INTRODUCCIÓN

Todo SI tiene dos tipos de requerimientos → funcionales (comp. lógico del sistema, qué hace el sistema) / no funcionales (comp. físico del sistema, cómo opera el sistema)

En los sist. distribuidos, los req. funcionales reflejan su diseño de modo similar al caso de sistemas basados en 1 ordenador.
En cambio, los req. no funcionales producen un mayor impacto ~~el~~ en el diseño global del sistema por la mayor complejidad física.

REQUISITOS NO FUNCIONALES

Desde el punto de vista de arquitectura del sistema como aspectos fundamentales en la elaboración del mismo: rendimiento, capacidad, disponibilidad, fiabilidad y mantenibilidad, seguridad, integridad, gestionabilidad, usabilidad, escalabilidad, ...

En todo SI que presente un servicio a usuarios finales, los req. no func. se plantean bajo los Acuerdos de Nivel de ~~Acceso~~ Servicio (Service Level Agreements, SLA). Representan l acuerdo realizado entre el proveedor de un servicio y el suario ~~calificado~~ calificando los parámetros mínimos aceptables. os sistemas deben asegurar los SLA's pero no excederse / invertir superándolos

ENDIMIENTO: Es el atributo de un SI que asegura la correcta isponibilidad temporal de los servicios del sistema.
o es sinónimo de velocidad. El rendimiento incide en la capacidad predecir el comportamiento temporal de un sistema en el mayor mero posible de situaciones.

- Objetivos del rendimiento para ...

  - Latencia: intervalo de tiempo para producir una respuesta a un even
    ||
    tiempo de respuesta del sistema
  - Productividad (throughput): número de respuestas a eventos que se realizan por unidad de tiempo en un intervalo de observación.
  - Capacidad: Medida de cantidad máxima de trabajo. Puede estar limitada por una condición de latencia máxima
  - Modos: Pueden existir diferentes tipos de requerimientos en función de la fase de ejecución.

La cadena de proceso presenta dos tipos de tiempo de espera:
- tiempo de proceso: depende de la capacidad del elemento
- tiempo de espera ante recursos compartidos
  - acceso concurrente
  - teoría de colas

DISEÑO ORIENTADO AL RENDIMIENTO

1. Establecimiento de los req. no funcionales
2. Definir cadena/s de procesamiento
3. Cuantificar los elementos conocidos de la cadena
4. Partir de unas especificaciones iniciales de capacidad para los elementos de bajo diseño.
5. Realizar una estimación del rendimiento del conjunto.
6. Alternativas si las estimaciones no dan resultados satisfactorios
   6.1. Variar especificaciones de capac. de elem. de bajo diseño
   6.2. Variar la propia estructura de la cadena
   6.3. Volver a 3
7. Si los resultados son satisfactorios, finaliza el diseño.

(ORBs)

- Orientado a objetos
- No necesitamos encargarnos de la traducción de datos (lo hace JavaRMI)
- Alto nivel
- 1 lenguaje: Java
- Más fácil que Corba

- Útil para varios lenguajes de programación.
- Alto nivel
- Muy compatible
- Mejor para cosas no muy difíciles
- Ya proporciona mecanismo de traducción de datos.
- Tamaño msg pequeño

WS - Soap

- Atraviesan corta-fuegos
- Mensajes de gran tamaño
- Ineficiente
- Estándar
- Soporta múltiples lenguajes y sist. heterogéneos
- Sencillo integrar nuevos servicios descubiertos en tiempo de ejecución mediante UDDI y WSDL.
- Usa XML para repr. datos (indep. plataforma). ⇒ No hacemos nada

WS - Rest

- Atraviesan corta-fuegos
- Más eficiente que soap
- Mensajes más pequeño que soap.
- Habría que ponerse de acuerdo con la representación de datos

RPC

- Sin técnicas orientadas a objetos.
- Alto nivel
- Depende de la versión utilizada.
- SUN-RPC ya se encarga del marshalling y unmarshalling.
- Interfaz de progr. definida

UDP

- Tamaño mensajes grande
- No proporciona transparencia de datos (malo para clientes heterogéneos)
- Útil para funcionalidad fácil. (bajo nivel)

TCP

- Útil para funcionalidad fácil. (bajo nivel)
- Malo para clientes heterogéneos ⇒ ↑ homogéneos

Colas mensajes

- Cliente y servidor no necesitan estar conectados a la vez (desacople en el tiempo)
- El cliente no queda bloqueado a la espera (asincronismo)
- Procesamiento por prioridad
- Deben encargarse de la traducción de datos para garantizar la tr. datos