**Introducción**

los componentes se comunican uno con el otro por medio de interfaces

La complejidad de los sistemas computacionales actuales nos ha llevado a buscar la reutilización del software existente. El desarrollo de software basado en componentes permite reutilizar piezas de código preelaborado que permiten realizar diversas tareas, conllevando a diversos beneficios como las mejoras a la calidad, la reducción del ciclo de desarrollo y el mayor retorno sobre la inversión. Al comparar la evolución del ambiente de IT con el crecimiento de las metrópolis actuales, podemos entender el origen de muchos problemas que se han presentado históricamente en la construcción de software y vislumbrar las posibles y probables soluciones que nos llevarán hacia la industrialización del software moderno.

Este proceso de industrialización ha dado ya sus inicios con implementaciones como la plataforma .net, la cual impulsa la idea de industrializar el software utilizando tecnologías de componentes. Los avances y mejoras presentados en esta plataforma van mucho más allá de las implementaciones iniciales como COM y CORBA, convirtiendo a los componentes .net en verdaderas piezas de ensamblaje, en un estilo muy similar a las líneas de ensamblaje modernas. Asimismo, los nuevos paradigmas como las Fábricas de Software proveen de los medios para hacer la transición desde el 'hacer a mano' hacia la fabricación o manufactura de software.

Si hay algo que ha aprendido el ser humano desde tiempos muy antiguos es a reutilizar el conocimiento existente para sus cada vez más ambiciosas empresas. En efecto, al reutilizar trozos de experiencias, ideas y artefactos, no solo nos aseguramos de no cometer los mismos errores del pasado, sino que logramos construir cosas cada vez más grandes y maravillosas, con bases firmes y calidad incomparable. Este concepto de la reutilización, uno de los primeros que se nos enseñan a quienes entramos al mundo del desarrollo de software, habremos de utilizarlo desde el mismo instante en que escribamos nuestra primera línea de código.

Los sistemas de hoy en día son cada vez más complejos, deben ser construidos en tiempo récord y deben cumplir con los estándares más altos de calidad. Para hacer frente a esto, se concibió y perfeccionó lo que hoy conocemos como Ingeniería de Software Basada en Componentes (ISBC), la cual se centra en el diseño y construcción de sistemas computacionales que utilizan componentes de software reutilizables. Esta ciencia trabaja bajo la filosofía de "comprar, no construir", una idea que ya es común en casi todas las industrias existentes, pero relativamente nueva en lo que a la construcción de software se refiere.

* Introducción Los continuos avances en la Informática y las Telecomunicaciones están haciendo cambiar la forma en la que se desarrollan actualmente las aplicaciones softwareAumento de la potencia de los ordenadoresAbaratamiento de los costos del hardwareRedes de datos de cobertura globalModelos de programación existentes desbordadosNuevos paradigmas de programación
* 3. Los desarrollos tradicionales de aplicaciones incurren en altos costos y en una inversión de tiempo extensa. El DSBC busca, dentro de otros objetivos, reducir el tiempo de trabajo, el esfuerzo que requiere implementar una aplicación y los costos del proyecto, y, de esta forma, incrementar el nivel de productividad de los grupos desarrolladores y minimizar los riesgos globales.Desarrollo de Software Basado en Componentes
* 4. Desarrollo de Software Basado en ComponentesEnsamblaje de partes de software previamente elaboradasInspirada en los procesos de producción de sistemas físicos: Producción de aviones, vehículos, computadores, aparatos electrónicos, etc.Fundamentada en la Reutilización de SoftwareOrientar esfuerzos hacia una industria de partes

**Estado del Arte**

* Concepto Componente Un componente es una unidad binaria de composición de aplicaciones software, que posee un conjunto de interfaces y un conjunto de requisitos, y que ha de poder ser desarrollado, adquirido, incorporado al sistema y compuesto con otros componentes de forma independiente, en tiempo y espacio.
* 3. Desarrollo De Software Basado En Componentes El desarrollo de sistemas de software basado en componentes, (DSBC), es una aproximación del desarrollo de software que describe, construye y utiliza técnicas de software para la elaboración de sistemas abiertos y distribuidos mediante el ensamblaje de partes software reutilizables. La aproximación DSBC es utilizada para reducir los costes, tiempos y esfuerzos de desarrollo del software, a la vez que ayuda a mejorar la fiabilidad, flexibilidad y la reutilización de la aplicación final. Durante algunos años, DSBC fue referida como una filosofía conocida como “compre, y no construya“ promulgada por Fred Brooks en 1987 y que abogaba por la utilización de componentes prefabricados sin tener que desarrollarlos de nuevo.

2. Beneficios del Desarrollo de Software basado en Componentes

En esencia, un componente es una pieza de código preelaborado que encapsula alguna funcionalidad expuesta a través de interfaces estándar[(1)](http://msdn.microsoft.com/es-es/library/bb972268.aspx#ref01). Los componentes son los "ingredientes de las aplicaciones", que se juntan y combinan para llevar a cabo una tarea[(2)](http://msdn.microsoft.com/es-es/library/bb972268.aspx#ref02). Es algo muy similar a lo que podemos observar en el equipo de música que tenemos en nuestra sala. Cada componente de aquel aparato ha sido diseñado para acoplarse perfectamente con sus pares, las conexiones son estándar y el protocolo de comunicación está ya preestablecido. Al unirse las partes, obtenemos música para nuestros oídos.

El paradigma de ensamblar componentes y escribir código para hacer que estos componentes funcionen se conoce como Desarrollo de Software Basado en Componentes. El uso de este paradigma posee algunas ventajas:

1. **Reutilización del software.** Nos lleva a alcanzar un mayor nivel de reutilización de software.
2. **Simplifica las pruebas.** Permite que las pruebas sean ejecutadas probando cada uno de los componentes antes de probar el conjunto completo de componentes ensamblados.
3. **Simplifica el mantenimiento del sistema.** Cuando existe un débil acoplamiento entre componentes, el desarrollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar otras partes del sistema.
4. **Mayor calidad.** Dado que un componente puede ser construido y luego mejorado continuamente por un experto u organización, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo.

**Crítica / Valoración**

La reusabilidad es una importante característica de un componente de software de alta calidad. Los programadores deben diseñar e implementar componentes de software de una manera tal que diversos programas puedan reutilizarlos. Además, cuando los componentes de software interactúan directamente con los usuarios, debe ser considerada la prueba de usabilidad basada en componentes.

Metrópolis: Analogía de la Evolución del Software basado en Componentes

**Pat Helland** [(3)](http://msdn.microsoft.com/es-es/library/bb972268.aspx#ref03), uno de los arquitectos con mayor experiencia en Microsoft, ha desarrollado recientemente una metáfora llamada **Metrópolis** [(4)](http://msdn.microsoft.com/es-es/library/bb972268.aspx#ref04), en la cual compara la evolución de las tecnologías de la información con la forma como las ciudades de Estados Unidos han evolucionado durante los 2 últimos siglos. Al comprender la evolución de las ciudades actuales, podemos darnos una idea del futuro promisorio que tiene el desarrollo de software basado en componentes.

Fábricas de Software: el Nuevo Paradigma

Las Fábricas de Software son una iniciativa propuesta por Microsoft que plasma la necesidad y provee de los medios para hacer la transición desde el 'hacer a mano' hacia la fabricación o manufactura. En sí, una Fábrica de Software es un ambiente de desarrollo configurado para soportar el desarrollo acelerado de un tipo específico de aplicación. Las Fábricas de Software no son más que el siguiente paso lógico en la evolución continua de los métodos y prácticas de desarrollo de software; sin embargo, prometen cambiar el carácter de la industria de software introduciendo patrones de industrialización[(8)](http://msdn.microsoft.com/es-es/library/bb972268.aspx#ref08).

**Propuestas de mejora o Conclusiones**

los componentes pueden ser sustituidos por una versión actualizada o una alternativa sin romper el sistema en el cual operan.

6. Conclusión

Tenemos la fortuna de presenciar el nacimiento de una nueva forma de hacer software, que traerá beneficios inmensos para todos. El desarrollo de software basado en componentes desde siempre fue la idea revolucionaria que nos llevó a pensar que sí era posible el construir software de calidad en corto tiempo y con la misma calidad que la mayoría de las industrias de nuestro tiempo. Al mirar hacia atrás, vemos los increíbles avances que hemos logrado en la comprensión de la forma correcta de reutilizar el software y el conocimiento existente, y nos asombramos cada vez más al darnos cuenta de que este solo es el inicio.

El desarrollo de software basado de componentes se convirtió en el pilar de la Revolución Industrial del Software y se proyecta hoy en día en diversas nuevas formas de hacer software de calidad con los costos más bajos del mercado y en tiempos que antes eran impensables. Empresas como Microsoft entendieron el potencial de esta metodología hace años y hoy nos ofrecen nuevas iniciativas y herramientas que buscan llevar al proceso de construcción de software hacia el sitial privilegiado en el que debió colocarse desde un principio.

Los beneficios derivados de la reutilización de

software están ocasionando un cambio acelerado

en la manera en que la industria de software

desarrolla sus productos.

**Tendencias Futuras**