Presente y futuro de los Sistemas de Información

Los tiempos han cambiado, la **tecnología ha cambiado** de una forma que nunca hubiéramos imaginado, los avances que se han logrado en **últimos cien años** son mayores que los que se ha conseguido en todo el **resto de la historia del hombre**. Y la **tecnología de la información** no está exenta de estos cambios. Los sistemas que en un primer momento fueron desarrollados dentro de las **posibilidades de la técnica** que existía en ese momento hoy se presentan como ineficaces para la tecnología y las **necesidades de los negocio** actuales.

Sistemas de hoy para la realidad de ayer

El mayor problema reside en que los sistemas, como hoy son utilizados, fueron desarrollados para una **realidad del ayer**:

Recursos limitados

En menos de 5 años, el poder de cómputo aumenta en al menos 10 veces. Un disco duro de 1 Terabyte, muy común hoy, era asombrosamente grande hace 5 años. Los almacenamientos de **múltiples terabytes** serán muy comunes en poco tiempo. El ritmo de **crecimiento de la memoria** y **capacidad de procesamiento** es de tal magnitud que la utilización de estos recursos es cada vez menos importante.

Computadoras aisladas

Es una de las limitaciones más importantes. La mayoría de las aplicaciones que se usaron y se siguen utilizando hoy en día no consideran la existencia de **grandes redes** ni de Internet. Las aplicaciones se enfocan principalmente en la **productividad personal**, muy pocas toman ventaja de la **colaboración electrónica**.

Otros

Muy variados. Por ejemplo, la imposibilidad de utilización de una misma aplicación en **diferentes dispositivos** (ordenadores móviles, teléfonos celulares, notebooks, etc.), muy comunes actualmente; la dificultad para que una **aplicación acompañe al crecimiento** de la empresa sin necesidad de grandes reemplazos de software, etc.

Diferentes Arquitecturas

La historia de los **sistemas informáticos** comienzan con una **arquitectura monocapa o centralizada** que aunque podía ser vista hasta hace algunos años, ya prácticamente **se ha dejado de utilizar**. Consistía principalmente de un “gran” ordenador central (*mainframe*) y una serie de terminales que no ejecutaban ningún proceso (*Terminales bobas)*. Tanto el **acceso a los datos, la lógica de la aplicación y la presentación** de la información estaba completamente implementada en **un sólo bloque monolítico** de software. Cualquier modificación sobre la aplicación debía ser hecha en este único módulo.

Un avance sobre este modelo fue realizado a partir de **bases de datos basadas en servidores de archivos**. En este caso, la **BASE DE DATOS** consiste en uno o más archivos reconocibles por el sistema operativo. En esta arquitectura, el programa que permite el **acceso y administración de la base de datos** debe estar muy estrechamente unido a la aplicación cliente.

Arquitectura de 2 Capas

Un avance más a la **arquitectura anterior** consiste en dividir los sistemas de una sola capa en **dos capas bien diferenciadas**. La mayoría de las **aplicaciones Cliente-Servidor** funcionan bajo una arquitectura de dos capas. Estas aplicaciones están compuestas por **una capa de interfaz** con el usuario (front-end), que es la capa en donde el usuario interactúa con su PC y que además generalmente **concentra toda la lógica del negocio**, y una **capa de acceso a datos** (back-end), cuya función generalmente la realiza un **servidor de base de datos** y típicamente reside en un servidor central bajo un entorno controlado.

Uno de los problemas en este tipo de arquitecturas es la **dificultad de manipular los cambios** en la capa que interactúa con el cliente. En estos casos, varias (a veces decenas, tal vez cientos o miles) estaciones de trabajo clientes necesitarán **ser actualizadas con una nueva versión** de la **aplicación del cliente** simultáneamente al **cambio en la base de datos**. Esta generalmente no es una tarea sencilla, sobre todo si las aplicaciones cliente están geográficamente dispersas.

Otro problema es la dificultad de **compartir procesos comunes**. Luego de largas horas de trabajo frente a la máquina para lograr un proceso en particular, este **código es difícilmente reutilizable** en otras aplicaciones.

Un problema más, **la seguridad.** Esta puede ser establecida en cualquiera de las dos capas pero cada una tiene sus limitaciones. La primera solución consiste en **dar privilegios a cada uno de los objetos** que componen la base de datos y a los usuarios. Sin embargo, las corporaciones no requieren sólo **asegurar cuales datos pueden ser actualizados o accedidos**, sino cómo.

En cuanto al segundo punto, que es el más usado, aunque el usuario puede **acceder a la base de datos con su identificación**, tiene dos problemas: dado que ninguno de los objetos en la base de datos es segura, cualquier usuario puede tener **acceso total** a la misma con alguna otra herramienta de front-end (como Excel, Access, etc.); en segundo lugar, la **implantación de la seguridad** deberá ser desarrollada, probada y mantenida en absolutamente toda la red, no importa dónde se encuentren las estaciones cliente.

Más ventajas y desventajas de las 2 Capas

Además de las **desventajas** comentadas anteriormente, tenemos algunas más:

- Los servidores de base de datos no proporcionan un lenguaje de programación “completo” y los procedimientos almacenados, aunque ayudan, no son la solución.
- Los datos no están encapsulados, por lo que sigue siendo necesario que el programador de las aplicaciones de los clientes realice bastante de las tareas de control de la integridad.
- No resulta fácil realizar cambios en la estructura de una base de datos de la que dependen un montón de aplicaciones.
- A medida que el negocio crece, y el número de usuarios utilizando el sistema al mismo tiempo aumenta, se descubre que estos sistemas no escalan. Aplicaciones desarrolladas en 2 capas, funcionan perfectamente en entorno pequeños, pero no pueden acompañar el crecimiento del negocio.

Sin embargo, **no todas son desventajas** en esta arquitectura. Ella ha mejorado de manera significativa algunos problemas que tenía la anterior:

- Cuando un servidor de bases de datos procesa una consulta, la eficiencia en la devolución de la respuesta a esta petición dependerá de la máquina donde se encuentra alojado el servidor y no del computador del cliente que en este caso no es quien procesa la consulta sino quien recibe el resultado.
- El servidor de datos devuelve sólo la información solicitada a través de la red, de tal modo que el tráfico de la misma resulta sustancialmente reducido. Esto permite crear aplicaciones que acceden a grandes cantidades de datos utilizando tan sólo un módem telefónico, por ejemplo, el cual tiene un ancho de banda bajo.
- Un servidor de base de datos puede asegurar más eficazmente la integridad y consistencia de los datos.

Arquitectura de 3 Capas

El esquema anterior ha ido evolucionando en el tiempo y ha dado lugar a una **arquitectura mejorada de 3 capas**. Aparece entre la **capa de interfaz** (presentación) y la de **acceso a datos** una tercera **capa de reglas o lógica de negocio** que es quien realmente representa a la empresa y debe obviar tanto la estructura de los datos como su ubicación. El cliente "pesado" que en la arquitectura de dos capas junta la interfaz con la lógica de la aplicación se divide en un **cliente "ligero" o "liviano"** y la lógica de la aplicación se traslada completamente a un servidor. Por ejemplo, en un **aplicación Web** generalmente el cliente está representado por un navegador que muestra las páginas enviadas por el servidor que administra la lógica del negocio y que permite también el ingreso de datos.

Una **explicación más detallada** de cada una de las capas es:

1. Acceso a datos

Sus funciones incluyen el **almacenamiento, la actualización y la consulta** de todos los datos contenidos en el sistema. En la práctica, esta capa es esencialmente un **servidor de bases de datos** aunque podría ser cualquier **otra fuente de información**. Gracias a esta división, es posible agregar soporte para una **nueva base de datos** en un período de tiempo relativamente corto. La capa de datos puede estar en el mismo servidor que las de lógica de negocio y presentación, en un servidor independiente, o incluso estar distribuida entre un conjunto de servidores.

2. Lógica de negocio

El **comportamiento de la aplicación** es definido por los componentes que modelan la **lógica de negocio**. Estos componentes reciben las acciones a realizar a través de la capa de presentación, y llevan a cabo las tareas necesarias utilizando la capa de datos para manipular la información del sistema. Tener la **lógica de negocio separada** del resto del sistema también permite una **integración más sencilla y eficaz con sistemas externos**, ya que la misma lógica utilizada por la capa de presentación puede ser accedida desde **procesos automáticos** que intercambian información con los mismos.

3. Presentación

La **capa de presentación** representa la parte del sistema con la que **interactúa el usuario**. En una aplicación Web, un navegador puede utilizarse como **cliente del sistema**, pero esta no es la única posibilidad, también puede generarse una aplicación que cumpla las funciones de un **cliente "ligero**" para interactuar con el usuario.

Ventajas de la Arquitectura de 3 Capas

La arquitectura de 3 capas tiene todas las **ventajas de los sistemas cliente/servidor** además de las que de por sí tienen los sistemas que son diseñados de **forma modular**. Pero también han conseguido mejorar muchos de los aspectos que han resultado difíciles de solucionar en la arquitectura de 2 capas:

- **Permite la reutilización**

  La aplicación está formada por una serie de componentes que se comunican entre sí a través de interfaces y que cooperan para lograr el comportamiento deseado. Esto permite no solamente que estos componentes puedan ser fácilmente reemplazados por otros, por ejemplo porque se necesita mayor funcionalidad sino también que los mismos puedan ser utilizados para otras aplicaciones.

- **Acompaña el crecimiento**

  Cada uno de los componentes de la aplicación pueden colocarse en el mismo equipo o distribuirse a través de una red. De esta manera, proyectos de gran envergadura pueden dividirse en pequeños proyectos más simples y manejables, que se pueden implementar en forma progresiva, agregando nuevos servicios según la medida de crecimiento de la organización.

- **Uso eficiente del hardware**

  Debido a que los componentes pueden ser distribuidos a través de toda la red, se puede hacer un uso más eficiente de los recursos de hardware. En vez de necesitarse grandes servidores que contengan la lógica de negocios y los datos, es posible distribuirlos en varias máquinas más pequeñas, económicas y fáciles de ser reemplazadas.

- **Mínima inversión inicial**

  Generalmente, un cambio en el sistema de gestión traía asociado una inversión importante en actualización de hardware en los clientes debido a nuevas necesidades de cómputo de las aplicaciones “pesadas”. Los clientes “ligeros” de esta nueva modalidad permite mantener el equipamiento actual o adquirir uno de muy bajo costo y actualizar, sólo en caso de ser necesario, la tecnología del servidor o servidores.

- **Distintas presentaciones**

  Debido a que separa la presentación de la lógica de negocios, es mucho más sencillo realizar tantas presentaciones diferentes como dispositivos con capacidades e interfaces se tenga (PC, PDA, celulares, etc.)

- **Encapsula los datos**

  Debido a que las aplicaciones cliente se comunican con los datos a través de peticiones que los servidores responden ocultando y encapsulando los detalles de la lógica de la aplicación, obtenemos un nivel de abstracción que permite un acceso a los datos consistente, seguro y auditable. Con esto se pretende que si hay cambios en la capa de

datos, la capa de negocios se haga cargo de administrar tales cambios y el cliente, en la mayor parte de los casos ni se entere.

- **Ahorra tiempo y costos**

  En el desarrollo de nuevas aplicaciones y la integración en el resto de los procesos de gestión de la empresa.

- **Mejor calidad en las aplicaciones**

  Como las aplicaciones son construidas en unidades separadas, estas pueden ser probadas independientemente y con mucho más detalle, esto conduce a obtener un producto mucho más sólido.

Conclusión

El **modelo de 3 capas, o incluso de más capas,** propone un ambiente para la **construcción y ejecución de aplicaciones** de avanzada, que muy probablemente reemplazará a los obsoletos que quedan hoy en día. Una de sus mayores ventajas es que los sistemas se independizan en cierta forma de la **capacidad tecnológica** y el **tamaño del negocio**, por lo que pueden acompañar de manera eficiente el **crecimiento de las empresas** que los utilizan. Dadas las características del modelo, se puede implementar y dejar operativa una **solución de negocios** en tiempos extremadamente cortos, permitiendo conseguir una **ventaja competitiva** particular respecto a otros negocios. También permite la **modificación del sistema** en períodos de tiempo reducidos, incluso cuando es necesario agregar características especiales a las aplicaciones.

ARTICULO PUBLICADO POR **Estr@tegia Magazine** EN:
https://www.estrategiamagazine.com/tecnologia/presente-y-futuro-de-los-sistemas-de-informacion-cambios-tecnologia/