Importancia de la arquitectura empresarial en las organizaciones modernas

Alfonso Rodríguez Suárez*
* Ingeniero de Sistemas. Especialista en Tecnologías Avanzadas para el Desarrollo de Software. Docente de la Facultad de Ciencias Naturales e Ingenierías. Co-Investigador Grupo HYDRA.
arodriguez@unisangil.edu.co

Palabras clave: arquitectura empresarial, arquitectura de software, datos, patrones, Tecnologías de la Información (TI)

Key words: enterprise architecture, software architecture, data, patterns, Information Technology (TI).

Resumen

¿Por qué se requiere la arquitectura empresarial? Para apoyar el negocio, proporcionando la tecnología fundamental y la estructura de procesos necesarios para un efectivo plan de gobernabilidad de Tecnologías de la Información (TI), siendo TI un activo sensible para una estrategia de negocio exitosa. La explotación y manejo eficiente de la información es de vital importancia para alcanzar el éxito de la organización e indispensable para lograr una ventaja competitiva.

Abstract

Why is it required Enterprise architecture? In order to support business, offering the main technology and the structure of necessary processes, for an effective TI strategy, being TI a sensitive active for a successful business strategy. The exploitation and efficient management of information is quite important to reach success in the organization and indispensable to get a competitive advantage.

I. INTRODUCCIÓN

Este artículo hace un recorrido por la evolución de los Patrones Arquitectónicos, presenta las ventajas y desventajas de cada uno de ellos y da a conocer el impacto en las organizaciones; además, hace una presentación del significado, la importancia y el rol que desempeñan la arquitectura empresarial y la arquitectura de software en las organizaciones modernas.

II. RESEÑA HISTÓRICA

A principios de 1950, las organizaciones manejaban sus procesos de forma manual. Se centraban en la industrialización; su estructura organizacional era piramidal y centralizada. En ese momento, inician las ciencias computacionales y aparecen en el mercado las primeras computadoras, cuyo poder de cálculo era extremadamente inferior al hardware que encontramos actualmente. Para esa época existían pocas aplicaciones de negocio; sin embargo, surge la era Batch, caracterizada por la recolección manual de datos de procesos de negocio, cuyo objetivo era aumentar la productividad, explotando el poder de cálculo de los computadores para reducir el tiempo de procesamiento.

Ya para el año 1970 fueron generados varios cambios en el campo tecnológico y organizacional; la competencia entre las empresas se intensificó; por lo tanto, las organizaciones intentaron establecer una estructura matricial, solución que resultó ineficiente y dio lugar a la innovación. De esta manera, al cambiar la forma de administrar, surgieron nuevas necesidades y con ellas, nuevas soluciones. El modelo Terminal – Host – PC es un ejemplo de estas soluciones, que consiste en un host que alberga la lógica de negocio, los datos y la presentación de los datos en la misma máquina y es accedido por las terminales y los PC. Con este modelo de arquitectura, nacen los sistemas de información transaccionales, las bases de datos relacionales, el hardware para redes, etc. Partiendo de esta plataforma tecnológica y los cambios en las organizaciones, se generan requerimientos específicos como son la integración y la independencia de los datos. Sin embargo, este modelo tiene un gran problema: la sobrecarga de procesamiento en el host, que hace muy lento el acceso a los datos y el procesamiento de los mismos.

Por esta razón, los desarrolladores se vieron obligados a adquirir nuevos conocimientos y habilidades en el manejo de lenguajes de programación, bases de datos, sistemas operativos, infraestructuras de comunicaciones, entre otros. Sin embargo, en esta actividad se encontraron dificultades en la integración y mantenimiento, tanto de aplicaciones nuevas como existentes.

En el año 1980 surgen los computadores personales, cuya aparición generó un impacto muy alto en la visión de negocio de las organizaciones. Para esa época se expandieron las aplicaciones de oficina y dieron lugar a nuevos retos de distribución de información, dando paso al modelo Cliente – Servidor. Inicialmente este modelo distribuye los datos en el servidor, la lógica de negocio y presentación de los datos en el cliente, denominándose *cliente pesado*. Este modelo hace responsable al cliente de manejar la mayor parte de la aplicación, la concurrencia de los datos; además, debe conocer el modelo de los datos

para poder acceder al servidor. Presenta una debilidad bastante grande en cuanto al cambio en un requerimiento, ya que obliga la actualización de los clientes [1].

Con el fin de solucionar este problema, en el año 1990 surgió el modelo Cliente – Servidor modificado, el cual permite el almacenamiento de los datos y la lógica de negocio en el servidor, gracias a la evolución de los Sistemas Gestores de Bases de Datos (SGDB), con el cual es posible almacenar la lógica de negocio a través de paquetes, procedimientos almacenados, funciones, disparadores, entre otras; convirtiendo el cliente en un cliente ligero. En este patrón de arquitectura se evidencia que un cambio en la lógica de negocio o en el modelo de datos, afecta sólo al servidor y no a los clientes, quienes responden únicamente por la lógica de presentación [1].

Otro cambio tecnológico que surgió en este periodo y que marcó el destino de la humanidad es la llamada World Wide Web (www). Internet cambió el estilo de vida de millones de personas en el mundo, gracias a la disponibilidad, facilidad de acceso y diversidad en la información. Este crecimiento vertiginoso de las TIC impactó fuertemente la política y economía mundial, acelerando a pasos agigantados la globalización [2].

Otro de los aportes de internet a la humanidad es la colaboración desinteresada de miles de personas que se unen en busca de un bien común. Un ejemplo es el desarrollo y la distribución del software Free/Open/Source, como GNU, Linux, etc. [2].

Teniendo en cuenta el cambio que genera internet en la vida de la sociedad y las empresas, junto con la proliferación del uso de los computadores personales y otras tecnologías como los dispositivos móviles, a partir del año 2000 surge un nuevo modelo arquitectónico denominado *computación distribuida*. La computación distribuida divide los diferentes elementos de la arquitectura en capas, que separan la lógica de negocio, la persistencia de los datos y la presentación, y a su vez se distribuyen en diferentes nodos computacionales. En este tipo de soluciones tecnológicas se utilizan frameworks y patrones de diseño, que soportan muchos de los requerimientos no funcionales, que son características y restricciones de la solución [3].

Una gran cantidad de organizaciones adoptan el *outsourcing* para implementar este tipo de tecnologías en sus empresas; de esta forma disminuyen los costos de implementación y los riesgos de fracaso en los proyectos, debido al alto nivel de capacitación y experiencia que requiere el talento humano que las implementa [4].

En el año 2006, empresas como Google y Amazon construyeron una infraestructura que denominaron *cloud computing* o computación en nube. Este modelo de arquitectura consiste en un sistema de recursos horizontalmente distribuidos, introducidos como servicios virtuales, masivamente escalados [5].

En este sentido, surgen opiniones como la de Scott McNealy, presidente de Sun Microsystems, quien interroga: *¿Para qué mantener sistemas de computación en la empresa si hace más de una década el computador es la red?* Lo anterior refiriéndose a que la computación se hace en un sitio remoto, en lugar de hacerlo en un computador de escritorio [6].

Cloud computing es un modelo relativamente joven y presenta una serie de ventajas y limitaciones. En algunos casos, este modelo reduce costos. Aquí se describen algunos ejemplos tomados de la revista Sistemas, publicada por la Asociación Colombiana de Ingenieros de Sistemas. La versión empresarial de Google Apps cuesta US$50 por usuario, por año, mientras que la licencia de Microsoft Office Profesional cuesta US$499, y aunque la primera es más limitada, ofrece ventajas de trabajo en grupo.

También hay fuertes diferencias en servicios de "hosting" a través de la nube. Mantener un servidor a una empresa le cuesta entre US$800 y US$1000 al mes, pero si se usa un servicio como el de Amazon cuesta entre 10 y 15 centavos la hora.

Una de las preocupaciones de las empresas con este modelo de arquitectura es la seguridad, privacidad de la información, copias de respaldo, aspectos legales, etc. [6].

Teniendo en cuenta esta reseña histórica de la evolución de los modelos arquitectónicos, se hace evidente que tanto las organizaciones como la forma de administrarlas, van evolucionando de la mano con la tecnología. Encontrar un equilibrio entre estos aspectos puede marcar diferencias en la competitividad y liderazgo en el mercado.

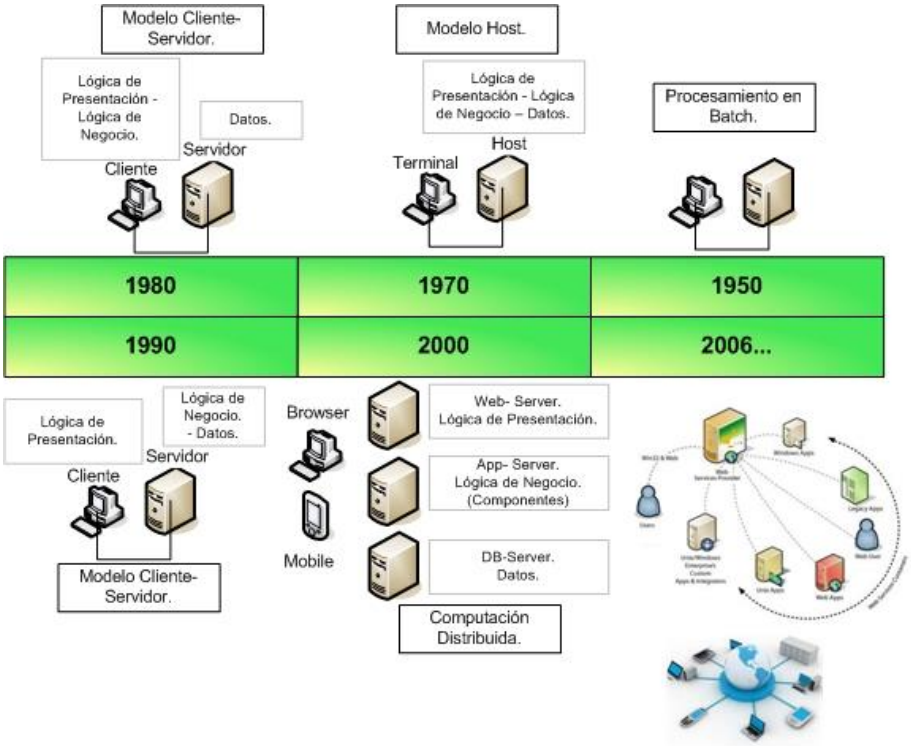

Fig. 1. Evolución de patrones arquitectónicos

III. ARQUITECTURA EMPRESARIAL

Una de las grandes posibilidades para generar ventajas competitivas en las empresas es lograr un equilibrio entre los sistemas de información, las comunicaciones y las plataformas tecnológicas, asociadas con la misión, visión y modelo de negocio.

Por lo tanto, se tiene una relación indispensable entre negocio y tecnología, compuesta de la siguiente manera: en el negocio intervienen la estructura, estrategia, procesos, productos, clientes, socios y fuerzas externas como la competencia, reglamentación, estándares y oportunidades. Por otro lado, en el nivel tecnológico se encuentran servidores, comunicaciones, datos, aplicaciones, gobierno de TI, entre otras [7].

En este sentido, las organizaciones actuales requieren operar eficientemente; es decir, disminuir costos en el apoyo operativo, flexibilizar sus procesos con el fin de adaptarse al cambio rápidamente. Además, la operación de las empresas debe ser centrada en el cliente: esto significa que se debe evitar depender en lo posible de un funcionario de la empresa para realizar la petición y adquisición de bienes y/o servicios. Por lo tanto, los procesos deben estar sincronizados entre sí; además se requiere información precisa y oportuna, que permita el monitoreo de forma flexible y en tiempo real, que maneje indicadores de negocio que conlleven al mejoramiento continuo [7].

En cuanto a las tecnologías de la información y las comunicaciones, se demandan soluciones que sean: flexibles, adaptables, confiables y que soporten los procesos; que sus decisiones de arquitectura sean justificadas en el negocio; que garanticen la calidad del servicio, sean administrables y monitoreadas constantemente [3].

Sin embargo, la situación actual de las organizaciones es otra. Los procesos de negocio son apoyados por componentes tecnológicos aislados; la operación se centra en procedimientos; existen altos costos de integración, operación y estandarización de TI; y hay problemas para obtener información actualizada.

Estos problemas se presentan por el bajo conocimiento que se tiene de la organización, la escasa importancia que se le da a las TI como punta de lanza para ser líderes en el mercado y la poca o nula sincronización entre los procesos de negocio y las Tecnologías de la Información y las comunicaciones.

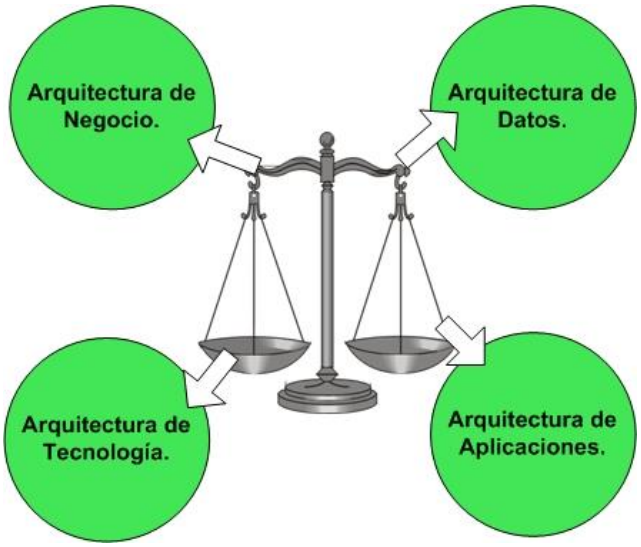

Fig. 2. Arquitectura empresarial

IV. ARQUITECTURA DE SOFTWARE

En secciones anteriores se ha hablado de los avances que se han dado en cuanto a TI, en qué consiste la arquitectura empresarial y cómo ha influido en los últimos años en el desempeño y calidad de los servicios que prestan las organizaciones. A continuación se precisa el concepto de arquitectura de software y su importancia dentro de una organización.

La arquitectura de software de un programa o sistema informático corresponde a la estructura o estructuras del sistema, que incluyen elementos de software, las propiedades visibles desde el exterior de esos elementos y las relaciones entre ellos [8]. La arquitectura es básica para la comunicación, razonamiento y análisis en la construcción de un sistema de información.

Dentro de las organizaciones, muchas personas están interesadas en la construcción de un sistema de información: stakeholders, los clientes, usuarios, desarrolladores, gerente del proyecto, entre otros. Cada uno de ellos, desde el rol que desempeña, presenta inquietudes y peticiones que deben ser solucionadas en el desarrollo del proyecto: la prestación de un determinado comportamiento en tiempo de ejecución, facilidad de uso, bajo costo en la construcción, empleo bien remunerado, prestación de una alta gama de funciones, son algunos ejemplos. En este caso, el arquitecto de software es el encargado de recibir estas sugerencias; y analizar, priorizar y negociar cuáles de ellas pueden ser satisfechas [8].

Además de disponer de un sistema aceptable, se deben tener en cuenta características como: rendimiento, fiabilidad, reutilización, disponibilidad, compatibilidad de la plataforma, utilización de la memoria, uso de la red, seguridad, escalabilidad, usabilidad, tolerancia a fallos, interoperabilidad con otros sistemas así como el comportamiento, entre otros [9].

El problema de fondo, para este caso, es que cada actor tiene diferentes necesidades y objetivos, algunos de los cuales pueden ser contradictorios. Ahora bien, las propiedades pueden ser mencionadas y discutidas, pero en realidad es necesario mediar los conflictos para obtener un equilibrio entre todas las partes del sistema.

Asimismo, el diseño y construcción de una arquitectura está sujeta a un proceso de ingeniería de software y acompañada de actividades muy importantes como son: documentar el caso de negocio para el sistema, comprender los requisitos funcionales y no funcionales, elegir una arquitectura candidata, documentar, comunicar, analizar, evaluar e implementar la arquitectura [8].

Desde el punto de vista técnico, se recomienda tener bien definidos los módulos funcionales y las responsabilidades que se asignan a cada uno de ellos; mantener el principio de ocultación y encapsulamiento de los datos; exponer interfases bien definidas que encapsulen la lógica de negocio; separar las capas que producen los datos de las que los consumen; implementar patrones de interacción simples en todas partes del sistema para mejorar la comprensión; reducir el tiempo de desarrollo; y aumentar la fiabilidad, flexibilidad y reutilización de componentes.

Por lo tanto, una arquitectura de software es una abstracción de un sistema, que suprime los detalles de implementación, en el que cada elemento es parte de la arquitectura, en la medida que el comportamiento pueda ser observado [10].

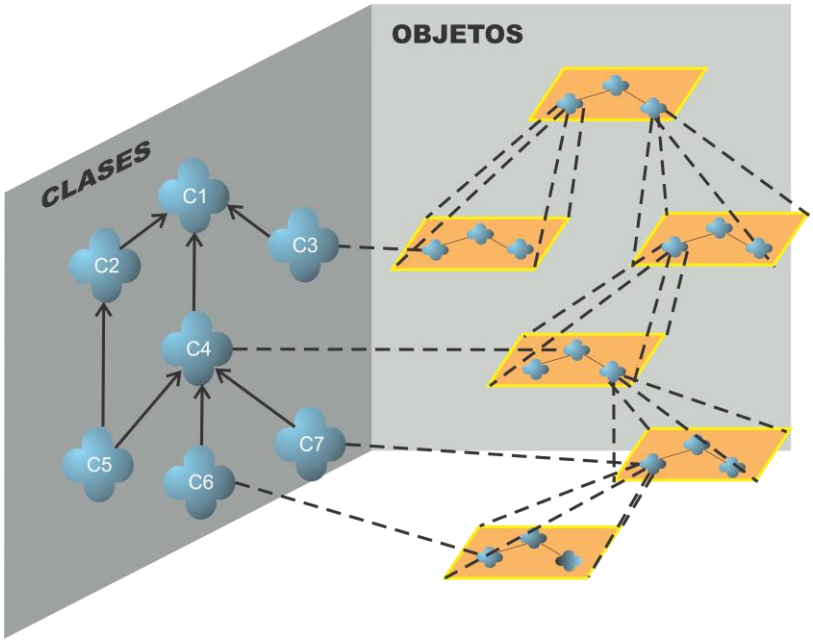

Fig. 3. Arquitectura de Software

V. CONCLUSIONES

Una organización debe conocer y entender muy bien el negocio para poder construir y mantener una infraestructura tecnológica que lo soporte, teniendo en cuenta el papel estratégico y competitivo que aportan las tecnologías de la información y las comunicaciones cuando son alineadas de forma eficiente con los objetivos de la organización.

Conocer los requerimientos funcionales para el diseño de una arquitectura de software no es suficiente; además, es de vital importancia tener en cuenta aspectos como los requerimientos no funcionales, modelo de negocio y procesos de negocio.

Para lograr el éxito, el cliente debe estar en constante comunicación con el equipo de desarrollo con el fin de evitar la solución de problemas inexistentes, levantar requisitos y especificaciones incompletas que posteriormente acarrean errores irreversibles en el análisis, diseño e implementación de la arquitectura de software.

La arquitectura para sistemas de información modernos debe estar centrada en los clientes y los servicios que se presten, más que en los productos; esto disminuye los costos de funcionamiento y el desgaste administrativo; mejora la eficiencia en los procesos; e incrementa la cultura del autoservicio.

Por su parte, los procesos son activos valiosos para una organización; éstos influyen en su eficiencia, por lo tanto: procesos + arquitectura empresarial + arquitectura de software + TI = alta competitividad y liderazgo.

No todos los sistemas merecen considerar numerosos patrones arquitectónicos; cuanto mayor sea el sistema, más dramáticos tienden a ser las diferencias entre patrones. Sin embargo, para sistemas pequeños se puede llegar a funcionar con menos; un solo patrón puede ser suficiente.

Si en un proyecto no se ha logrado definir una arquitectura de software, junto con su justificación, no es recomendable iniciar su construcción a gran escala.

Adicionalmente, es importante resaltar que la evaluación de la arquitectura es un mecanismo económico para evitar desastres: entre más temprano se encuentre un problema, mucho mejor. La evaluación nos entrega como resultado los puntos más riesgosos, obligando a la rápida toma de decisiones.

Por último, vale la pena considerar que el retorno de la inversión está centrado en: reducción de la complejidad en la infraestructura de TI; retorno máximo sobre la inversión existente; flexibilidad para construir, comprar o extender soluciones de TI; y reducción del riesgo total en las nuevas inversiones y costos de las adquisiciones de TI.

REFERENCIAS

[1] C. Binildas, M. Barai y V. Caselli. *Service Oriented Architecture with Java*. Birmingham UK: Packt Publishing, 2008.

[2] s.a. (s.f.) *Internet* (en línea). Disponible en: http://es.wikipedia.org/wiki/Internet

[3] P. Allen y J. Bambara. *SCEA Sun Certified Enterprise Architect for Java EE Study Guide*. Nueva York, Estados Unidos: McGraw-Hill, 2007.

[4] J. Tulach. *Practical API Design. Confessions of a Java Framework Architect*. Nueva York, Estados Unidos: Apress, 2008.

[5] s.a. (s.f.) *Computación en nube* (en línea). Disponible en: http://es.wikipedia.org/wiki/Computaci%C3%B3n_en_nube

[6] F. Rueda. ¿Qué es la Computación en la nube? *Revista Sistemas*, nro. 112, pp. 72-80. Octubre, 2009-enero, 2010.

[7] B. Raynard. *The Open Group Architecture Framework (TOGAF)*. San Francisco, Estados Unidos: The Open Group, 2007.

[8] L. Bass, P. Clements y R. Kazman. *Software Architecture in Practice,* 2ª ed. Boston, Estados Unidos: Addison Wesley, 2006.

[9] K. Zaman y C. Umrysh. *Developing Enterprise Java Applications with J2EE and UML*. Indianapolis, Estados Unidos: Addison Wesley, 2001.

[10] S. Haines. *Pro Java EE 5 Performance Management and Optimization*. Nueva York, Estados Unidos: Apress, 2006.