

DESARROLLO

El futuro de la ingeniería de software



El diseño de programas a partir de información binaria significó un paso sustantivo para la industria desarrolladora de software. Desde ese hallazgo, ocurrido en la década de los cincuenta, hasta hoy, el crecimiento de metodologías para su desarrollo ha subido notablemente en complejidad. Sin embargo, los círculos viciosos que acusa hoy la confección de programas informáticos, impiden conceptualizar el objeto a un nivel superior de abstracción, tal y como lo exige el avance tecnológico y los procesos de negocios. El siguiente artículo sondea el futuro de la ingeniería, a la luz de su propia evolución.

Por Aleksandar Orlic.

La calidad de los sistemas informáticos, satisfacción de sus usuarios y clientes, son temas ampliamente conocidos, recurrentes y de constante preocupación por parte de los practicantes de la Ingeniería de software.

Esta joven y dinámica ingeniería siempre está en busca de mejoras en el desarro-

llo de sistemas, aumento de productividad del ingeniero de software, mayor control del proceso de desarrollo, establecimiento de nuevos métodos de desarrollo. Estos elementos, combinados y aplicados de buena forma, logran un buen proceso de desarrollo y, en definitiva, un buen producto.

Identificar los pasos que ha recorrido

esta ingeniería, analizar su contexto actual y, finalmente, proyectar hacia dónde se dirige, es el pilar de este artículo.

• ALGO DE HISTORIA

La evolución del desarrollo de software ha empezado por las tarjetas perforadas (Fig.

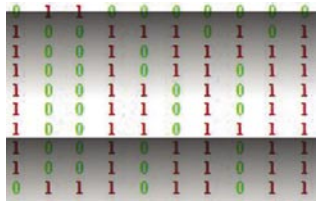


Fig. 1: Primeros programas computacionales en formato de tarjeta perforada.

1) en la década de los cuarenta, cuando los desarrolladores debían conocer detalles de bajo nivel de la máquina que ejecutaba los programas. La solución desarrollada era para pequeños

problemas y muy ligada a la máquina que ejecutaba el programa.

En la década de los cincuenta se avanzó a la confección de programas que contenían información binaria (Fig. 2). La solución desarrollada abordaba problemas de mayor magnitud y complejidad. Se logra la primera separación entre el programador y la máquina.

En la década de los sesenta hacen su aparición los lenguajes estructurados (Fig. 3). Los programas se escribían en un lenguaje que se parecía al lenguaje hablado. Las soluciones desarrolladas involucraban principalmente fórmulas y cálculos matemáticos. La separación entre el programador y la máquina es evidente, quedando ésta en un plano inferior al momento de desarrollar una solución.

```

class factura
{
    var fecha:date;

    function crear_factura(orden de compra :orden de compra)
    {
        this.fecha=date(dd-mm-aaaa);
        while item 1 to itemn
        {
            item::agregar(factura)
        }
        this.estado="enemachina"(creada);
    }
}
    
```

Fig. 2: Lenguaje de máquina como secuencia de 0's y 1's.

Al analizar la evolución de la ingeniería de software, se detectan dos tendencias. La primera tiene en cuenta la solución de problemas a gran magnitud y complejidad, acercándose al mundo real y estableciendo una lejanía con máquina. La otra, tiene relación con el enfoque abstractivo, dejando escondidos los detalles de tecnologías de bajo nivel.

Con respecto a la metodología de desarrollo de software (o proceso de desarrollo), a lo largo del tiempo es notable el crecimiento en su complejidad e importancia. Desde un solo ingeniero que preparaba la tarjeta perforada, la instalaba y la usaba, se llegó a los proyectos de software de varios años de duración, con los ejercicios de ingenieros de muchas especialidades realizando distintas tareas. En estas condiciones, la aplicación de una buena metodología se hace indispensable.

• ACTUALIDAD

La gran mayoría de los sistemas de

software creados hoy en día son los llamados sistemas corporativos; es decir, son orientados a formar una parte importante de los negocios de las grandes empresas.

Continuando en nuestro contexto, se identifica un nuevo enfoque del problema de desarrollo, el apoyo en la sincronización de los procesos de negocio, estructuras complejas de información manejada por el negocio, todo esto entrelazado por restricciones dadas por las reglas del negocio. La tecnología actual, por otro lado, ha alejado más todavía los ingenieros de las máquinas, sistemas operativos, incluso de las tareas de programación.

El enfoque de la mayoría de los equipos de desarrollo e ingenieros, participe en proyectos, sigue siendo con un bajo nivel de abstracción, cerca de la codificación. Se tiende a pensar en la base de datos a utilizar, establecer la navegación de las páginas, programar los protocolos de comunicación, etc. Esto lleva a utilizar las tareas de programación y de pruebas para validar el entendimiento y cumplimiento de la funcionalidad de los sistemas.

Lo anteriormente expuesto muestra un problema metodológico, consecuencia de un desfase entre el enfoque teórico óptimo (análisis de negocio) y el enfoque práctico real (programación y pruebas) en los proyectos de hoy. En otras palabras, la metodología actualmente usada está atrasada con respecto al avance tecnológico.

Desarrollando más esta idea, se identifican algunos problemas concretos de las metodologías:

- Ellas no obligan a sus ejecutores a respetar todas las normas y los procedimientos establecidos, especialmente en las etapas tempranas del proyecto. Es muy fácil y tentador "saltar" una o más de estas etapas, uno o más documentos o modelos, pasando directamente a implementación, producción y posterior corrección de los sistemas desarrollados.
- Control de calidad del producto final. La codificación y complejidad del programa, es demasiada para ser revisada y verificada fehacientemente. Por otro lado, los códigos fuentes y los ejecutables actualmente son los únicos entregables eficientemente verificables.

Estos dos problemas al parecer forman una situación contradictoria: la metodología tradicional no permite validar eficientemente la calidad de los sistemas antes de llegar al código fuente y, por otro lado, tampoco permite llegar eficientemente a códigos fuentes de calidad.

• FUTURO

La salida de este "círculo vicioso" que

se propone es bastante natural: analizar la situación actual de la ingeniería de software en la luz de sus tendencias históricas.

La primera de ellas, relacionada con los problemas que resuelven los sistemas, obviamente presente y muy utilizada en nuestra realidad: los sistemas de hoy no sólo resuelven los problemas del mundo real, sino que están fuertemente influenciando el desarrollo del mundo real.

El enfoque de los ingenieros, desde hace unos años, no se ha movido significativamente por el camino del aumento de abstracción establecido históricamente. Se han hecho ciertos avances en temas puntuales (como arquitecturas de componentes, patrones de diseño, nuevos lenguajes de programación, etc.), pero conceptualmente, metodológicamente, la tarea de programación sigue siendo ejecutada de la misma forma: escribiendo códigos fuentes en forma de los programas textuales.

Éste es, justamente, el "atraso" metodológico que se mencionó. Para disminuir la brecha entre las tendencias históricas, es necesario ajustar la metodología, en términos de aumentar la abstracción del enfoque de los ingenieros de software y acercarlo a la abstracción del problema.

El aumento de la abstracción del proceso de desarrollo del software hace pensar que la próxima generación de métodos de desarrollo se perfila en un conceptualmente nuevo conjunto de herramientas, que permitan aumentar en un mayor grado al actual la abstracción, escondiendo los detalles de más bajo nivel.

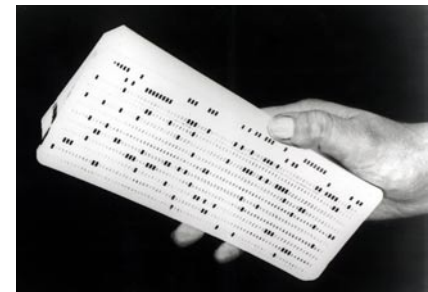


Fig. 3: Lenguaje estructurado se parece al lenguaje natural.

Las nuevas herramientas permitirán "programar" en nivel de análisis, generando códigos en un lenguaje de programación tradicional de forma automática, tal como hoy los compiladores generan el lenguaje máquina a partir del código fuente. Nuevos "lenguajes de programación" naturalmente serían visuales y se parecerían a las especificaciones de software en uno de los lenguajes de modelamiento, por ejemplo UML (Fig. 4).

De hecho, hoy en día existe una tendencia que ya está formalizando este paso evolutivo

en la ingeniería de software. Se llama MDA (Model Driven Architecture) y es establecida por el OMG (Object Management Group), la institución dueña de UML, el lenguaje estándar de modelamiento de software.

Finalmente, ¿cómo estas mejoras influyen en el “círculo vicioso”? Por un lado, nuevas herramientas obligarán a los ingenieros a prestar mayor atención a los modelos de más alto nivel, que al código fuente, y no “saltar” etapas tempranas del proyecto. Por otro lado, los modelos generados serían perfectamente verificables y no se tendría que esperar la codificación y las pruebas para identificar errores funcionales.

• CONCLUSIÓN

Es muy probable que estemos próximos a ver un nuevo paso evolutivo en la ingeniería de software. Tan grande como el invento de lenguajes de alto nivel o análisis y diseño orientado a objetos. Una nueva generación que absorberá a la actual, automatizando la mayoría de las buenas prácticas usadas actualmente.

Una pregunta natural es si una máquina puede llegar a generar programas tan buenos como los hechos por un programador experto. En vez de “romper la

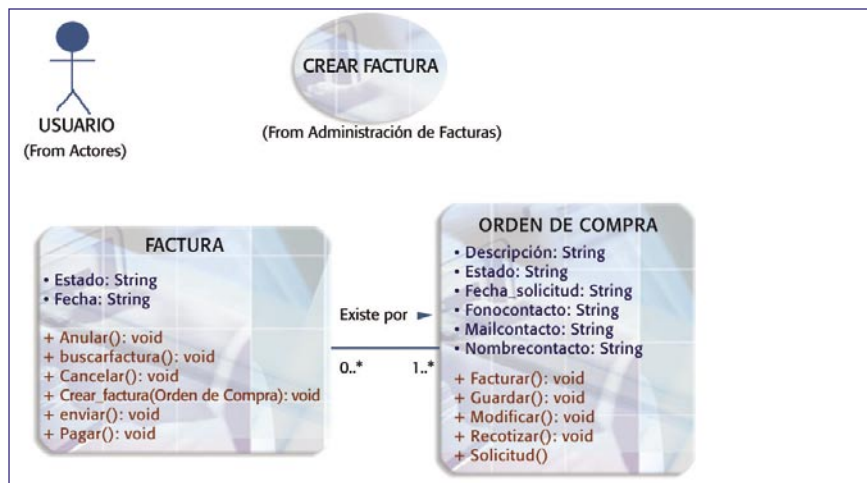


Fig. 4: Probable modelo a futuro.

cabeza” con esta pregunta ahora, quizás sería suficiente recordarse otra vez de la historia. Los primeros compiladores de ninguna generación alcanzaban de inmediato lo deseable y óptimo. Sin embargo, con el paso del tiempo, apoyados en la experiencia de los ingenieros y en el avance tecnológico, se mejoraban hasta el nivel de superar significativamente las generaciones anteriores. Es poco factible que hoy en día alguien cuestione la calidad de los compiladores de, por ejemplo, Java y la calidad del código de máquina generado por ellos.

Aleksandar Orlic

Aleksandar Orlic es ingeniero de electrotécnica con un postgrado en Arquitectura de Sistemas. En la actualidad, es consultor principal y socio fundador de Craftware Consultores.

Más de 25 años difundiendo Tecnología y Gestión para el Hombre de Negocios

Distribución Personalizada a:

- Principales clientes de las empresas avisadoras; base de contactos proporcionada por los propios interesados en difundir su mensaje publicitario.
- Gerente General, Gerente de Administración y Finanzas, Gerente de Operaciones, Gerente de Sistemas y Jefes de Informática de las mayores empresas del sector privado, especialmente la banca, telecomunicaciones, comercio e industria.
- Instituciones Gubernamentales, Regionales y Municipalidades a nivel nacional.
- Universidades e Institutos de Educación Superior: Académicos de las carreras de computación y Bibliotecas.
- Socios de ACTI y Gechs.
- Base de Suscriptores.

Gerencia Comercial
Rodrigo Fuentes Olivos
ventas@informatica.cl

Departamento Suscripción
Marcela Vitale Schute
suscripciones@informatica.cl

Serrano 63, Oficina 52, Santiago, Chile • Fono: 632 2181 / 664 2689