EL PROCESAMIENTO COOPERATIVO Y LA ARQUITECTURA CLIENTE-SERVIDOR. ARQUITECTURA SOA.

Actualizado a 13/01/2019

1. PROCESAMIENTO COOPERATIVO

El sistema centralizado es aquel en el que un Host ejecuta el 100% de la lógica del sistema, y el terminal de usuario se limitaba a presentar los resultados:

Pros	Contras
• Facilidad de gestión • Todo reside en una máquina física	• Poca escalabilidad y flexibilidad • Desaprovecha procesamiento • Dependencia • Falta de integración

La arquitectura distribuida o de procesamiento cooperativo consiste en utilizar componentes más o menos autónomos que se ejecutan en unidades de hardware interconectadas. Los distintos elementos lógicos trabajan en la realización de una tarea común.

Las principales características son:

- Sincronización a través de elementos distintos a un reloj común (característico de los sistemas centralizados).
- Concurrencia global, es decir, la ejecución paralela por demanda de distintos usuarios.
- Tolerancia a fallos de forma que en el caso de fallo de un nodo, este fallo será transparente al resto de usuarios.
- Sistemas abiertos y heterogéneos, que facilita el ahorro de coses al no ser dependiente de un único fabricante.

Los objetivos principales que persiguen los sistemas distribuidos son:

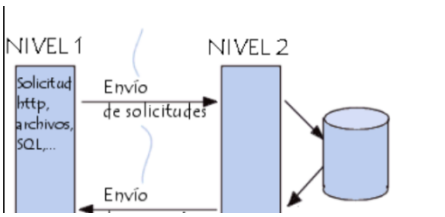
- Transparencia – visión de un único sistema para el usuario.
- Fiabilidad – tolerancia a fallos
- Rendimiento
- Escalabilidad
- Flexibilidad
- Seguridad

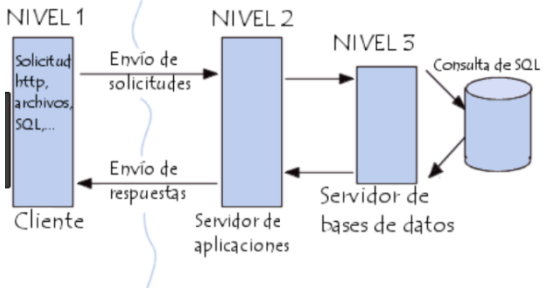
2. ARQUITECTURA CLIENTE-SERVIDOR

Uno de los modelos más extendidos en los sistemas distribuidos es el cliente/servidor. Se basa en la existencia de un sistema Servidor encargado de proporcionar un servicio a través de una interfaz o protocolo a un Cliente, quien lo ha solicitado previamente.

Dentro de este modelo, se distinguen dos arquitecturas:

- Cliente/servidor de 2 capas, donde el servicio se divide en dos capas:
  - Presentación y negocio
  - Datos





- Cliente/servidor de 3 o N capas, donde se divide el servicio en 3 o N capas
    - Presentación
    - Lógica de negocio
    - Acceso a datos

3. ARQUITECTURA SOA

El **objetivo principal** de la arquitectura SOA es ofrecer una estrategia para la **integración** de aplicaciones, enfocada en la **construcción de servicios y no de aplicaciones finales**.

Un servicio expondrá una funcionalidad definida, mientras que la aplicación solamente será la encargada de orquestar la ejecución de un conjunto de servicios. Estos servicios podrán ser, a su vez, reutilizados en otras aplicaciones.

En definitiva, la arquitectura definirá:

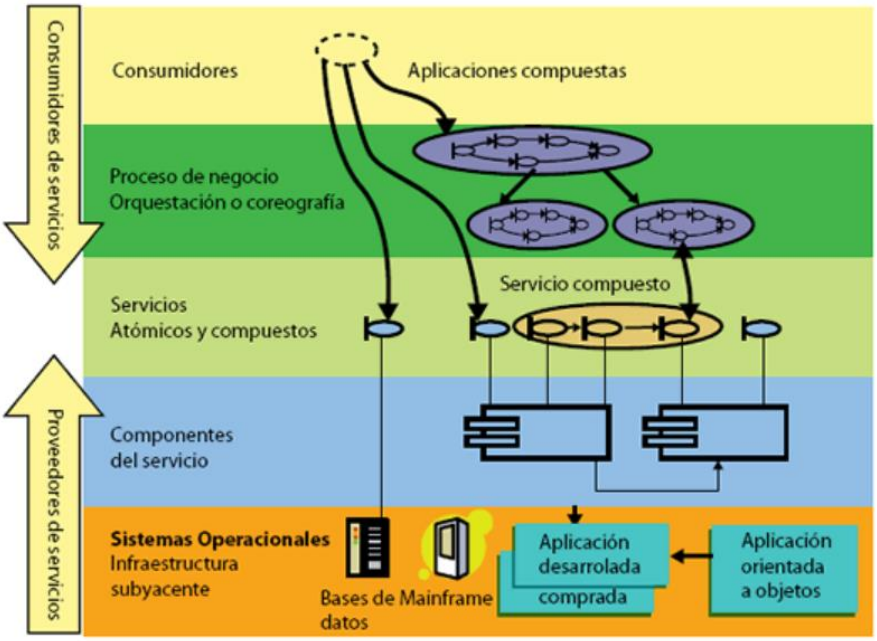
- Servicios a utilizar
- Interacciones entre servicios
- Tecnologías con que se implementarán
- Interfaces entre servicios (mensajes soportados, contenidos, etc)

Según OASIS, SOA se puede definir como: paradigma para organizar y utilizar capacidades distribuidas y bajo el control de diferentes propietarios y dominios. Provee una manera uniforme de ofrecer, descubrir, interactuar y usar dichas capacidades para producir los efectos deseados de manera consistente y medible."

El diseño orientado a servicios, base de esta arquitectura, cuenta con 8 principios de diseño:

1. Contrato de servicio, cumpliendo los mismos estándares de diseño
2. Bajo acoplamiento, evitando acoplarse a la tecnología que los implementa
3. Abstracción, presentando la información mínima requerida
4. Reusabilidad, convirtiéndose en activos de la empresa
5. Autonomía
6. Sin estado, delegando el manejo de estados en la aplicación
7. Garantizan su descubrimiento
8. Preparados para ser utilizado en composiciones (orquestados por la aplicación)

Las capas de la arquitectura SOA se pueden representar como sigue en el siguiente gráfico:



- La capa de sistemas operacionales está constituida por sistemas legacy, sistemas orientados a objetos, y aplicaciones de BI, que podrán ser integrados utilizando técnicas de integración orientadas a servicios.
- Los componentes de servicio aseguran los acuerdos a nivel de servicio, y aseguran la calidad de los servicios externos. Está soportada por los servidores de aplicaciones que ofrecen alta disponibilidad, balanceo de carga, etc.
- La capa de servicios permite que estos servicios sean expuestos y descubiertos para ser invocados por las aplicaciones.
- En la capa de coreografía se establece el flujo para que los servicios actúen conjuntamente ofreciendo el resultado como si fuera una única aplicación.

Existirán adicionalmente las capas transversales de integración y calidad del servicio y monitorización, encargados de las tareas de seguridad, disponibilidad, etc.

COLABORACIÓN ENTRE SERVICIOS

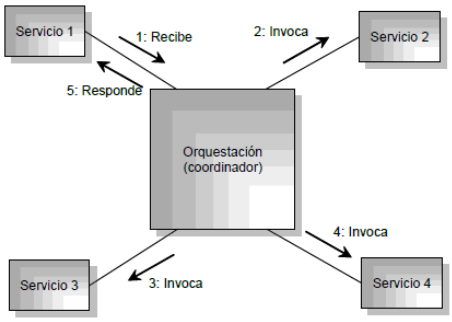
- **Orquestación:** Un proceso se puede considerar una orquestación de servicios cuando es controlado totalmente por una única entidad. Este proceso define completamente las interacciones con los servicios componentes y la lógica requerida para conducir correctamente esas interacciones, y sólo la entidad que está orquestando el proceso conoce el flujo de control e información que sigue el proceso que se está orquestando.

- **Coreografía:** Un proceso es una coreografía de servicios cuando define las colaboraciones entre cualquier tipo de aplicaciones componentes, independientemente del lenguaje o plataforma en el que estén definidas las mismas.

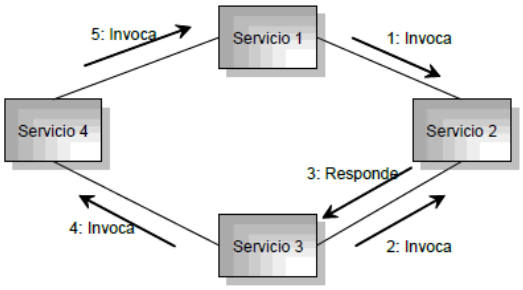
  Un proceso de coreografía no es controlado por uno solo de los participantes, y a diferencia de la orquestación, puede verse como un **proceso público y no ejecutable**. Es público porque define un comportamiento común que todas las entidades participantes deben conocer, y es

no ejecutable porque está pensado para verse más bien como un **protocolo de negocio** que dicta las reglas para que dichas entidades puedan interactuar entre sí.

ORQUESTACIÓN



COREOGRAFÍA



3.1. SOA CON WEB SERVICES

SOA se puede construir sobre servicios web estándar, como por ejemplo SOAP, que gozan de gran aceptación en el mercado. Sin embargo, se puede utilizar cualquier tecnología basada en servicios. **SOA define QUÉ, los servicios web definen CÓMO.**

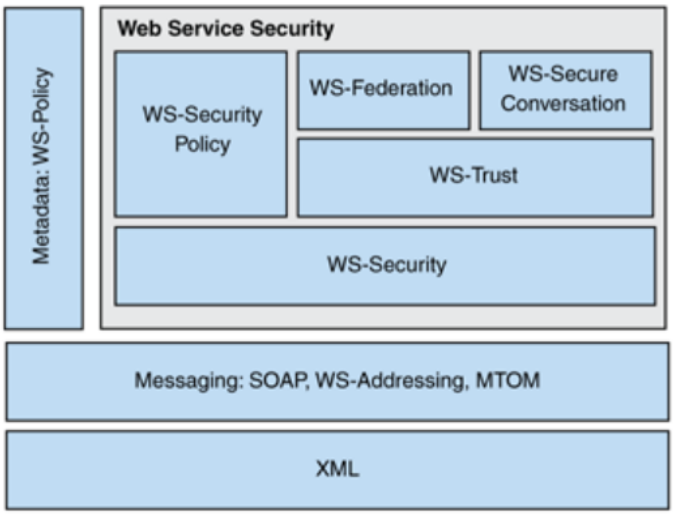
Según el W3C, se puede definir un servicio web (web Service) como: sistema software diseñado para soportar la interacción máquina-a-máquina a través de una red y de forma interoperable. Normalmente, cuenta con una interfaz descrita en una interfaz procesable por un sistema (normalmente WSDL) a través de la que es posible interactuar mediante mensajes SOAP transmitidos usando serialización XML sobre HTTP conjuntamente con otros estándares web[1].

Cabe destacar y tener en cuenta el conjunto de estándares ligados a los servicios web:

XML	W3C	http://www.w3.org/xml/	Lenguaje de marcado
HTTP	W3C		
SOAP	W3C	http://www.w3.org/TR/soap/	Formato de los documentos
WSDL	W3C		Definición de la gramática XML
UDDI	UDDI	http://www.uddi.org	Servicio de registro y descubrimiento

Utilizando este tipo de implementación, también será necesario tener en cuenta la seguridad que permite implementar, a través lo denominado Web Service Security (o WS-Security). Se adjunta a continuación el esquema que permite visualizar la pila de especificaciones relacionadas:

[1] Para profundizar sobre los web services, o resolver dudas, ver la documentación proporcionada por el W3C en el siguiente enlace: https://www.w3.org/TR/ws-arch/



- **WS-Security** – Extensión de SOAP publicado por OASIS, que especifica cómo reforzar la integridad y confidencialidad en mensajes, y permite la comunicación de varios tokens de seguridad como SAML o Kerberos.
- **WS-Secure conversation** – Creado por IBM, que trabaja junto con WS-Security, WS-Trust y WS-Policy para la creación y compartición de contenidos.
- **WS-Trust** – Estándar de OASIS que provee extensiones de WS-Security.
- **WS-Federation** – Especificación de federación de identidades.
- **WS-SecurityPolicy –** Creado por IBM y otros, que se convirtió en un estándar de OASIS.

ESB - ENTRERPRISE SERVICE BUS

Categoría de productos y/o tecnologías middleware, basados en estándares de servicios, que habilitan la creación de arquitecturas basadas en servicios. Definen la infraestructura de integración para la construcción de arquitecturas SOA. Es la pieza donde reside el grueso de la comunicación de la arquitectura SOA

Los sistemas ESB están implementados siguiendo los conceptos SOA, de tal modo que las capacidades de integración son proporcionadas por servicios desplegados dentro del mismo bus de comunicación.

Posibilita una red de sistemas dispares interactuando como un sistema unificado corporativo y resolviendo las diferencias en sistemas HW, SW, Redes y Localizaciones.

¿Qué **ventajas** proporciona el uso de ESBs?

- Enrutado basado en contenidos, en función de uno o varios parámetros de entrada
- Transformación de mensajes
- Configuración y no programación
- Capa de abstracción de servicios
- Auditorías y logs, implantando políticas de monitorización
- Securización
- Validación de mensajes