Evaluación y Explotación de Sistemas Informáticos

Carga de Trabajo
Selección, caracterización y predicción

- Introducción
  - Definiciones
  - Características de los modelos de carga
  - Construcción de modelos
- Representatividad de un modelo de carga
- Tipos de carga de trabajo
- Comparación de sistemas
  - Benchmarks
- Selección de la carga de trabajo
- Caracterización de la carga
  - Tendencia central, dispersión
  - Histogramas
  - Análisis de componentes principales
  - Análisis de conglomerados (cluster)

Introducción (I)

- Carga de trabajo (*workload*)
  Conjunto de todas las peticiones que el sistema recibe de su entorno durante un periodo de tiempo dado.

- El análisis de la carga desempeña un papel fundamental en todos los estudios en los que hay que determinar índices de rendimiento.
- Los índices de rendimiento están directamente relacionados con la carga y no pueden expresarse independientemente de la carga.
- El índice del rendimiento de un sistema siempre debe ir acompañado de la información de la carga bajo la que fue determinado.

Introducción (II)

- Dificultades
  - Manejar cargas reales con un gran número de elementos
  - Variación de la carga a lo largo del tiempo
  - Carga interacciona con el sistema que la procesa
    - Realimentación interna: variaciones de los parámetros del sistema operativo
    - inducen en la carga. Ej. tiempo de servicio de los discos en función de la carga.
    - Realimentación externa: incidencia del comportamiento del sistema (tiempos de respuesta) sobre los hábitos y comportamiento de los usuarios.
- El comportamiento de la carga real es muy complejo y es difícil de reproducir.
- Es necesario construir un modelo de la carga que capture las características más relevantes de la carga real.

Introducción (III)

- El modelo de carga ha de capturar el comportamiento estático y dinámico de la carga real y ha de ser compacto, repetible y preciso.
- El modelo de carga supone una descripción cuantitativa de las características de la carga.
  - A esta descripción cuantitativa se le denomina **caracterización de la carga**.
- El modelo ha de establecerse en función de los parámetros que pueden afectar al comportamiento del sistema.
- Una carga está perfectamente caracterizada si su resultado es un conjunto de parámetros cuantitativos seleccionados de acuerdo con los objetivos de la operación de caracterización.

Características de los modelos de carga (I)

- Reproducibilidad.
  Debemos ser capaces de reproducir la carga de prueba sobre todo en situaciones de ajuste de sistemas o de comparación de sistemas.
- Representatividad.
  Los aspectos de la carga real han de estar representados en el modelo.
- Compacidad.
  La ejecución de modelos reales puede llevar demasiado tiempo. Por este motivo es conveniente utilizar modelos de carga compactos (sin pérdida de representatividad) que permitan realizar las mediciones del sistema en tiempos cortos.

Características de los modelos de carga (II)

- Privacidad.
  En muchas ocasiones la utilización de cargas reales puede afectar a programas y datos reales protegidos. La utilización de modelos nos permite evitar estos problemas de privacidad y seguridad.
- Consistencia/coherencia
  Es necesario obtener una representación de la carga consistente con su aplicación: procesamiento de modelos matemáticos, consultas SQL en un SGBD.
- Flexibilidad.
  Posibilidad de variar los parámetros del modelo de carga para ajustarlo a las variaciones que se produzcan en el sistema real.
- Independencia del sistema.
  Es importante que, sobre todo, en los problemas de selección la representatividad del modelo no varíe al ir cambiando el sistema sobre el que se procesa.

Técnicas de modelado de la carga de trabajo

- La ejecución de la carga de trabajo es un fenómeno aparentemente determinista, pero ...
  - la gran cantidad de datos a medir
  - el gran número de variables que interactúan
  - y el gran número de fenómenos ha considerar
- hacen que la carga se modele como un fenómeno no determinista mediante la aplicación de técnicas estadísticas.

Modelado de la carga de trabajo

- Los pasos principales en la construcción de un modelo de carga :
  - Formulación
  - Recolección de parámetros de la carga
  - Análisis estadístico de los datos medidos
  - Representatividad.

Workload: Formulación (I)

- Nivel de caracterización. ¿Propósito del estudio?
  - Ejemplo. Estudiar un web server.
    - **Objetivo:** Coste/beneficio de crear un servidor proxy con caché.
    - Características necesarias de la carga:
      - **frecuencia de la referencia de documentos, concentración de referencias, tamaños de los documentos, tiempo entre referencias.**
    - **Objetivo:** Impacto de una CPU más rápida en el tiempo de respuesta.
    - Características necesarias de la carga:
      - **Tiempo medio de CPU por petición, número medio de operaciones de E/S por petición, tiempo medio de respuesta.**
- Componente básico de la carga.
  - Nivel de detalle: comando interactivo, transacciones de la base de datos, peticiones, etc.

Workload: Formulación (II)

- Seleccionar los parámetros para la descripción de la carga.
- Los parámetros caracterizan cada tipo de componente básico
  - De tipo general
    - Tiempo de CPU, operaciones de E/S, tamaño memoria demandada, ficheros de disco utilizados, prioridad asignada, tiempo medio de ráfaga de CPU, tiempo medio de ráfaga de E/S.
  - Elegidos según el tipo de información de entrada requerido por el modelo analítico a construir. En este caso se pueden dividir en dos categorías:
    - **Intensidad de carga**: tasa de llegadas, nº de clientes y tiempo de reflexión, nº de procesos o hilos de proceso en ejecución simultanea.
    - **Demandas de servicio**: se especifica mediante una k-tupla
      $(D_{i1}, D_{i2},...,D_{ik})$
      donde k es el nº de recursos considerados y D_{ik} es la demanda de servicio del componente básico i en el recurso j.
- Definir el criterio de evaluación de la representatividad del modelo.

Workload: Análisis estadístico (I)

- Análisis preliminar. Partición de los datos (partición de la carga)
  - Carga real es una colección de componentes heterogéneos y los datos recogidos pueden incluir poblaciones diferentes de la carga.
  - Es necesario descubrir las particiones naturales.
  - Conjuntos de datos más manejables.
  - Se mejora la representatividad
  - Se incrementa el poder predictivo del modelo.
- Análisis de las distribuciones de los parámetros.
  - Transformaciones de los datos originales. Ej: transformación log.
  - Identificación y eliminación de datos raros/aislados.

Workload: Análisis estadístico (II)

- Muestreo
  - Si la cantidad de datos es muy grande, se puede elegir una muestra de los datos medidos para conseguir un tiempo de procesamiento y una cantidad de espacio de almacenamiento razonable.
- Análisis estático.
  - Clasificación y partición de los componentes de la carga.
    - Análisis de componentes
    - Análisis cluster.
- Análisis dinámico.
  - Se realiza cuando hay que reproducir las características de variación temporal de la carga. Series temporales, procesos estocásticos.

Workload: Representatividad

- Verificación del modelo
  - Consistencia de los componentes del modelo.
  - Consistencia con el sistema.
  - Consistencia con la carga real.

- *Un modelo de carga W' es perfectamente representativo de W si produce los mismos valores de los índices de rendimiento P que son obtenidos cuando W se ejecuta sobre el sistema S.*

Representatividad de un modelo de carga

- Un modelo constituye una abstracción de la realidad que se pretende representar.
- Debe describir los aspectos más importantes de la carga de forma precisa.
- La precisión de un modelo para representar la carga real de un sistema se conoce como representatividad del modelo.

  Representatividad : medida de similitud entre el modelo y la carga.
- La representatividad del modelo depende del nivel de modelado adoptado y de los objetivos del estudio.

Representatividad de un modelo de carga: Niveles

- Físico
  - El modelo de carga se basa en los consumos absolutos o unitarios de los recursos hardware y software.
- Virtual
  - En este nivel se consideran los recursos a nivel lógico.
- Funcional
  - La carga viene determinada por las aplicaciones que la componen. En el modelo deben aparecer programas que efectúen las mismas funciones que la carga original.

Representatividad de un modelo de carga: Nivel Físico

- Orientación al consumo de recursos físicos. Gran dependencia del sistema
  - Relativamente fáciles de construir
  - Los datos que se necesitan para construir estos modelos son fácilmente obtenibles de los sistemas operativos de los equipos (log, account) o pueden obtenerse fácilmente con monitores)
- Aplicaciones
  - Ajuste del sistema
  - Planificación de la capacidad residual (porcentaje de la potencia del sistema que no se utiliza)
  - Diseño
- Ejemplos
  - Tiempo de CPU consumido, Nº de instrucciones máquina ejecutadas, espacios de memoria, tiempo E/S consumido, nº archivos utilizados, nº y duración de los accesos físicos de E/S a canal y disco, etc.
- *Un modelo de carga W' representa perfectamente la carga W si solicita los mismos recursos físicos en las mismas* ***proporciones*** *que W.*

Representatividad de un modelo de carga: Nivel Virtual

- Orientación a recursos lógicos. Menor dependencia del sistema
  - Mayor proximidad al punto de vista del usuario, próximos al punto de vista del programador.
  - Mayor dificultad para obtener los parámetros para la construcción del modelo.
- Aplicaciones
  - Modificación de la configuración del sistema.
  - Estudios de ampliación.
  - Determinación de la capacidad de un sistema.
- Ejemplos
  - Nº de sentencias de cada tipo del lenguaje de alto nivel, nº de accesos lógicos a cada archivo o base de datos, nº y tipo de órdenes interactivas, etc.
- *Un modelo de carga W' representa perfectamente la carga W si solicita los mismos recursos físicos con la misma* ***frecuencia*** *que W.*
  - El modelo de carga, por ej., deberá provocar por unidad de tiempo el mismo nº de ráfagas de CPU y de E/S de la misma duración que en el sistema real.

Representatividad de un modelo de carga: Nivel Funcional

- Orientado a aplicaciones. Independencia del sistema
  - Dificultad de diseñar sistemáticamente.
  - Las funciones que componen la carga deber describirse de forma independiente del sistema.
- Aplicaciones
  - Adquisición de sistemas.
  - Planificación de la capacidad.
- Ejemplos
  - Si un sistema realiza 400 horas de compilaciones, 250 horas de pruebas de trabajos, 700 horas de explotación de trabajos y 300 horas de cálculo científico, un modelo funcional sería el que usara 40 minutos de compilaciones, 25 de pruebas de trabajo, 70 de explotación de trabajos y 30 de cálculos científicos.
- *Un modelo de carga W' representa perfectamente la carga W si realiza las mismas funciones con las mismas proporciones que W.*

Representatividad de un modelo: Evaluación

- Caracterización orientada a recursos.
- Parámetro para cada componente básico de carga de trabajo: pasos de programa.

Valores medios de los parámetros que caracterizan la carga real *W* y los dos modelos *W'* y *W''*

Parámetro		*Clase 1*	*Clase 2*	*Clase 3*	*Mín*	*Máx*
Para W						
Tiempo CPU	1 t_{CPU}	80	60	5	0	190
Operaciones E/S de disco	2 $n_{I/O}$	150	1200	40	30	1650
Memoria utilizada (Kbytes)	3 mem.	40	90	20	8	180
Fichero de disco usados	4 n.fich	2,5	4,1	1,8	0	6
Líneas impresas	5 n.lin	300	1100	50	6	2250
Para W'						
Tiempo CPU	1 t_{CPU}	75	62	4		
Operaciones E/S de disco	2 $n_{I/O}$	144	1220	51		
Memoria utilizada (Kbytes)	3 mem.	38	89	18		
Fichero de disco usados	4 n.fich	2,2	4,2	2		
Líneas impresas	5 n.lin	220	1250	65		
Para W''						
Tiempo CPU	1 t_{CPU}	71	64	9		
Operaciones E/S de disco	2 $n_{I/O}$	146	1190	48		
Memoria utilizada (Kbytes)	3 mem.	40	86	18		
Fichero de disco usados	4 n.fich	2,4	4,1	2,1		
Líneas impresas	5 n.lin	320	1180	48		
Número de pasos de programa por clase /% de la carga W) q_j		27%	31%	36%		

Representatividad de un modelo: Evaluación: Normalización

- ¿Cuál de los modelos *W'* y *W''* es más representativo?

- Normalizar los parámetros de todos los modelos y la carga real, dando un valor en el intervalo (0,1). (Escalado lineal)
  - Sea:
    - v_j el valor del parámetro j para un componente de la carga.
    - v_{ij} el valor el valor medio del parámetro j para los componentes de la clase i.
    - $v_{jmín}$ el valor mínimo de v_j para todos los componentes.
    - $v_{jmáx}$ el valor máximo de v_j para todos los componentes.
  - El valor normalizado v'_{ij} será:

$$v'_{ij} = \frac{v_{ij} - v_{jmín}}{v_{jmáx} - v_{jmín}}$$

- En el ejemplo:
  - $v_{jmín}$ el valor mínimo de todos los pasos de programa v_j
  - $v_{jmán}$ el valor máximo de todos los pasos de programa v_j

Representatividad de un modelo: Evaluación: Normalización (II)

- Se calcula la distancia entre los valores normalizados de los modelos y el valor normalizado de la carga real como el valor absoluto de la diferencia de los valores normalizados.

$$\left| v'_{ij} - v_{jj} \right| \quad , \left| v''_{ij} - v_{jj} \right|$$

- … o sus diferencias cuadráticas.

Representatividad de un modelo: Evaluación: Normalización (III)

- Se pondera doblemente la distancia entre los parámetros del modelo y los de la carga que se quiere modelar, teniendo en cuenta:
  - q_i el porcentaje de programas de la clase i en la carga
  - w_j el peso asociado al parámetro j. Estos pueden tomar cualquier valor.
- Esta distancia se calcula mediante:

$$D' = \sum_{i=1}^{n} q_i \sum_{j=1}^{k} w_j \times \left| v'_{ij} - v_{ij} \right|$$

$$D'' = \sum_{i=1}^{n} q_i \sum_{j=1}^{k} w_j \times \left| v''_{ij} - v_{ij} \right|$$

- Los pesos asociados a los parámetros (w_j) tienen unos valores dependiendo del objetivo respecto al cual se quiera medir la representatividad.
  - Si se quiere evaluar la bondad de un modelo respecto a la actividad de entrada salida se dará un peso mayor a los w_j que estén relacionados con operaciones de E/S.

Representatividad de un modelo: Evaluación: Normalización (IV)

- El modelo que diste menos de la carga real será el más representativo para ese objetivo específico.

Representatividad de un modelo: Ejemplo

Parámetro			*Clase 1*	*Clase 2*	*Clase 3*
Para W					
	Tiempo CPU	1 t_{CPU}	0,4211	0,3158	0,0263
	Operaciones E/S de disco	2 $n_{I/O}$	0,0741	0,7222	0,0062
	Memoria utilizada (Kbytes)	3 mem.	0,1860	0,4767	0,0698
	Fichero de disco usados	4 n.fich	0,4167	0,6833	0,3000
	Líneas impresas	5 n.lin	0,1310	0,4875	0,0196
Para W'					
	Tiempo CPU	1 t_{CPU}	0,3947	0,3263	0,0211
	Operaciones E/S de disco	2 $n_{I/O}$	0,0704	0,7346	0,0130
	Memoria utilizada (Kbytes)	3 mem.	0,1744	0,4709	0,0581
	Fichero de disco usados	4 n.fich	0,3667	0,7000	0,3333
	Líneas impresas	5 n.lin	0,0954	0,5544	0,0263
Para W''					
	Tiempo CPU	1 t_{CPU}	0,3737	0,3368	0,0474
	Operaciones E/S de disco	2 $n_{I/O}$	0,0716	0,7160	0,0111
	Memoria utilizada (Kbytes)	3 mem.	0,1860	0,4535	0,0581
	Fichero de disco usados	4 n.fich	0,4000	0,6833	0,3500
	Líneas impresas	5 n.lin	0,1399	0,5232	0,0187

Representatividad de un modelo: Ejemplo

Parámetro		$\lvert v'_{ij}-v_{ij}\rvert$ Clase 1	Clase 2	Clase 3	$\lvert v''_{ij}-v_{ij}\rvert$ Clase 1	Clase 2	Clase 3
1 Tiempo CPU	1 t_{CPU}	0,0263	0,0105	0,0053	0,0474	0,0211	0,0211
2 Operaciones E/S de disco	2 $n_{I/O}$	0,0037	0,0123	0,0068	0,0025	0,0062	0,0049
3 Memoria utilizada (Kbytes)	3 mem.	0,0116	0,0058	0,0116	0,0000	0,0233	0,0116
4 Fichero de disco usados	4 n.fich	0,0500	0,0167	0,0333	0,0167	0,0000	0,0500
5 Líneas impresas	5 n.lin	0,0357	0,0668	0,0067	0,0089	0,0357	0,0009

$$D' = \sum_{i=1}^{3} q_i \sum_{j=1}^{5} w_j \times \left| v'_{ij} - v_{ij} \right|$$

$$D'' = \sum_{i=1}^{3} q_i \sum_{j=1}^{5} w_j \times \left| v''_{ij} - v_{ij} \right|$$

- Donde:
  - q_i es el porcentaje de pasos de programa pertenecientes a la *i*-ésima clase: (q_1=27%, q_2=31%, q_3=36%)
  - w_j es el peso asociado al *j*-ésimo parámetro y depende del contexto del estudio.

Representatividad de un modelo: Ejemplo

- Objetivo 1: Estudio de la actividad de E/S del disco para ajustar el subsistema de discos.
  - Parámetros importantes $n_{I/O}$ y n.fich.
  - Pesos w_1=1, w_2=2, w_3=1, w_4=2 y w_5=0.5
  - Se obtiene D'=0.1137 y D"=0.0989
  - W'' es más representativa que W'.
- Objetivo 2: Estudio de la capacidad residual de la CPU.
  - Parámetros importantes t_{CPU}.
  - Pesos w_1=2, w_2=1, w_3=1, w_4=1 y w_5=0.5
  - Se obtiene D'=0.0881 y D"=0.0989
  - W' es más representativa que W''.
- Objetivo 3: Estudio de la mejora debida a acciones de ajuste en varios componentes del sistema: memoria, CPU, subsistemas E/S.
  - Parámetros importantes: todos pesan igual.
  - Pesos w_1=1, w_2=1, w_3=1, w_4=1 y w_5=1
  - Se obtiene D'=0.0924 y D"=0.0791
  - W'' es más representativa que W'.

Carga de trabajo de prueba: Tipos

- Carga de trabajo de prueba (*test workload*)
  - Es la carga de trabajo procesada por un sistema mientras se realizan mediciones.
- Tipos:
  - Real
  - Sintética
    - Natural (*Benchmarks*)
    - Híbrida
  - Artificial
    - Ejecutable
      - Mix de intrucciones
      - Programas sintéticos
      - Kernels
      - Script sintéticos (Application workloads)
    - No-ejecutable
      - Modelos analíticos
      - Modelos dirigidos por distribuciones

Carga *de trabajo* de prueba: Terminología

- **Carga de prueba**: Cualquier carga utilizada en estudios de rendimiento
- **Carga real**: Consiste en todos los programas, transacciones, etc, procesadas durante un periodo de tiempo dado. No puede repetirse.
- **Modelo de carga**: Características similares a las de la carga real y puede ser aplicada repetidamente de forma controlada.
- **Carga sintética**: Se construye utilizando los componentes básicos (programas, comandos interactivos, etc.) de la carga real y componentes desarrollados especialmente (programas sintéticos, kernels, scripts sintéticos, etc.)
- **Carga artificial**: Son cargas de prueba implementados sin hacer uso de los componentes de la carga real.
- **Carga ejecutable**: Carga que puede ser directamente ejecutada en un sistema real.
- **Carga no ejecutable**: Se describe por un conjunto de valores medios de parámetros (tiempo de llegadas entre peticiones, demanda de servicios, mezcla de peticiones) que reproduce la utilización de los recursos de la carga real. No es adecuada para la ejecución en un sistema real. Es adecuada para modelos analíticos y de simulación.

Carga de Prueba Real

- Carga que se está procesando en el sistema
- Es la más barata y, potencialmente, la más representativa.
- Modelo en el sentido que la duración de su ejecución es normalmente más corta que la carga real que representa.
- Hay que decidir qué porción del tiempo es el que debe usarse en los experimentos de medición.
- Hipótesis de carga estacionaria.
- Problema principal la reproducibilidad de situaciones.
- Falta de flexibilidad: imposibilidad de modificar programas y consumos de recursos.
- Confidencialidad.
- Reutilización de datos originales.

Carga de Prueba Sintética

- Naturales. (benchmark)
  - Consta de un conjunto de programas extraídos de la carga real.
  - Deben definirse descripciones detalladas, al menos, de los objetivos.
    - las reglas operativas para su ejecución.
    - los resultados a presentar
    - mediciones a recoger en cada ejecución.
- Híbridas
  - Cuando la carga que se pretende modelar no existe completamente, se representa la carga conocida por un conjunto de programas extraídos de ella y la no existente mediante algún elemento artificial.

Carga de Prueba Artificial No-ejecutable

- Modelo artificial de una carga
- Consta de componentes diseñados con el propósito de cargar el sistema real o un modelo matemático de él.
- Ningún componente de la carga real del sistema forma parte de una carga de prueba artificial.
- Tipos
  - Instrucción de suma
  - Mix de instrucciones o sentencias
  - Kernels
  - Programas sintéticos
  - *Benchmarks*

Carga de Prueba Artificial Ejecutable

- Modelo artificial de una carga
- Se usa para el análisis analítico
- Empleado cuando la carga es difícil de modelar o cuando la construcción del modelo es costosa.
- Tipos:
  - Modelos analíticos
  - Modelos dirigidos por distribución (*distribution-driven models*).
  - Redes de Petri*

Tipos de carga: Instrucción de suma (I)

- Instrucción de suma
- Los primeros estudios de evaluación se centraban en la comparación de los tiempos de suma.
- Con la evolución de los ordenadores la consideración de potencia de los mismos cambió y se añadió al tiempo de suma, el tiempo de acceso a memoria y el tiempo de ciclo de CPU.
- Los resultados solamente proporcionan la potencia de computación en bruto. No es una buena métrica para comparar máquinas ya que no todas las máquinas son iguales, diferentes arquitecturas.
- Las máquinas utilizan diferentes SO, conjuntos de instrucciones .
- El rendimiento actual varía con el software y los periféricos.
- La potencia de procesamiento de la información es difícil de definir y cuantificar de forma satisfactoria.

Tipos de carga: Instrucción de suma (II)

- Ejemplos de definición de potencia para comparación de máquinas:
  - $P = M / t_{ciclo}$
  - $P = M / (t_{suma} + t_{multiplicación})$

  donde *M* es el tamaño en palabras de la memoria principal t_{ciclo}, t_{suma} y $t_{multiplicación}$ son los tiempos de ciclo, suma y multiplicación.
- Son ecuaciones orientadas al hardware e independientes de la carga.

Tipos de carga:
Mix (mezcla) de instrucciones y de sentencias

- Parte del análisis de frecuencias de ejecución de instrucciones máquina en una carga real. Solamente se consideran cierto tipo de instrucciones
- Es una especificación de varias instrucciones junto con su frecuencia de utilización.
- Puede constar de un único programa o de varios programas cuyas frecuencias de ejecución de instrucciones coincidan con las de la carga total que se está modelando.
- Fue uno de los primeros modelos de carga artificial propuestos en evaluación de prestaciones. Depende de la arquitectura de la CPU.

Tipos de carga:
Mix (mezcla) de instrucciones y de sentencias

- En algunos casos, dependiendo de los objetivos del estudio, se pueden asignar pesos para caracterizar mejor ciertas aplicaciones
  - En aplicaciones científicas se valoran más las instrucciones de cálculo que las de E/S.
  - Solamente miden la velocidad del procesador.
- Se calcula para cada tipo de instrucción j, el tiempo de ejecución de la instrucción t_j cuya frecuencia relativa se denota por f_j .
- El tº medio de ejecución de ese mix en el sistema sería:

$$t = \sum_{i=1}^{n} t_i f_i$$

Tipos de carga:
Mix (mezcla) de instrucciones y de sentencias

- Cuando se comparen dos o mas arquitecturas de CPU el mix de instrucciones ha de construirse tan independiente del sistema como sea posible.
  - Una de las primeras y más famosas es el Gibson-mix.
    - Desarrollada en 1959 para sistemas IBM 704.
    - Identifica 13 clases de instrucciones
    - Las frecuencias se obtuvieron de medidas de instalaciones científicas y técnicas soportadas por sistemas IBM 704 y 650.
- Mix de sentencias
  - Extensión a los lenguajes de alto nivel del concepto de mix de instrucciones.
  - Se construyen programas que tengan la misma frecuencia de aparición de las distintas sentencias del lenguaje de alto nivel considerado que la carga que se está modelando.
  - Es dependiente del compilador

Tipos de carga:
Gibson Mix

Load and Store	13.2
Fixed-Point Add/Sub	6.1
Compares	3.8
Branches	16.6
Float Add/Sub	6.9
Float Multiply	3.8
Float Divide	1.5
Fixed-Point Multiply	0.6
Fixed-Point Divide	0.2
Shifting	4.4
Logical And/Or	1.6
Instructions not using regs	5.3
Indexing	18.0
Total	100

Tipos de carga:
Problemas con los Mix de instrucciones

- En sistemas modernos el tiempo de ejecucíón de las instrucciones depende de varios factores.
  - Modos de direccionamiento
  - Tasa de aciertos de caché
  - Pipelining (Segmentación)
  - Interferencia de otros dispositivos con el procesador durante el acceso a la memoria.
  - Distribución de ceros en el multiplicador.
  - Número de veces que un salto condicional es realizado.
- Los Mix de instrucciones no reflejan características especiales de hardware tales como el paginado basado en tablas (*page table lookup*)
- Solo representan el rendimiento (velocidad) del procesador.
  - Por ejemplo, si se buscan cuellos de botella, estos pueden estar en otras partes del sistema.

Tipos de carga:
Kernel

- Es un programa o fragmento de programa que representa lo más característico de la carga.
- Son programas cerrados, consumo de recursos predeterminados y no parametrizables. Proporcionados por asociaciones de usuarios y agencias.
- Utilizados en las primeras etapas del diseño de CPU.
- Generalización de los mix de instrucciones, se considera un conjunto de funciones en lugar de un conjunto de instrucciones.
- En su forma ejecutables suelen ser programas de tamaño pequeño.

Tipos de carga: Kernel

- Basados en funciones específicas
  - Cribado (Sieve)
  - Función de Ackerman
  - Inversión de matrices
  - Búsqueda en árboles.
- Problemas
  - Algunos problemas como los ceros y las bifurcaciones desvían las mediciones obtenidas.
  - No están basados en medidas actuales del sistema.
  - Generalmente no consideran la E/S

Tipos de carga: Programas sintéticos

- No realizan ningún trabajo útil, se limitan a consumir recursos del sistema.
- Es un programa que se codifica y ejecuta, difiere de los *benchmarks* en que no se parece a la carga real. Combina los atributos de los *kernel* y de los *benchmark*.
  - Añaden requerimientos de E/S a la carga de prueba
  - Añaden ciclos de control parametrizados para realizar las operaciones deseadas de acuerdo a requerimiento.
  - Fáciles de distribuir
  - Pueden contener monitores u otras herramientas de medición incluidas en el mismo código.
- Efectúa demandas de los diferentes recursos del sistema permitiendo que se mida el rendimiento de dicho sistema.

Tipos de carga: Programas sintéticos

- El consumo de los recursos del sistema es función de los valores de los parámetros que puede elegir el usuario.
  - Tiempo total de CPU
  - Nº de operaciones de E/S
  - Nº de registros de entrada
  - Nº de líneas a imprimir
  - Nº y características de los archivos o bases de datos a los que hay que acceder.
  - Nº de accesos lógicos a los archivos o bases de datos
  - Servicios de SO: creación de procesos, reserva de memoria
- A pesar de eso no realizan un uso representativo de la memoria o del acceso a discos.
- También suelen ser pequeños, por lo que no comprueban el rendimiento de la memoria visual.

Tipos de carga: Programas sintéticos: Ejemplo

Buckholz, 1969

```
      DIMENSION Record(500)
!Control parameters:
      Num_Computes=500        !Repeat count for computation
      Num_Reads=35            !Number of records read
      Num_Writes=40           !Number of records written
      Num_Iterations=1000     !Repeat count for the experiment
!Open files:
      OPEN(UNIT=1,NAME='In.dat',TYPE='Old',
     1FORM='Unformatted',ACCESS='Direct')
      OPEN(UNIT=2,NAME='Out.dat',TYPE='New',
     1FORM='unformatted',ACCESS='Direct')
      CALL Get_Time(CPU1,Elapsed1)    !Record starting time

      DO 500 Iteration=1,Num_Iterations

!Perform a number of read I/Os
      DO 100 i=1,Num_Reads
      READ(1'i),Record
100   CONTINUE
!Do computation
      DO 200 j=1,Num_Computes
      DO 200 i=1,500
200   Record(i)=1 + i + i*i + i*i*i
!Perform a number of write I/Os
      DO 300 i=1,Num_Writes
      WRITE(2'i),Record
300   CONTINUE
500   CONTINUE
      CALL Get_Time(CPU2,Elapsed2)    !Get ending time
!Close files:
      CLOSE(UNIT=1)
      CLOSE(UNIT=2)
      CPU_Time=(CPU2-CPU1)/Num_Iterations
      Elapsed_Time=(Elapsed2-Elapsed1)/Num_Iterations
      TYPE *,'CPU time per iteration is ',CPU_Time
      TYPE *,'Elasped time per iteration is ',Elapsed_Time
      STOP
      END
```

Script en el sistema M5

- M5 es una plataforma de simulación de arquitecturas de sistemas computacionales.
- Permite la simulación a nivel de arquitectura de sistema y a nivel de microarquitectura de procesador

```
my_cpu = SimpleCPU(clock = '2GHz', width = 2)
my_cpu.icache = BaseCache(size = '32KB', assoc = 2)
my_cpu.dcache = BaseCache(size = '64KB', assoc = 2)
my_system = LinuxSystem(cpu = my_cpu)
root = Root(system = my_system)
:
import m5
m5.instantiate(root)
exit_event = m5.simulate()
print 'Exiting @ cycle', m5.curTick(), 'because',
   exit_event.getCause()
```


Tipos de carga: Script sintéticos (Application workloads)

- Variación de los programas sintéticos para entornos interactivos.
- Secuencia de comandos interactivos separados por tiempos fijos de pausa/revisión/planeación humana.
- Se supone que representa una sesión típica de un usuario frente a una terminal.
- Se puede establecer una colección de scripts que modelen un número de usuarios que acceden simultáneamente al sistema interactivo.
- Se aplican para el estudio de sistemas de propósito específico:
  - Reservación de billetes aéreos
  - Transacciones bancarias
- Hacen uso de todo el sistema (CPU, E/S, Memoria, Red)

Tipos de carga:
Script sintéticos (Application workloads)

- Ejemplo:

Operación	Comentarios
El usuario se conecta	-
Creación de Programa	Se editan los errores de tecleo, se introducen errores sintácticos.
Compilación de Programa	Se diagnostican errores sintácticos.
Debugging	Se eliminan los errores sintácticos.
Compilación	La compilación finaliza bien.
Ejecución del programa	Se ejecuta correctamente el programa y se imprimen los resultados.
Listado del programa	Se imprime el código fuente del programa.
Listado de ficheros	Se imprime el listado de los ficheros asociados al programa.
Borrado del programa	-
Usuario sale de sesión.	-

Tipos de carga:
Diferencias entre programas sintéticos y kernels

- Los programas sintéticos incorporan un número de tareas y cubren un rango más amplio de tareas que los programas kernel (estos últimos no tienen en consideración los dispositivos de E/S y los servicios de SO).
- Los programas sintéticos son más representativos de la carga real que los programas kernel.
- Los programas sintéticos son programas modificables mediante parámetros para ajustar el consumo de recursos. Los programas kernel no son modificables.
- La mayoría de los programas paramétricos incorporan facilidades de medida: una vez desarrollados, el proceso de medida es automático y puede repetirse de forma sencilla en los diferentes SO para caracterizar las ganancias o pérdidas de rendimiento.

Benchmarks

- El término *benchmark* se usa frecuentemente como si fuese un sinónimo de *Carga de Prueba*.
- En general, un benchmark es un programa o conjunto de programas representativos bien definidos que se ejecuta en diferentes sistemas y redes para evaluar o comparar rendimientos.
- Dos definiciones
  - Forma de evaluar las prestaciones de un sistema informático.
  - Conjunto de programas reales, escritos en lenguajes de alto nivel, que representan una carga genérica.
- El *benchmarking* es el proceso de comparación de rendimiento para dos o más sistemas mediante la obtención de medidas.
- El *benchmarking* hace referencia a la ejecución de un conjunto de programas representativos en diferentes sistemas y la medida de los resultados.

Benchmarking

- Características de un *benchmark* útil:
  - Relevante: tiene que proporcionar medidas significativas del rendimiento para un dominio de problema específico.
  - Comprensible: Los resultados del *benchmark* deben ser simples y fáciles de entender.
  - Escalable: las pruebas tienen que ser aplicables a un rango amplio de sistemas en términos de costes, rendimiento y configuración.
  - Aceptable: debe presentar resultados no sesgados y reconocidos por usuarios y vendedores.
- Evitar problemas
  - Comprender el entorno del *benchmark*
  - Para interpretar los resultados hay que entender:
    - Las características de la carga
    - El sistema en estudio
    - Las pruebas realizadas
    - Las medidas observadas
    - Los datos experimentales obtenidos

Benchmarking

- Objetivos
  - Comparación de ordenadores, periféricos y redes
  - Ajuste y planificación de carga
- Los más conocidos los realizados por empresas o instituciones independientes.
  - SPEC (Standard Performance Evaluation Corporation).
    http://www.spec.org
  - TPC (Transaction Processing Performance Council).
    http://www.tpc.org/
  - Mindcraft:Webstone
    http://www.mindcraft.com/webstone
  - Benchmarking Linux – COMO (How to)
    http://es.tldp.org/COMO-INSFLUG/es/pdf/Benchmarking-COMO.pdf
- Dirección interesante
  - http://www.benchmarkresources.com/

Benchmarks sintéticos

- Operaciones básicas. Utilidad limitada.
  - Criba (Sieve)
    - Basado en la criba de Eratóstenes para calcular números primos
    - Comparación de microprocesadores, ordenadores personales y lenguajes de alto nivel.
  - Función de Ackermann
    - Escrito de forma recursiva
  - Evaluar la facilidad que tiene un lenguaje para realizar llamadas recursivas a procedimientos.
  - Compara el tiempo medio de ejecución, el nº de instrucciones, espacio de pila requerido por cada llamada.
  - Whetstone
    - Desarrollado por la British Central Computer Agency
    - Evalúa la velocidad de punto flotante.
  - Dhrystone
    - Desarrollado en Siemens
    - Para representar entornos de sistemas de programación
    - Velocidad de cómputos de punto fijo

Benchmarks: Kernels

- Benchmarks. Kernels
  - Utilizados generalmente para evaluar rendimiento de CPU
  - No sirven para medir el rendimiento percibido por los usuarios
- Linpack
  - Programa de álgebra lineal creado por Dongarra en 1976.
  - Multiplicación de matrices. Rutinas BLAS
  - Sensible a las operaciones de punto flotante.
  - Evaluar sistemas científicos y de ingeniería.
  - Existe una base de datos de prestaciones (PDB)
    - http://www.netlib.org/performance/html/PDSreports.html
- Livermore Kernels (Livermore Loops)
  - Introducido en 1970 para supercomputadores.
  - Mide computaciones numéricas.
  - Operaciones de punto flotante
  - También mide la precisión computacional.

Benchmarks: Programas Reales

- Se utilizan para obtener una visión más precisa del rendimiento del sistema que percibe el usuario
- Estos benchmarks se pueden agrupar en dos categorías:
  - Nivel componente: un conjunto de pruebas y cargas de trabajo especificados para medir el rendimiento de componentes o subsistemas tales como velocidad de CPU, tiempo E/S, productividad del servidor de ficheros, etc.
    - **Caracterización de la CPU**. Los mas conocidos las suites de SPEC
    - **Servidores de ficheros**. Modela cargas de trabajo de varios tipos de entradas en los servidores de ficheros.
  - Nivel de sistema: considera el sistema entero – mide el procesador, el subsistema de E/S, la red, la base de datos, el compilador y el SO.
    - **Debit-credit benchmark**: Comparación de sistemas de procesamiento de transacciones.
    - **TPC**. Miden prestaciones de sistemas completos cliente/servidor

Tipos de carga: Benchmarks-Función de Ackerman

- Ackermann, Wilhelm: *Zum Hilbertschen Aufbau der reelen Zahlen. Math. Annalen* v. 99 (1928), pp. 118-133.

$$A(m,n) = \begin{cases} n+1 & \text{si } m=0; \\ A(m-1,1) & \text{si } m>0 \text{ y } n = 0; \\ A(m-1, A(m,n-1)) & \text{si } m>0 \text{ y } n > 0; \end{cases}$$

- En 1928, Wilhelm Ackermann consideró una función $A(m, n, p)$ de tres variables: $m \rightarrow n \rightarrow p$ en la notación de Conway. Ackermann demostró que se trata de una función recursiva que no es primitiva recursiva.
- La función diagonal f (n) = A(n, n) crece muy rápidamente en tanto que su inversa crece muy lentamente, por ello se utiliza frecuentemente en análisis de algoritmos.

Tipos de carga: Benchmarks-Función de Ackerman

- La función de Ackermann se utiliza con frecuencia para comparar compiladores en cuanto a su habilidad para optimizar la recursión.
  - Un compilador capaz de notar que $A(3, 30)$ se puede calcular basándose en potencias de 2, o que guarda resultados intermediarios tales como $A(3, n)$ y $A(2, n)$ en lugar de recalcularlos cada vez, ahorraría tiempo de ejecución por un factor de 100 o 100
  - Igualmente, al calcular directamente $A(1, n)$ en lugar de hacer una llamada recursiva se realizan ahorros significativos.
  - Es posible calcular el término $A(4, 2)$ pero no recursivamente, sino por otros medios.

Tipos de carga: Benchmarks-Función de Ackerman-C

```
int ackerman(int m, int n) {
  int temp;
  if(m == 0)
    temp = n + 1;
  else {
    if (n == 0)
      temp = ackerman(m-1, 1);
    else
      temp = ackerman(m-1, ackerman(m, n-1));
  }
  return(temp);
}
```


SPEC

- SPEC (Standard Performance Evaluation Corporation), una organización de compañías informáticas que ha desarrollado un conjunto de benchmarks (SPEC Benchmark suites) elaborados a partir de aplicaciones de ingeniería y científicas.
- SPEC no solamente desarrolla benchmarks, también publica los resultados de rendimientos de: CPU, servidores de ficheros, servidores web y gráficos.
- Consiste en tres grupos diferentes
  - OSG (Open Systems Group) Es el comité original de SPEC. Se centra en benchmaks para systemas de sobremesa, estaciones de trabajo y servidores que ejecutan sistemas abiertos.
  - HPC (High Performance Group) mide prestaciones de ordenadores dedicados a cálculo intensivo. Arquitectura de sistemas de alto rendimiento: sistemas multiprocesador de simétricos, cluster de estaciones de trabajo, sistemas paralelos de memoria distribuida y supercomputadores vectoriales y vectoriales paralelos.
  - GPC (Graphics Performance Group) que mide prestaciones de subsistemas gráficos.
- Una visión general de los benchmark de Spec se encuentra en:
  - http://www.benchmarkresources.com/handbook/chapter9.pd

SPEC OSG subcomités.

- CPU: SPECmarks y bechmarks de CPU (SPECint, SPECfp, SPECrates, etc)
  - SPECint – Rendimiento de cálculo intensivo de enteros.
  - SPECfp – Rendimiento de cálculo intensivo de punto flotante.
- JAVA: Benchmarks para clientes y servidores Java. JVM98 y JBB2000.
- MAIL: Benchmarks servidores de correo de ISP. SPECmail2001.
- SDM: Benchmarks para comandos UN*X multi usuario. Desarrollo de software en un entorno multitarea. Nivel de sistema.
  - SDET: Entorno de desarrollo comercial en UNIX y C.
  - Kenbus1: Uso de UNIX y C en un entorno de investigación y desarrollo.
- SFS: Benchmarks para servidores de ficheros. SFS93(LADDIS).
  - Mide el rendimiento de los servidores NFS en varios niveles de carga.
  - Se pueden modificar y ajustar los parámetros para obtener un nivel de carga representativo del usuario
  - Genera el throughput y el tiempo medio de respuesta
- WEB: Benchmarks para servidores web. WEB96 y WEB99

SPEC: CPU

- CPU2006
  - CINT2006
  - CFP2006
  - Versión actual CPU2006 v1.0
- CPU2000
  - CINT2000
  - CFP2000
  - Versión descontinuada: CPU2000 v1.3.

SPEC: Graphics/Applications

- SPECviewperf® 9
- SPECviewperf® 8.1
- SPECapcSM for 3ds Max™ 8
  - http://www.extremetech.com/article2/0,1697,2145926,00.asp
- SPECapcSM for 3ds Max™ 7
- SPECapcSM for Maya 6.5
- SPECapcSM for Pro/ENGINEERTM Wildfire 2.0
- SPECapcSM for Solid Edge V14™
- SPECapcSM for SolidWorks 2005
- SPECapcSM for UGS NX 3

SPEC: High Performance Computing, OpenMP, MPI

- HPC2002
  - Aplicaciones de tipo industrial
  - Desarrollado especialmente para evaluar el rendimiento de sistemas paralelos y arquitecturas distribuidas.
- OMP2001
  - Basado en el estándar OpenMP para proceso paralelo de memoria compartida.
    - OMPM2001
    - OMPL2001
  - Contiene conjuntos de trabajo más grandes y con mayores tiempos de ejecución que OMPM2001.
- ¿ MPI2006 ?
  - El SPEC está buscando candidatos para evaluar métricas de rendimiento de interfaces de paso de mensajes, Message Passing Interface (MPI)

SPEC: Java Client/Server

- jAppServer2004
  - Para evaluar servidores de aplicaciones J2EE 1.3.
  - Incluyen cargas ampliadas mediande la adición de un web tier, JMS, y otros cambios a SPECjAppServer2002.
- JBB2005
  - Para evaluar servidores ejecutando aplicaciones de negocio típicas Java.
  - JBB2005 representa a una aplicación de procesamiento de ordenes para un distribuidor de productos al por mayor.
- JVM98
  - Mide la eficiencia de la JVM, el compilador just-in-time (JIT), y las implementaciones en distintos sistemas operativos.
  - En el aspecto de hardware, incluye la evaluación de rendimiento de la CPU (punto flotante y operaciones enteras), cache, memoria y otras medidas de rendimiento típicas de la plataforma.

SPEC: Mail Servers (Servidores de Correo)

- MAIL2001
  - Benchmark estandarizado de servidores de correo electrónico diseñado para medir la habilidad de un sistema para actuar como servidor de correo proporcionando servicios basados en los estándares SMTP y POP3.
  - El benchmark caracteríza el rendimiento y el tiempo de respuesta de un servidor de correo poniéndolo a prueba con cargas de red, almacenamieno y requerimientos de clientes realistas.
- SPECimap
  - Actualmente bajo desarrollo.
  - Esta diseñado para medir el rendimiento de servidores de correo corporativos.
  - diseñado para medir la habilidad de un sistema para requerimientos de correo electrónico basados en los estándares SMTP e IMAP4.

SPEC: Network File System

- SFS97_R1 (3.0)
  - Benchmark diseñado para evaluar la velocidad y capacidades de gestión de requerimientos de servidores NFS.
  - Los servidores están basados en las versiones 2 y 3 del protocolo NFS.

SPEC: Power and Performance

- SPECpower
  - El comité SPEC de rendimiento de energía ha empezado el desarrollo de la primera generación de benchmarks SPEC para evaluar la eficiencia de consumo de energía de ordenadores de clase servidos.
  - La meta de este benchmark es proporcionar un medio para obtener información fiable y consistente sobre el uso de energía por el sistema bajo distintos niveles de trabajo.

SPEC: Web Servers

- WEB2005
  - El SPECweb2005 emula los requerimientos de navegadores de usuarios, los cuales envian requerimientos sobre una conección Internet de banda ancha a un servidor web.
  - Proporciona tres nuevas cargas de trabajo:
    - Un servicio de banca (HTTPS)
    - Un servicio de comercio electrónico (mix HTTP/HTTPS)
    - Un servicio de soporte (HTTP).
  - El contenido dinámico es implementado en PHP y JSP.
  - La versión actual es la 1.10.

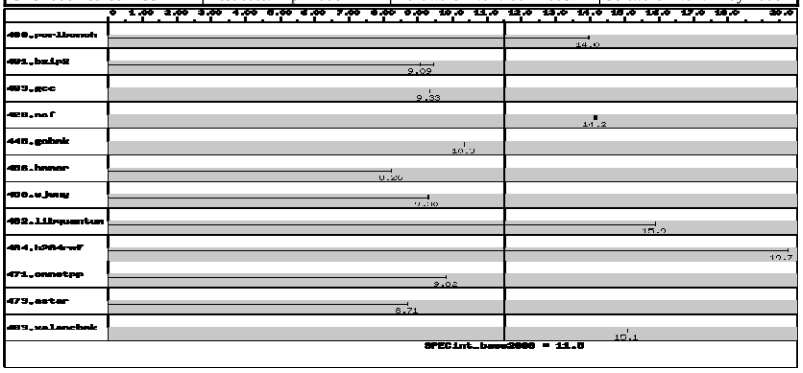
SPEC
Ejemplos – CPU2006 - CINT2006 (32)

Test Sponsor	System Name	Processor				Results	
		Cores	Chips	Cores/ Chip	Threads/ Core	Base	Peak
Advanced Micro Devices	Tyan Thunder K8E Tomcat (S2865) (AMD Opteron 146) HTML | CSV | Text | PDF | PS | Config	1	1	1	1	9.02	--
Advanced Micro Devices	Shuttle SN25P (AMD Athlon64 FX60) HTML | CSV | Text | PDF | PS | Config	2	1	2	1	10.1	--
Apple	iMac 20-inch 2.0GHz Intel Core Duo HTML | CSV | Text | PDF | PS | Config	2	1	2	1	10.1	10.1
Dell	Precision 380 Workstation (3.73 GHz, Pentium Exteme Edition 965) HTML | CSV | Text | PDF | PS | Config	2	1	2	1	11.6	--
Dell	Precision 380 Workstation (3.8 GHz, Pentium 4 processor 670) HTML | CSV | Text | PDF | PS | Config	1	1	1	1	11.5	--
Fujitsu Siemens Computers	CELSIUS M440 Pentium 4 670 HTML | CSV | Text | PDF | PS | Config	1	1	1	1	11.6	12.3
Fujitsu Siemens Computers	CELSIUS V830, Opteron 256, Linux 64-bit HTML | CSV | Text | PDF | PS | Config	1	1	1	1	11.9	13.3
Hewlett-Packard Company	ProLiant DL380 G4 (3.8GHz, Intel Xeon Processor) HTML | CSV | Text | PDF | PS | Config	2	2	1	2	11.4	--
Hewlett-Packard Company	ProLiant DL385 (AMD Opteron 254) HTML | CSV | Text | PDF | PS | Config	1	1	1	1	11.1	12.4
Hewlett-Packard Company	ProLiant DL585 (AMD Opteron 854) HTML | CSV | Text | PDF | PS | Config	2	2	1	1	11.2	12.7
Hewlett-Packard Company	HP Integrity rx6600 (1.6GHz/24MB Dual-Core Intel Itanium 2) HTML | CSV | Text | PDF | PS | Config	2	1	2	1	14.5	15.7
Hewlett-Packard Company	HP Integrity rx2620 (1.6GHz/18MB Dual-Core Intel Itanium 2) HTML | CSV | Text | PDF | PS | Config	2	1	2	1	13.4	14.5
Hewlett-Packard Company	HP Integrity rx4640 (1.6GHz/24MB Dual-Core Intel Itanium 2) HTML | CSV | Text | PDF | PS | Config	2	1	2	1	13.8	15.1

SPEC
Ejemplos - Dell

SPEC® CINT2006 Result

Dell, Inc. Precision 380 Workstation 3.8 GHz, Pentium 4 Processor 670)	SPECint®2006 = -- SPECint_base2006 = 11.5
Test Sponsor: Dell	Tested by: Dell
CPU2006 license # 55	Test date: Apr-2006
Hardware Avail: Jun-2005	Software Avail: May-2006



SPEC
Ejemplos - Dell

Hardware		Software	
CPU Name:	Intel Pentium 4 670	Operating System:	Windows XP Professional x64 Edition
CPU Characteristics:		Compiler:	Intel C++ Compiler 9.1 for IA32 (20060323Z) Microsoft Visual Studio .NET 2003 MicroQuill SmartHeap Library 7.4
CPU MHz:	3800	Auto Parallel:	No
FPU:	Integrated	File System:	NTFS
CPU(s) enabled:	1 core, 1 chip, 1 core/chip	System State:	Default
CPU(s) orderable:	1 chip	Base Pointers:	32-bit
Primary Cache:	12 K micro-ops I + 16 KB D on chip per chip	Peak Pointers:	Not Applicable
Secondary Cache:	2 MB I+D on chip per chip	Other Software:	None
L3 Cache:	None		
Other Cache:	None		
Memory:	2 GB (4x512 MB 533MHz CL4 DDR2 SDRAM)		
Disk Subsystem:	1 x 80GB SATA 7200 RPM		
Other Hardware:	None		

SPEC
Ejemplos - Dell

Benchmark	Base						Peak					
	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio
400.perlbench	699	14.0	700	14.0	**700**	**14.0**						
401.bzip2	1022	9.44	1063	9.08	**1062**	**9.09**						
403.gcc	862	9.33	863	9.33	**863**	**9.33**						
429.mcf	646	14.1	643	14.2	**644**	**14.2**						
445.gobmk	1014	10.3	1014	10.3	**1014**	**10.3**						
456.hmmer	1130	8.26	1130	8.25	**1130**	**8.26**						
458.sjeng	**1302**	**9.30**	1301	9.30	1302	9.29						
462.libquantum	1304	15.9	**1305**	**15.9**	1305	15.9						
464.h264ref	1121	19.7	1121	19.7	**1121**	**19.7**						
471.omnetpp	636	9.83	**636**	**9.82**	636	9.82						
473.astar	**806**	**8.71**	805	8.72	806	8.71						
483.xalancbmk	458	15.1	458	15.1	**458**	**15.1**						

Results appear in the order in which they were run. Bold underlined text indicates a median measurement.

SPEC
Ejemplos - Dell

Base Compiler Invocation

C benchmarks:

icl -Qc99 -Qvc7.1

C++ benchmarks:

icl -Qvc7.1

Base Portability Flags

C benchmarks (except as noted below):

No flags used

403.gcc:
-DSPEC_CPU_WIN32

464.h264ref:
-DSPEC_CPU_NO_INTTYPES -DWIN32

C++ benchmarks (except as noted below):

No flags used

473.astar:
-DSPEC_CPU_LITTLE_ENDIAN

SPEC
Ejemplos - Dell

Base Optimization Flags

C benchmarks:

-fast -link /FORCE:MULTIPLE shlw32m.lib /F512000000

C++ benchmarks:

-fast -Qcxx_features -link /FORCE:MULTIPLE shlw32m.lib /F512000000

Base Other Flags

C benchmarks (except as noted below):

No flags used

403.gcc:
-Dalloca=_alloca

C++ benchmarks:

No flags used

The flags file that was used to format this result can be browsed at
http://www.spec.org/cpu2006/flags/Intel-ic91-flags-file-20060622.xml.html

You can also download the XML flags source by saving the following link:
http://www.spec.org/cpu2006/flags/Intel-ic91-flags-file-20060622.xml.xml

SPEC
CPU2006 – CINT2006

- Contiene 12 benchmarks
  - 9 usan C
  - 3 usan C++
- Benchmarks:

400.perlbench	C	PERL Programming Language
401.bzip2	C	Compression
403.gcc	C	C Compiler
429.mcf	C	Combinatorial Optimization
445.gobmk	C	Artificial Intelligence: go
456.hmmer	C	Search Gene Sequence
458.sjeng	C	Artificial Intelligence: chess
462.libquantum	C	Physics: Quantum Computing
464.h264ref	C	Video Compression
471.omnetpp	C++	Discrete Event Simulation
473.astar	C++	Path-finding Algorithms
483.xalancbmk	C++	XML Processing

SPEC
CPU2006 – CFP2006

- Contiene 17 benchmarks
  - 4 usan C++, 3 usan C, 6 usan Fortran, 4 usan una mezcla de C y Fortran
- Benchmarks:

410.bwaves	Fortran	Fluid Dynamics
416.gamess	Fortran	Quantum Chemistry
433.milc	C	Physics: Quantum Chromodynamics
434.zeusmp	Fortran	Physics/CFD
435.gromacs	C/Fortran	Biochemistry/Molecular Dynamics
436.cactusADM	C/Fortran	Physics/General Relativity
437.leslie3d	Fortran	Fluid Dynamics
444.namd	C++	Biology/Molecular Dynamics
447.dealII	C++	Finite Element Analysis
450.soplex	C++	Linear Programming, Optimization
453.povray	C++	Image Ray-tracing
454.calculix	C/Fortran	Structural Mechanics
459.GemsFDTD	Fortran	Computational Electromagnetics
465.tonto	Fortran	Quantum Chemistry
470.lbm	C	Fluid Dynamics
481.wrf	C/Fortran	Weather Prediction
482.sphinx3	C	Speech recognition

SPEC
CPU2006 – Métricas (I)

- Luego de que los benchmarks son ejecutados en el sistema bajo pruebas (system under test - SUT), un índice/ratio es calculado para cada uno usando el tiempo de ejecución sobre el SUT y el tiempo de referencia determinado SPEC.

- A partir de estos índices se calcúlan las siguientes métricas:
  - CINT2006 (para comparaciones de rendimiento intensivas de computación entera):
    - SPECint2006: la media geométrica de doce ratios normalizados – uno por cada benchmark entero- cuando los benchmarks son compilados con configuración punta (peak tuning).
    - SPECint_base2006: la media geométrica de doce ratios normalizados cuando los benchmarks son compilados con configuración base (base tuning).
    - SPECint_rate2006: la media geométrica de doce ratios de productividad normalizados cuando los benchmarks son compilados con configuración punta (peak tuning).
    - SPECint_rate_base2006: la media geométrica de doce ratios de productividad normalizados cuando los benchmarks son compilados con configuración base (peak base).

SPEC
CPU2006 – Métricas (II)

- CFP2006 (para comparaciones de rendimiento intensivas de computación de punto flotante)
  - SPECfp2006: la media geométrica de diecisiete ratios normalizados cuando los benchmarks son compilados con configuración punta (peak tuning).
  - SPECfp_base2006: la media geométrica de diecisiete ratios normalizados cuando los benchmarks son compilados con configuración base (base tuning).
  - SPECfp_rate2006: la media geométrica de diecisiete ratios de productividad normalizados cuando los benchmarks son compilados con configuración punta (peak tuning).
  - SPECfp_rate_base2006: : la media geométrica de diecisiete ratios de productividad normalizados cuando los benchmarks son compilados con configuración base (base tuning).

SPEC
CPU2006 – Máquina de referencia (I)

- SPEC usa una máquina de referencia para normalizar las métricas de rendimiento usadas en los paquetes CPU2006.
  - Cada benchmark es ejecutado y medido en está máquina para estableces un tiempo de referencia para ese benchmark.
  - Estos tiempos son entonces usados en los cálculos SPEC.
- SPEC usa un sistema SUN histórico: el "Ultra Enterprise 2" el cual fué introducido en 1997 como máquina de referencia.
  - Usa un procesador UltraSPARC II a 296 MHz al igual que la suite CPU2000.
  - Pero las máquinas no son idénticas:
    - La máquina de referencia de CPU2006 tiene mejores prestaciones de cache.
    - La máquina de referencia CPU2000 podría no tener suficiente memoria para ejecutar CPU2006.

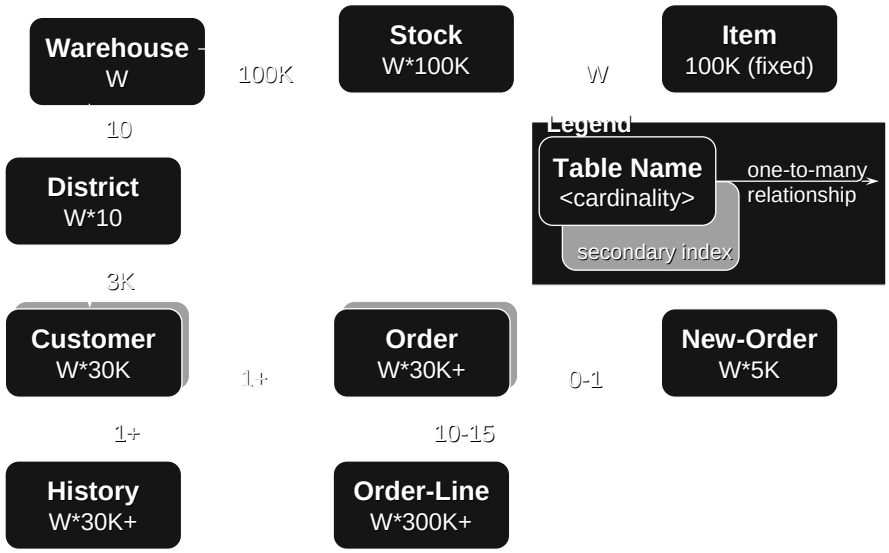
SPEC
CPU2006 – Máquina de referencia (II)

- Toma cerca de 12 días establecer una ejecución “en regla” de las métricas base para CINT2006 y CFP2006 en la máquina de referencia CPU2006.
- Nótese que cuando se comparan dos sistemas medidos con CPU2006, el desempeño relativo de cada uno debe mantenerse aún si se usa una máquina de referencia diferente a la oficial.
  - Esto es una consecuencia del uso del tipo de estadísticas involucradas en el cálculo individual y general de las medias y ratios de productividad de los diecisiete benchmarks (media geométrica).

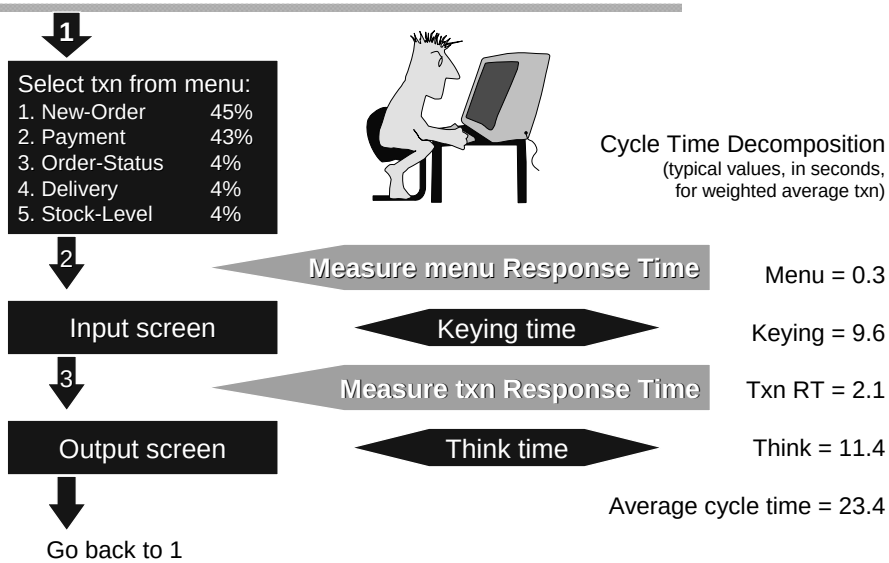
Nivel de sistema: TPC

- TPC (Transaction Processing Performance Council) es una organización sin ánimo de lucro que define benchmarks orientados a la evaluación del rendimiento y comparación de sistemas de proceso transaccional y bases de datos.
- Mide el procesador, el subsistema de E/S, la red, el sistema operativo, el gestor de base de datos y el monitor de transacciones.
- Tres benchmarks
  - TPC-C : simula un entorno de computación de tipo general.
  - TPC-H : simula un entorno de toma de decisiones. Evalúan la relación precio/rendimiento de un sistema dado ejecutando aplicaciones que soportan la formulación de consultas de negocio que han de resolverse mediante consultas complejas a grandes bases de datos.
  - TPC-App: simula un servidor de aplicaciones y servicions web.

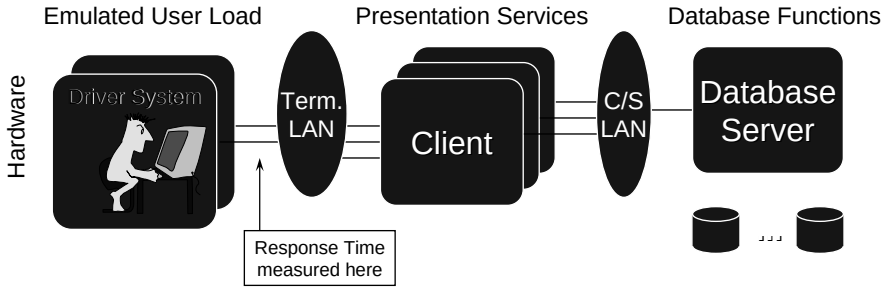
TPC-C: Esquema BD



TPC-C: Workflow



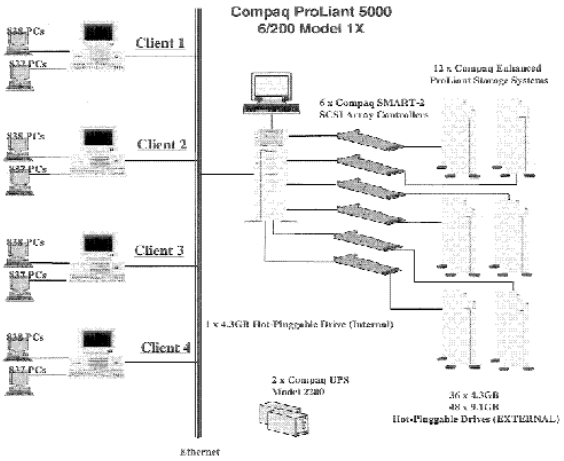
TPC-C: Configuración típica



Software			
	RTE, e.g.: Empower preVue LoadRunner	TPC-C application + Txn Monitor and/or database RPC library e.g., Tuxedo, ODBC	TPC-C application (stored procedures) + Database engine + Txn Monitor e.g., SQL Server, Tuxedo

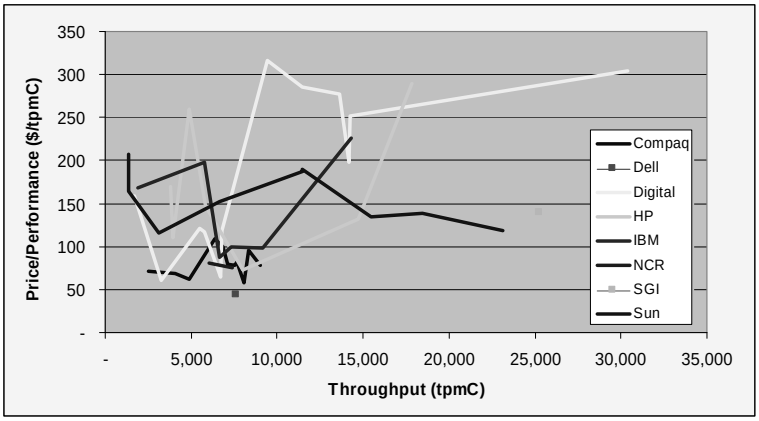
TPC-C: Configuración de referencia

- 8070 tpmC; $57.66/tpmC; 5-yr COO= 465 K$ (Cost of ownership)
- 2 GB memory, disks: 37 x 4GB + 48 x 9.1GB (560 GB total)
- 6,700 users

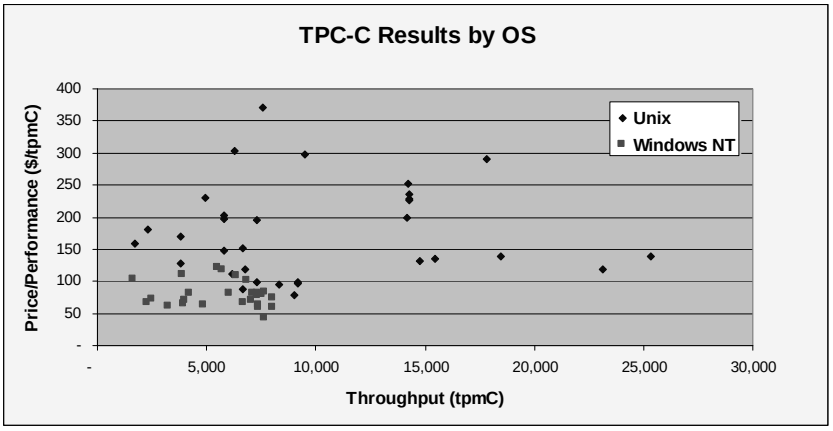


TPC-C: Resultados (1997)

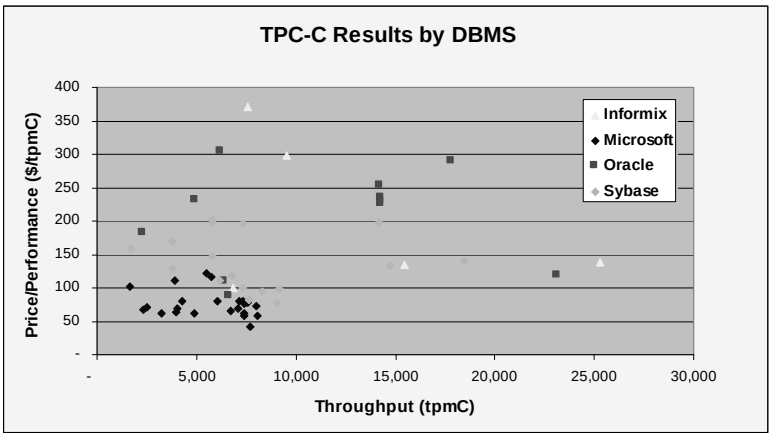
- Best Performance is 30,390 tpmC @ $305/tpmC (Digital)
- Best Price/Perf. is 7,693 tpmC @ $42.53/tpmC (Dell)



TPC-C: Resultados (1997), por S.O.



TPC-C: Resultados (1997), por DBMS



TPC-C: Resultados (2006) Top Ten

Rank	Company	System	tpmC	Price/tpmC	System Availability	Database	Operating System	TP Monitor	Date Submitted	Cluster
1	IBM	IBM System p5 595	4,033,378	2.97 US $	12/20/06	IBM DB2 9	IBM AIX 5L V5.3	Microsoft COM+	08/08/06	N
2	IBM	IBM eServer p5 595	3,210,540	5.07 US $	05/14/05	IBM DB2 UDB 8.2	IBM AIX 5L V5.3	Microsoft COM+	11/18/04	N
3	IBM	IBM eServer p5 595	1,601,784	5.05 US $	04/20/05	Oracle Database 10g Enterprise Edition	IBM AIX 5L V5.3	Microsoft COM+	04/20/05	N
4	HP invent	HP Integrity Superdome – Itanium2/1.6 GHz-64p/64c	1,231,433	4.82 US $	06/05/06	Microsoft SQL Server 2005 Enterprise Edt SP1	Microsoft Windows Server 2003 Datacenter Ed.(64-bit)SP1	Microsoft COM+	11/28/05	N
5	HP invent	HP Integrity rx5670 Cluster-Itanium2/1.5 GHz-64p/6	1,184,893	5.52 US $	04/30/04	Oracle Database 10g Enterprise Edition	Red Hat Enterprise Linux AS 3	BEA Tuxedo 8.1	12/08/03	Y
6	IBM	IBM eServer pSeries 690 Model 7040-681	1,025,486	5.43 US $	08/16/04	IBM DB2 UDB 8.1	IBM AIX 5L V5.2	Microsoft COM+	02/17/04	N
7	IBM	IBM System p5 570 Model 9117-570	1,025,169	4.42 US $	05/31/06	IBM DB2 UDB 8.2	IBM AIX 5L V5.3	Microsoft COM+	02/14/06	N
8	HP invent	HP Integrity Superdome – Itanium2/1.5 GHz-64p/64c	1,008,144	8.33 US $	04/14/04	Oracle Database 10g Enterprise Edition	HP UX 11.iv2 64-Bit Base OS	BEA Tuxedo 8.0	11/04/03	N
9	IBM	IBM eServer p5 570	809,144	4.95 US $	09/30/04	IBM DB2 UDB 8.1	IBM AIX 5L V5.3	Microsoft COM+	07/12/04	N
10	FUJITSU	PRIMEQUEST 480 c/s	792,101	8.92 US $	10/26/06	Oracle Database 10g Enterprise Edition	Red Hat Enterprise Linux 4 AS	BEA Tuxedo 8.1	04/26/06	N

Selección de la Carga de Trabajo

- Recordar:
  - El término caracterización de la carga hace referencia al proceso de construir un modelo que describa cuantitativa de las características de la carga que se ejecuta en un sistema.
  - Hemos visto los principales aspectos asociados a la caracterización de la carga.
    - Visión general de las características de los modelos y las fases de construcción.
    - Representatividad de los modelos.
    - Tipos de modelos de carga
      - Sintética (Natural).
      - Artificial
  - A continuación abordaremos el proceso y las técnicas necesarios para caracterizar la carga.

Selección de la Carga de Trabajo

- Se puede llegar a conclusiones erróneas si la carga de trabajo, sobre la que se realizará el estudio de rendimiento, no se selecciona adecuadamente.
- Consideraciones a tener en cuenta
  - Servicios prestados por la carga
  - Nivel de detalle
  - Representatividad
  - Oportunidad
- Otras consideraciones
  - Nivel de carga
  - Impacto de los componentes externos
  - Repetición

Selección: Servicios prestados.

- Sistema = proveedor de servicios.
- Dos criterios para la identificación y selección de los componentes de la carga.
- Punto de vista estructural
  - System Under Test (SUT) conjunto completo de componentes.
  - Component Under Study (CUS): Componente del SUT considerado en el estudio.
    - Estudio del comportamiento de la ALU (CUS) en la CPU (SUT)
    - Comparación de sistemas de discos (CUS) en un sistema de procesamiento de transacciones (SUT).
  - La carga y las métricas de rendimiento se determinan en primer lugar en relación al SUT.
  - La selección de la carga se hace en relación al sistema (SUT)
  - La consideración de los servicios prestados ha de tener en cuenta el propósito del estudio.

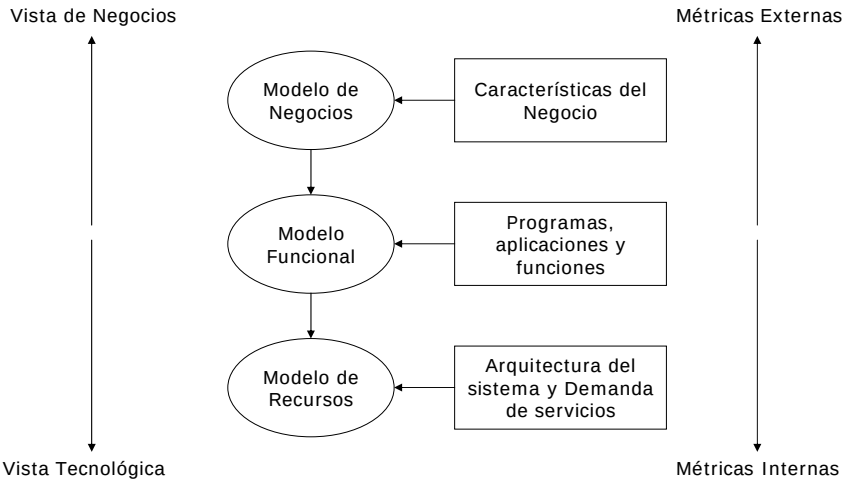
Selección: Servicios prestados. Ejemplo

Aplicaciones Transacciones	Objetivo: Comparar dos sistemas de transacciones. Servicio proporcionado: peticiones de aplicación. Workload: Frecuencia de las diferentes transacciones, transacción más frecuente
Sistema Operativo Servicios+ comandos S.O.	Objetivo: Comparar los servicios máquina S.O. Servicio proporcionado: Servicios de sistema. Workload: Programas sintéticos.
CPU Instrucciones	Objetivo: Comparar dos CPU Servicio proporcionado: Conjunto de instrucciones Workload: Mix de instrucciones. Kernel.
ALU Instrucciones Aritméticas	Objetivo: Comparar dos ALU Servicio solicitado: Instrucciones Aritméticas Workload: Frecuencia de instrucciones arit. La instrucción arit. más frecuente.

Selección: Servicios prestados (II)

- Punto de vista de la naturaleza del servicio prestado
  - Identificación de componentes básico de una carga
    - unidad genérica de trabajo que llega a un sistema desde una fuente externa.
  - Ejemplo
    - Entorno C/S: Petición de cliente o transacción de la base de datos.
- Niveles de descripción
  - Negocio
    - Descripciones orientadas al usuario: nº de empleados, transacciones por usuario, elementos en el catálogo de productos, facturas por cliente.
  - Funcional
    - Descripciones de programas, comandos y peticiones: Web request.
    - Independiente de sistema.
  - Recursos
    - Describe los consumos de recursos del sistema por la carga. Se incluirán solamente los recursos cuyo consumo tenga un impacto significativo en la carga
- Construcción de modelos multicapa.

Niveles de descripción de carga

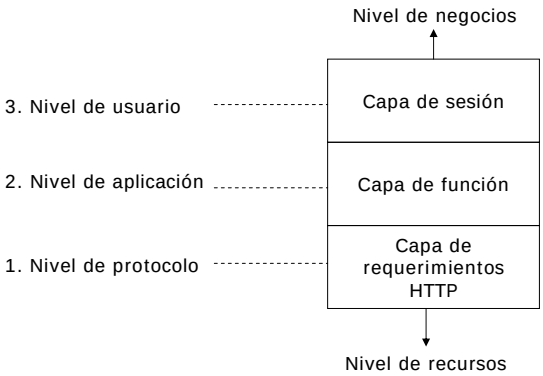


[Menascé 2002]

Selección: Servicios prestados (II).Ejemplo

- Librería en línea [Menascé02, p.212]
  - Clientes pueden realizar las siguientes funciones:
    - Buscar, Navegar, Seleccionar, Registrarse, Login, Añadir, Pagar
  - Clientes interaccionan con el sitio mediante una serie de peticiones
  - consecutivas y relacionadas realizadas en una sesión.
  - ¿Cuál es la duración media de la sesión?
  - ¿Qué funciones son las más utilizadas por los clientes?
  - ¿Porcentaje de imágenes en la carga de trabajo?
  - Modelo multicapa
    - Carga de e-business: se componen de sesiones
    - Durante una sesión el cliente solicita varias funciones de e-business
    - Una función de e-business genera varias "http request"
      - Hay que recuperar varias imágenes para mostrarlas en la página consecuencia de la ejecución de una función

Un modelo de carga jerárquico



[Menascé 2002]

Selección: Nivel de detalle

- Elegir el nivel de detalle de recogida (y posteriormente reproducción) de peticiones para los servicios identificados.
- Lista de posibilidades
  - Petición más frecuente
    - Nivel menos detallado.
    - Puede constituir una aproximación inicial a la carga.
  - Frecuencia de los tipos de petición.
    - Listar varios servicios, sus características y frecuencias.
  - Secuencia de peticiones con registro de tiempo.
    - Traza de las peticiones al sistema real.
    - Demasiado detallada. Más adecuada para simulación que para modelado analítico.
  - Demanda media de recursos.
    - Apropiada para el modelado analítico.
  - Distribución de las demandas de recursos.
    - Demandas agrupadas por clases con servicios similares.

Selección: Otras consideraciones

- Oportunidad.
  - Hay que tener en cuenta que el patrón de comportamiento de los usuarios cambia.
  - La carga de trabajo ha de representar el patrón de uso actual de los sistemas.
- Nivel de carga: Una carga puede llevar a un sistema hasta:
  - Su capacidad máxima (mejor caso)
  - Por encima de su capacidad (peor caso)
  - Nivel de carga observado en la carga real (caso típico).
  - De acuerdo al tipo de estudio hay que elegir un nivel u otro.
- Impacto de los componentes externos.
  - Ignorar este factor puede llevar a conclusiones erróneas.
  - Ej: La influencia del rendimiento de E/S en el tiempo de finalización (total) de un programa **sintético** no hace adecuada esta carga para comparar CPU.