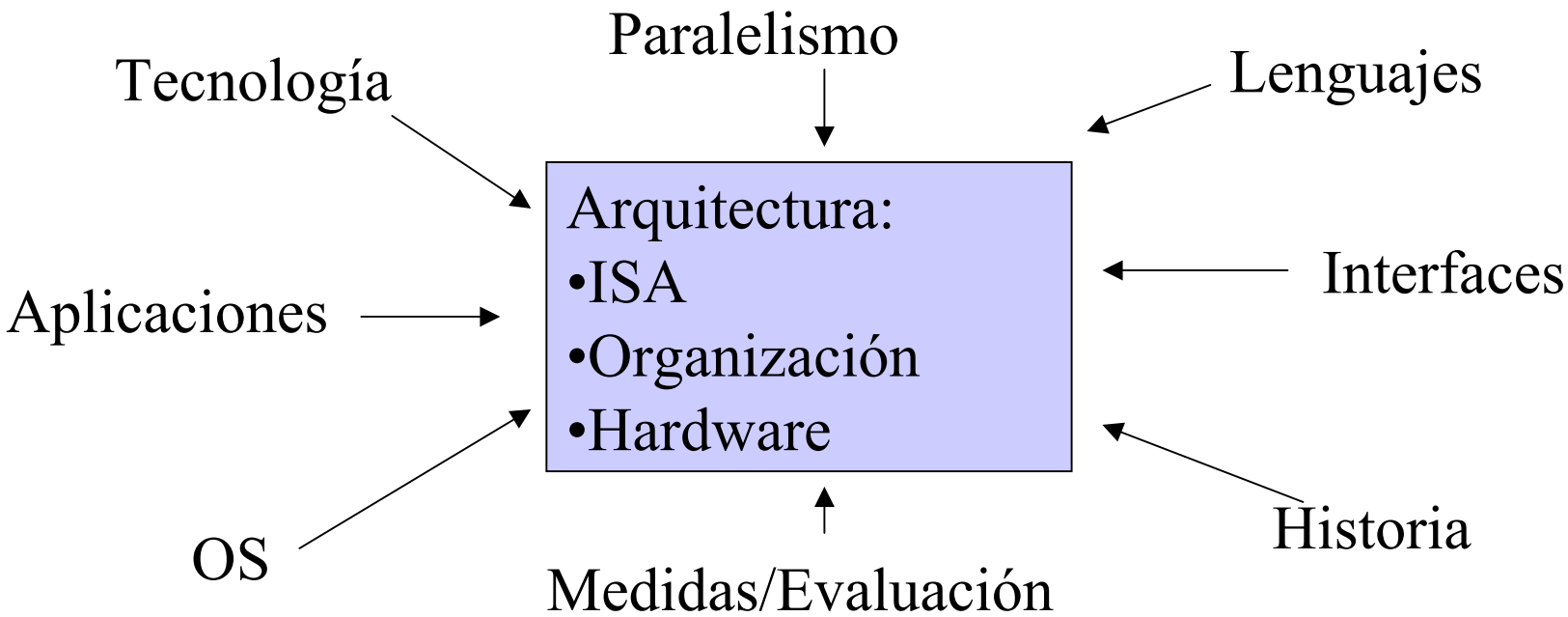
AIC: Enfoque del Curso

- Comprender técnicas de diseño, estructura de las máquinas, factores tecnológicos, métodos de evaluación que determinarán cómo serán los ordenadores del siglo XXI



1.2. Proceso de diseño

- Elaboración objeto artificial
  - Propósito u objetivo
  - Carácter del artefacto
  - Entorno con el que interactúa el artefacto

Artefacto

- **Interfaz** que presenta a su **entorno** = **arquitectura externa**.

  Está formada por las **propiedades funcionales** del objeto.

- **Estructura** del artefacto = **arquitectura interna.**

  Estructura + propiedades funcionales de los componentes.

Ej.: Arquitectura externa de un multiplexor

```
entity MUX is
  port( A, B, Ctrl: in bit; Z: out bit);
end MUX;

architecture externa of MUX is
begin
  Z <= (A and not Ctrl) or (B and Ctrl);
end externa;
```

Ej.: Arquitectura interna de un multiplexor

```
architecture interna of MUX is
      component INV port (x: in bit; y: out bit); end component;
      component AND2 port (x, y: in bit; z: out bit); end component;
      component OR2 port (x, y: in bit; z: out bit); end component;
      signal Ctrl_n, N1, N2: bit; -- Fundamental para la estructura
begin
      U0: INV  port map (Ctrl, Ctrl_n);
      U1: AND2 port map (x => A, y => Ctrl_n, z => N1);
      U2: AND2 port map (Ctrl, B, N2);
      U3: OR2  port map (N1,N2,Z);
end interna;
```

- *A esto hay que añadir las propiedades funcionales de los componentes*

```
architecture externa of INV is begin y <= not x end externa;
architecture externa of AND2 is begin z <= x and y end externa;
architecture externa of OR2 is begin z <= x or y end externa;
```

Descripción de un artefacto

- Aspecto clave: Describir **cómo la arquitectura interna da lugar a las propiedades funcionales del artefacto** (es decir, a la arquitectura externa)

Abstracción

- Artefacto: arquitectura externa más abstracta (se abstraen detalles) que arquitectura interna.
- Componentes. Tienen:
  - Arquitectura interna => otro nivel de detalle mayor.
  - Componentes (con su arquitectura interna y sus componentes, etc.).
- Conclusión: Diferentes **niveles de abstracción** en la descripción del artefacto.

Proceso de diseño

- 1.º Objetivos
- 2.º Descripciones más abstractas
- 3.º Descripciones progresivamente más detalladas con eventuales revisiones de las anteriores e incluso de los objetivos

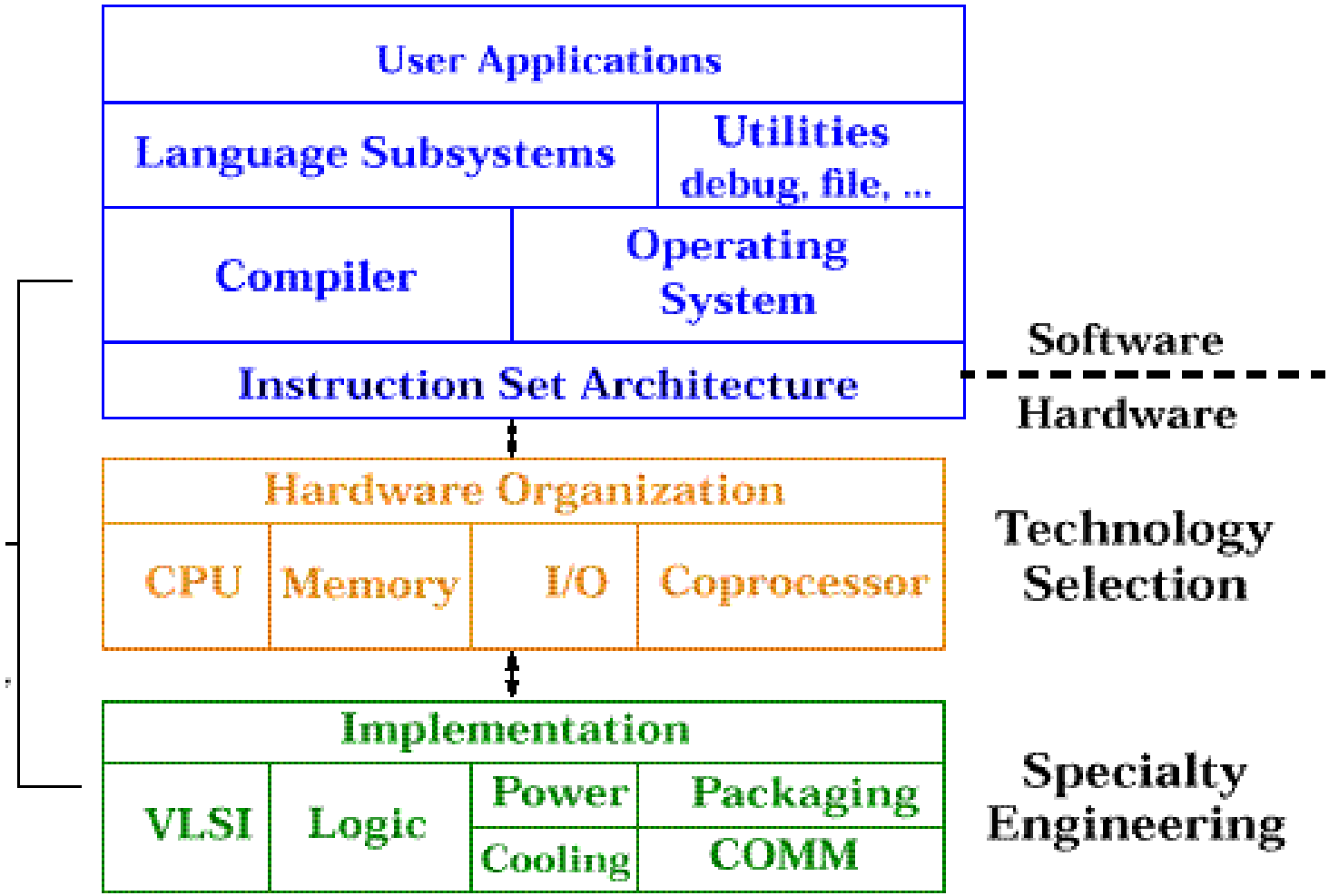
Refinamiento

- Paso básico: Pasar de una descripción funcional a una estructural => **Dividir un sistema en componentes**.
  - **Ventaja**: las interacciones dentro de un componente son más intensas que entre componentes => se puede diseñar un componente ignorando, total o parcialmente, la estructura de los otros componentes.
  - **Error** común: diseñar un componente sin tener en cuenta el resto del sistema.
- **Necesario**: **Métodos** para **evaluar** las distintas estructuras y descartar las peores.

1.3. Diseño de computadores

- Esquema básico:
  - 1.º Definición de objetivos
  - 2.º Diseño de la arquitectura externa o ISA
  - 3.º Diseño de la arquitectura interna u organización del computador
  - 4.º Diseño detallado del hardware

Aspectos del diseño



Diseño de computadores: Objetivos

- Características funcionales
- Rendimiento y coste
- Otros: Consumo, volumen, fiabilidad, tolerancia a fallos...

Requisitos funcionales

- Área de aplicación
  - propósito general ==> prestaciones equilibradas sobre una variedad de tareas
  - científico ==> rendimiento coma flotante es la clave
  - comercial ==> Cobol, aritmética decimal, bases de datos son la clave
  - tolerante a fallos ==> redundancia
- Compatibilidad SW
  - A nivel de lenguaje de programación: se precisan nuevos compiladores
  - A nivel binario: SW fácil pero HW difícil (debe soportar una arquitectura externa previamente definida)

Más requisitos funcionales

- Requisitos OS
  - Influencia del mercado destino
    - tamaño del espacio de direcciones
    - manejo memoria: paginada, segmentada
    - protección: HW debe ayudar
    - tiempo real: problemas de planificación
- Estándares
  - Mercado fuerza estándares
    - punto flotante (IEEE 754)
    - I/O - comunicaciones y periféricos (VME, SCSI, ATM...)
    - OS - Unix, DOS...
    - Lenguajes de programación: Afectan a ISA

Satisfacción de objetivos

- Hay objetivos
  - Imprescindibles (p. ej. un computador actual debe permitir multiprogramación)
  - Deseables pero no imprescindibles
  - Que se pueden conseguir en menor o mayor grado (p. ej. tamaño de memoria principal)
    - A este respecto es crítica la relación **coste/rendimiento**

Optimizando el diseño

- Requisitos: los pone el mercado/compañía
- Contradictorios
  - minimizar coste ==> diseño simple
  - maximizar prestaciones ==> diseño complejo, tecnología cara
  - minimizar time-to-market ==> simplicidad, reutilización
  - ¡batir a la competencia!
- Simulación, cuantificación, soporte...

Y además, predecir tendencias

- Si fallas, pierdes
  - IBM nos da ejemplos patológicos
  - Tecnología
    - Después de gastar $1-2 millardos en la Unión Josephson y IC 3D, escogieron CMOS (como todos)
  - Arquitectura
    - IBM desarrolló el micro cuando el resto, pero el PC se mantuvo escondido (¡usaba Intel!)
    - Pensaban que la línea 360/370 llegaba al 2000
    - Si no es por el PC, quiebran
    - Su apuesta por el mainframe les costó $5-8 millardos

Tendencias

- Memoria
  - media de programas crece 50-100% al año
  - añadir 1 bit de direcciones al año
- Lenguajes
  - utilización de HLL (C???)
  - importante papel de los compiladores
- Hacia un mercado de imágenes
  - capacidades gráficas (multimedia)
- Hacia un mercado de comunicaciones
  - Subsistemas I/O son ahora más importantes

Tendencias en Tecnología

- Circuitos integrados
  - densidad crece 50%/año
  - tamaño del dado crece 10-25%/año
  - complejidad del chip crece 60-80%/año
  - velocidad no crece tan rápido (conexiones)
- DRAM
  - densidad se cuadriplica cada 3 años
  - tiempo de acceso decrece lentamente (33%/10 años)
- Discos
  - densidad crece 50% al año
  - tiempo de acceso mejora 33%/10 años

Efectos de cambios tecnológicos

- Ciclo de producción: 2 años
- Mercado requiere algo nuevo cada 6-12 meses
- Implicaciones
  - Múltiples equipos de diseño
  - Diseño para un objetivo que sólo se conocerá al final del ciclo de diseño
  - Infraestructura y NRE (*non-recurring expense* = costes en los que sólo se incurre una vez)
    - Mínimo coste para hacer un micro: $50.000.000

1.4. Coste

- Decrece con el tiempo
  - Fundamental acertar
- Menor a mayor producción
  - Regla empírica: al doblarse el volumen el coste por unidad decrece un 10%

Coste HW

- Suma de:
  - NRE (*non-recurring expense*): se deben amortizar en el coste final de las piezas.
    - Coste del diseño
    - Fabricación de prototipos
  - Costes recurrentes o de producción (por componente o pieza). Depende de:
    - Área
    - Complejidad del proceso de fabricación
    - Rendimiento del proceso de fabricación
    - Volumen de producción

Coste de un CI (1 d 2)

Coste IC =

(Coste_del_dado + C_Comprobación + C_Empaquetado) /
/ aprovechamiento [*yield*] de la comprobación final

C. del dado =

= C_Oblea / Nº_dados_buenos_por_oblea

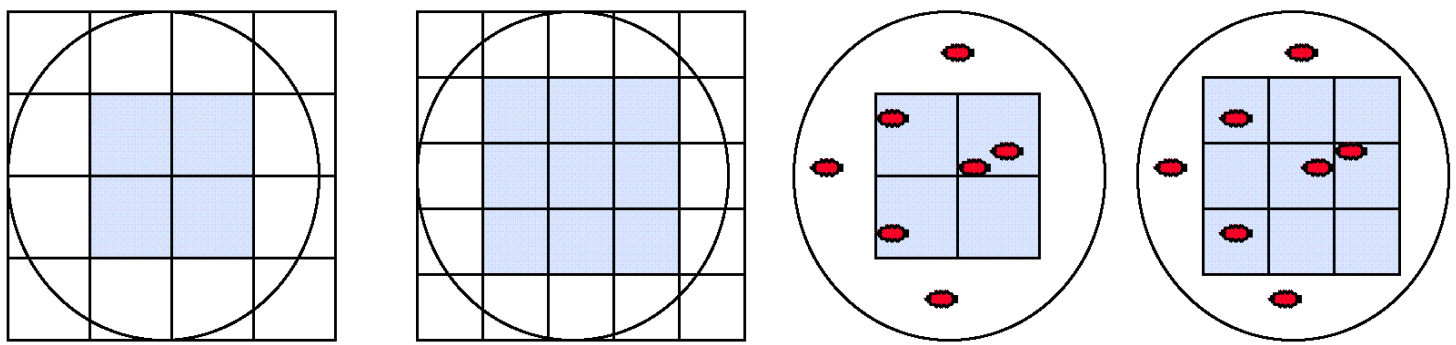
Nº_dados_buenos =

=⌊Dados_por_oblea * aprovechamiento_del_dado⌋

Coste de un CI (2 d 2)

Aprovechamiento del dado = aprovechamiento de la oblea *
*** (1 + defectos_por_unidad_de_área * área_dado /α)$^{-\alpha}$**

Dados por oblea ≅ ⌊π * radio_oblea^2 / área_dado –
- 2 * π * radio_oblea / √(2* área_dado)⌋ –
– dados_de_comprobación



Tamaño del dado

- Dados pequeños
  - Más área perdida entre dados adyacentes
  - Menor rendimiento
- Dados grandes
  - Más caros
    - Menos dados por oblea
    - Menor aprovechamiento del dado (die yield)
  - Rendimiento limitado por el número de pins
- Mejoras en la fabricación de circuitos integrados
  - Permiten dados cada vez mayores
  - El coste de la fábrica también es cada vez mayor

1.5. Rendimiento (repaso)

- Latencia y productividad
  - Latencia o tiempo de respuesta:
    - Tiempo **transcurrido** entre el comienzo y el final de un evento
      - Si el evento es la ejecución de un programa, se denomina tiempo de ejecución
  - Productividad (*throughput*) o ancho de banda
    - Cantidad de **trabajo realizado por unidad de tiempo**

Tiempo de ejecución

- En un computador de tiempo compartido conviene distinguir:
  - **Tiempo transcurrido** desde que comienza hasta que termina la ejecución de un programa
  - **Tiempo de ejecución de la CPU** (tiempo que la CPU trabaja en beneficio del programa). Tiene dos componentes:
    - Tiempo de CPU **de usuario, rendimiento de la CPU** o **TCPU**: tiempo dedicado al programa
    - Tiempo de CPU **del sistema**: tiempo dedicado a tareas del sistema operativo en beneficio del programa

Principales juegos de *benchmarks* (programas para medir el rendimiento)

- SPEC (ver *www.spec.org*). Destacamos:
  - SPEC CPU2000: rendimiento CPU
  - SPECSFS: rendimiento servidor de archivos
  - SPECWeb: rendimiento servidor web
- TPC (ver *www.tpc.org*).
  - Miden la capacidad para procesar transacciones (accesos y actualizaciones de bases de datos)

MIPS Y MFLOPS

- Medidas muy usadas pero poco fiables
- MIPS
  - Su medida depende del juego de instrucciones máquina y de los *benchmarks* utilizados
- MFLOPS
  - Se usan en computadores de orientación científica
  - Sólo puede ser orientativo el valor sostenido, nunca el de pico

Resumen de rendimientos

- Media **aritmética** de los tiempos
  - **Ponderación** ideal: el peso que se da al tiempo de un programa es proporcional a la frecuencia (a menudo desconocida) con que se ejecuta el programa
  - Alternativa: pesos inversamente proporcionales a los tiempos
- Media **armónica** (posiblemente ponderada) de las frecuencias (MIPS, MFLOPS...)
- Media **geométrica**: cuando se usan los tiempos de ejecución relativos a los de una máquina de referencia

1.6. Principios cualitativos y cuantitativos del diseño de computadores

- Para mejorar un sistema:
  - Dirigir los esfuerzos a **eliminar los cuellos de botella**, es decir, a equilibrar el sistema
- **Rendimiento de la CPU** o t. de CPU de **usuario**

  TCPU = CE * T = IE * CPI * T = IE * CPI / F
  - CE: Nº. ciclos de reloj dedicados a la ejecución
  - IE: Nº. ejecuciones de instrucciones
  - CPI: Nº. medio de ciclos por instrucción = CE / IE
  - T: período del reloj
  - F: frecuencia del reloj

Ley de Amdahl

Gg = To/Tm = 1/(1-F+F/Gp)

- To: tiempo en que el sistema original realiza una tarea
- Tm: tiempo en que el sistema mejorado realiza la misma tarea
- F: fracción de tiempo dedicado a la tarea que consumía, en el sistema original, la parte que se mejora
- Gp: ganancia de la parte mejorada

Sobre la ley de Amdahl

- Tener presente que **no es válida si** el tiempo de la parte que se mejora **se solapa** con el de la parte que no se mejora
- Debemos dirigir nuestros esfuerzos a **acelerar las partes que consumen más tiempo** (precisamente los cuellos de botella consumen mucho tiempo)
- Por mucho que aumentemos la ganancia parcial,la **mejora global** está **limitada** por el valor 1/(1-F)