niveles de abstracción. Un interfaz correcto será reducido y bien definido.

2.5. Vocabulario y Formato

En niveles bajos de abstracción, los tres métodos principales de formato son los:

- Orientados a Bit.
- Orientados a Carácter
- Orientados a nº de Bytes (*Byte-count oriented*)

En niveles altos:

- Cabeceras y Colas (*Headers and Trailers*)

2.5. Vocabulario y Formato

ORIENTADOS A BIT

Estos protocolos transmiten los datos como un flujo de bits.

- Para indicar al receptor el inicio y el fin de trama utilizan un patrón de bits prefijado (*flags*).
- Hay que evitar que los datos de usuario contengan el patrón de bits utilizado.

2.5. Vocabulario y Formato

ORIENTADOS A BIT

Ejemplo: Patrón de bits de seis unos seguidos rodeado de ceros.

- Emisor: Después de 5 “1” inserta un “0”.
- Receptor: Después de 5 “1”, si sigue un cero se elimina, sino es final de trama.

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.5. Vocabulario y Formato ORIENTADOS A BIT

Datos de usuario a transmitir

01110111110101111111110100010

Transmisión

01111110 01110111110010111111**0**111110100010 01111110

Esta técnica de *bit stuffing* se usa en el protocolo HDLC (*High level Data Link Control*) de la capa 2 de la OSI

DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

2.5. Vocabulario y Formato

ORIENTADOS A CARÁCTER

• Normalmente se asume cierto formato a nivel de bit, normalmente un número de bits -N- por carácter fijo. Así, las comunicaciones son múltiplos de N bits (8 típico).

• La unidad de datos (carácter) sirve para codificar tanto datos de usuario como información de control.

2.5. Vocabulario y Formato

Ejemplo: Código ASCII para inicio de texto STX (*Start of TeXt*) y para final de texto ETX (*End of TeXt*).

•Para evitar la confusión de datos de usuario con caracteres de control se usa código de escape DLE (*Data Link Escape*), que suprime el significado del siguiente carácter.

• Emisor añade DLE antes de cualquier carácter de control.

• Receptor interpreta que un carácter precedido por DLE no tiene significado especial.

2.5. Vocabulario y Formato
ORIENTADOS A CARACTER

Datos de usuario a transmitir

H 6 STX J 3 DLE A S ETX STX

Transmisión

STX H 6 DLE STX J 3 DLE DLE A S DLE ETX DLE STX ETX

Esta técnica se denomina *character stuffing*

2.5. Vocabulario y Formato

ORIENTADOS A Nº DE CARACTERES (*BYTE-COUNT*)

En este tipo de protocolos, para indicar donde empiezan y terminan los datos se utiliza un campo, en un lugar conocido (después de STX) de la trama, que indica la longitud en bytes de la misma.

Dejan de ser necesarios tanto el carácter ETX como las técnicas de *bit stuffing* y *character stuffing*.

2.5. Vocabulario y Formato

CABECERAS Y COLAS (*HEADERS AND TRAILERS*)

De manera genérica, los métodos de formato de datos de alto nivel se componen de 2 estructuras de datos - cabecera y cola- envolviendo los datos.

- Formato = {cabecera, datos, cola }

con

- cabecera = { tipo, destino, nº secuencia, nº de bytes, ... }
- cola = { checksum, origen, ... }

2.6. Reglas de procedimiento

Concurrencia:

- Las reglas son interpretadas por un conjunto de procesos que interaccionan.
- No siempre una interacción es reproducible.

No existe un método que a priori garantice un conjunto de reglas sin ambiguedades.

Existen herramientas que verifican la corrección y consistencia de las reglas. Existe el saber hacer.

2.7. Diseño Estructurado de Protocolos

SIMPLICIDAD: Protocolos ligeros (*Light-Weight*)

Un protocolo bien estructurado debería estar formado por un conjunto de piezas conocidas, que hagan una función y la hagan bien.

Se consigue facilidad de implementación y comprensión del funcionamiento.

2.7. Diseño Estructurado de Protocolos

MODULARIDAD: Jerarquía de funciones

Un protocolo que desempeñe funciones complejas debería construirse de piezas que interactuen de manera simple y bien definida. (Abierto, Extendible, Modificable)

Cada pieza puede desarrollarse, implementarse, verificarse y mantenerse por separado.

Funciones ortogonales deben separarse en módulos que puedan desconocerse entre sí.

2.7. Diseño Estructurado de Protocolos

ESPECIFICACIÓN ADECUADA

•Sobre-especificado: Partes de código nunca se ejecutan

•Sub-especificado (incompleto): situaciones inesperadas.

•Acotado: No desborda el sistema.

•Auto-estabilizado: Si un error modifica el estado del protocolo, éste debe volver a un estado deseable en un numero finito de transiciones.

•Auto-adaptativo: Puede modificarse de acuerdo al entorno.

2.7. Diseño Estructurado de Protocolos

ROBUSTEZ

•No es complicado diseñar protocolos que funcionan en circunstancias normales. Deben funcionar en circunstancias especiales, en cualquier condición y respondiendo a cualquier secuencia de eventos.

•Un diseño robusto se escala sin gran esfuerzo

•No sobrediseñar: Sí diseño mínimo, eliminando lo innecesario.

2.7. Diseño Estructurado de Protocolos

CONSISTENCIA

Hay circunstancias típicas donde fallan los protocolos:

- Bloqueos (*Deadlocks*): No es posible ejecutar. Todos los procesos esperan unas condiciones que nunca se darán.
- *Livelocks*: Secuencias que se repiten infinitamente sin que el protocolo progrese.
- Terminación incorrecta: Protocolo finaliza sin cumplir las condiciones de terminación adecuadas.

2.8. Diez Reglas de Diseño

Los principios anteriores conducen a estas reglas:

1.- Definir correctamente el problema: Enumerar todos los criterios de diseño, requerimientos y limitaciones (*constraints*).

2.- Definir el servicio a realizar para todo nivel de abstracción antes de escoger estructuras (que, luego cómo)

2.8. Diez Reglas de Diseño

3.- Diseñar la funcionalidad externa antes. La interna después. Primero considerar la solución como una caja negra que va a interactuar con el entorno. Después decidir cómo organizar internamente la caja negra (normalmente un conjunto de cajas negras que ...).

4.- Mantener el protocolo simple: Los problemas que parecen complejos suelen ser problemas simples juntos. Identificar problemas simples, separarlos y solucionarlos.

2.8. Diez Reglas de Diseño

5.- No conectar asuntos independientes.

6.- No introducir restricciones innecesarias. Un buen diseño suele ser extendible, más bien soluciona una clase de problemas que no un caso particular.

7.- Antes de implementar un diseño, verificar el mismo mediante un prototipo de alto nivel comprobando que se satisfacen los criterios de diseño

2.8. Diez Reglas de Diseño

8.- Implementar el diseño, medir y, si es necesario, mejorar el comportamiento.

9.- Comprobar que la versión optimizada es equivalente al diseño de alto nivel que fue verificado.

10.- No saltarse las reglas del 1 al 7 (frecuente, son las de diseño)

Tema 2: Estructura de los Protocolos

Ingeniería de Protocolos y Servicios

2.9. Resumen

•La definición de un protocolo, para ser completa, debe incluir los 5 elementos básicos (punto 2.2.). Muchos fallos en protocolos son suposiciones no definidas.

•Es necesario que no se puedan dar interpretaciones incorrectas de la especificación (no es suficiente una correcta interpretación).

•Las principales técnicas de estructuración son el software por capas y la estructuración de datos (ej. OSI).

DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es