UF2218: Desarrollo de un CMS

Elaborado por: José Manuel Molinero Parra

Edición: 5.1

EDITORIAL ELEARNING S.L.

ISBN: 978-84-16360-70-3

Impreso en España - Printed in Spain

Presentación

Identificación de la unidad formativa:

Bienvenido a la Unidad Formativa UF2218: Desarrollo de un CMS. Esta Unidad Formativa pertenece al Módulo Formativo MF0967_3: Creación y gestión de repositorios de contenidos que forma parte del Certificado de Profesionalidad IFCD0211: Sistemas de gestión de información, de la familia de Informática y Comunicaciones.

Presentación de los contenidos:

La finalidad de esta unidad formativa es enseñar al alumno a integrar en el sistema de información de la organización contenidos para homogeneizar y sistematizar su explotación y manipulación mediante herramientas específicas.

Para ello, se analizarán los proyectos de implementación, el entorno tecnológico y los modelos de procesamiento XML. También se estudiará el modelo de objeto de documento (DOM), y se profundizará en el modelo basado en eventos (SAX).

Objetivos del módulo formativo:

Al finalizar este módulo formativo aprenderás a:

- Distinguir las estructuras y técnicas de programación lógicas para el desarrollo de componentes software, teniendo en cuenta las tecnologías de desarrollo.
- Elaborar elementos software que integren o exploren contenidos de otros repositorios, utilizando interfaz de aplicaciones estándares del mercado para acceder a los repositorios de datos.
- Administrar plataformas de servicios CMS.
- Adecuar y customizar.
- Desarrollar nuevos componentes.

Índice

UD1. Proyectos de implementación

UD2. Entorno tecnológico

UD3. Modelos de procesamiento XML

UD4. Modelo de objeto de documento (DOM)

UD5. Modelo basado en eventos (SAX)

Área: informática y comunicaciones

UD1

Proyectos de implementación

1.1. Metodología de desarrollo
1.2. Análisis de Requerimientos
1.3. Descripción funcional
1.4.Diseño de arquitectura
1.5. Diseño Técnico
1.6. Programación
    1.6.1. Pruebas Unitarias
1.7. Auditoría Funcional
1.8. Auditoría Técnica
    1.8.1. Revisión de Código
    1.8.2. Métricas
    1.8.3. Pruebas de rendimiento
1.9. Despliegue
1.10. Liberación

1.1. Metodología de desarrollo

La primera pregunta que debemos plantearnos es qué es una metodología de desarrollo.

En primer lugar, no debemos confundir la metodología de desarrollo de software (framework) con los paradigmas o filosofías de desarrollo de software, en base a los cuales se aplica una determinada metodología.

Desde el origen de las tecnologías de la información, han surgido varios paradigmas de desarrollo de software; en términos generales son los siguientes:

– Metodologías basadas en ciclo de vida del software

– Metodologías Ágiles

En base a cada uno de estos paradigmas, han aparecido varios modelos:

Metodologías basadas en el ciclo de vida del software	Cascada: framework lineal Espiral: framework combinado lineal-iterativo Incremental: framework combinado lineal-iterativo o Modelo V Prototipado: framework iterativo Rapid application development (RAD): framework iterativo
Metodologías Ágiles	Scrum Extreme programming (XP) Adaptive software development (ASD) Dynamic system development method (DSDM)

Por tanto, una vez situado el contexto adecuado, podemos definir una metodología de desarrollo de software como un framework usado para estructurar, planificar y controlar el proceso de desarrollo de un sistema de información.

Además de lo descrito, una metodología también puede incluir aspectos sobre el entorno de desarrollo –IDE por sus siglas en inglés: Integrated Development Environment, el desarrollo basado en modelos, y la utilización de ciertos componentes (librerías de programación u otras herramientas).

Desarrollo ágil

Las metodologías de desarrollo ágil están de moda y son populares. Todas las compañías punteras en el mercado de tecnologías de la información las utilizan: Google, Yahoo, Symantec, Microsoft, y la lista sigue.

En 1986, [Brooks] predijo que en 1996, ninguna técnica o tecnología de gestión ofrecería un aumento de diez veces en la productividad, la fiabilidad o la simplicidad. Ninguna lo hizo. Las metodologías de desarrollo ágil tampoco.

De hecho, no es recomendable la adopción de una metodología de desarrollo ágil únicamente con el objetivo de aumentar la productividad. Sus beneficios -incluso la capacidad de liberar el software con más frecuencia- se obtienen por el hecho de trabajar *de manera diferente,* no trabajar *más rápido.* Aunque la evidencia anecdótica indica que los equipos ágiles tienen una productividad superior a la media, esa no debe ser la principal motivación de su adopción. Su equipo necesitará tiempo para aprender el desarrollo ágil. Mientras aprenden -y se necesitará un tiempo-, van a ir más lentos, no más rápidos. Además, haciendo hincapié en la productividad únicamente, podría animar a su equipo a tomar atajos y ser menos riguroso en su trabajo, lo que en realidad podría *perjudicar* la productividad.

El desarrollo ágil puede ser la “cosa de moda” que hacer en este momento, aunque no haya ninguna razón que justifique su utilización. Cuando se considera el uso de una metodología de desarrollo ágil, sólo debemos hacernos una pregunta.

¿El desarrollo ágil nos ayudará a tener más éxito?

Cuando pueda responder a esa pregunta, sabrá si el desarrollo ágil es el adecuado para su proyecto.

Entender el éxito

La idea tradicional del *éxito* es la entrega a tiempo, dentro del presupuesto, y de acuerdo con la especificación. A continuación, algunas definiciones clásicas:

- *Exitoso:* "Completado a tiempo, dentro del presupuesto, con todas las características y funciones que se especifican en un principio."

- *Desafiado:* "Completado y en funcionamiento, pero por encima del presupuesto, sobre la estimación de tiempo, [con] un menor número de características y funciones que se especifican en un principio."

- *Dañado:* "Cancelado en algún momento durante el ciclo de desarrollo."

A pesar de su popularidad, hay algo erróneo en estas definiciones. Un proyecto puede tener éxito incluso si nunca genera un solo euro. Y también puede ser un fracaso aunque proporcione millones de euros en ingresos.

La revista *CIO* aportó comentarios sobre esta rareza:

"Proyectos que se ha demostrado que cumplen con todos los criterios tradicionales para el éxito -tiempo, presupuesto y especificaciones- todavía pueden fracasar al final, bien porque no despiertan interés en los usuarios previstos, bien porque en última instancia no agregan mucho valor a la empresa.

... Del mismo modo, los proyectos considerados fracasos de acuerdo a las métricas de TI tradicionales pueden terminar siendo éxitos porque a pesar de los problemas de costes, tiempo o de especificaciones, el sistema es valorado por su público objetivo o proporciona un valor inesperado.

Por ejemplo, en una empresa de servicios financieros, un nuevo sistema tuvo seis meses de retraso y un coste de más del doble de la estimación inicial (el coste final puede 5,7 millones de dólares). Pero el proyecto finalmente creó una organización más adaptable (después de 13 meses) y se consideró un gran éxito -la compañía tuvo una reducción de 33 millones de dólares en cuentas fallidas-, y la reducción del tiempo de generación de valor y el aumento de capacidad se tradujeron en un incremento de un 50 por ciento en el número de pruebas estratégicas de recolección concurrentes en producción."

Más allá de plazos

Tiene que haber algo más en el éxito que el simple cumplimiento de plazos… ¿pero qué?

La definición de éxito dependerá, fundamentalmente, de la visión que se utilice para definirlo.

Software.

Los programadores de software disfrutan como niños creando nuevos sistemas y aplicaciones, e independientemente de si la aplicación funciona o no, se utiliza o no, el programador se sentirá recompensado personalmente porque se habrá divertido escribiendo el código y habrá aprendido nuevas cosas.

Definición

Por tanto, desde un punto de vista individual, *la definición del* éxito *estará centrada en recompensas personales*.

Cuando el proyecto alcanza cierta envergadura a nivel técnico, las aplicaciones y sistemas a desarrollar se vuelven más complejos, y es necesario la intervención de varios programadores codificando al mismo tiempo en el mismo sistema. En este momento, la legibilidad del código y su mantenibilidad se tornan conceptos importantes.

Definición

Por tanto, desde un punto de vista tecnológico, el concepto de éxito se asociará a la excelencia técnica.

A pesar de un buen código, algunos proyectos fracasan. Incluso los proyectos ejecutados impecablemente podrían provocar bostezos de los usuarios. Normalmente, los proyectos forman parte de un ecosistema mucho mayor, en el cual participan decenas, cientos o incluso miles de personas. Los proyectos de TI deben satisfacer las necesidades de esa gente.

De hecho, desde un punto de vista económico, el valor aportado por el software debería superar su coste. En este caso, *el* éxito *significa la entrega de valor a la organización*.

Estas definiciones no son incompatibles entre sí. Los tres tipos de éxito son importantes. Sin el éxito personal, tendrá problemas para motivarse a sí mismo y al resto de miembros de la organización. Sin éxito técnico, su código fuente con el tiempo se derrumbará por su propio peso. Y sin éxito de la organización, su equipo puede encontrar que ya no es requerido en la empresa.

La importancia del éxito de la organización

El éxito de la organización es a menudo descuidado por los equipos de software a favor de los éxitos personal y técnico, más fácilmente alcanzables. Tenga la seguridad, sin embargo, que incluso si *usted* no *está* tomando la responsabilidad de éxito de la organización, el resto de la organización –a nivel global- está juzgando a su equipo en este nivel. Es probable que los directivos senior y los ejecutivos no se preocupen excesivamente de si su software es elegante, fácil de mantener, o incluso valorado por sus usuarios; ellos se preocupan por los resultados. Es decir, del retorno de la inversión en el proyecto. Si no puede conseguir este tipo de éxito, tomarán medidas para asegurarse de que lo hace. Y desafortunadamente, los altos directivos –por lo general-, no tienen el tiempo o la perspectiva de aplicar una solución matizada a cada proyecto. Esperan con razón que sus equipos de trabajo cuiden los pequeños detalles. Cuando los directivos no están contentos con los resultados de su equipo, sacan las espadas. Los costes son el objetivo más evidente. Hay dos maneras fáciles de reducir los costes: establecer plazos agresivos para reducir el tiempo de desarrollo, o enviar el trabajo a un país con un menor coste de mano de obra. O las dos cosas. Estas son técnicas torpes. Plazos agresivos terminan aumentando los horarios en lugar de reducirlos, y la deslocalización tiene costes ocultos.

¿Qué valoran las organizaciones?

Aunque el valor de algunos proyectos proviene directamente de las ventas, hay más en el valor organizacional que simplemente los ingresos.

Los proyectos proporcionan valor de muchas maneras, y no siempre se puede medir ese valor en unidades monetarias.

Aparte de los aumentos de ingresos y los ahorros de costes, las fuentes de valor incluyen:

- Diferenciación competitiva
- Proyección de Marca
- Mejora de la lealtad del cliente
- Satisfacer los requisitos reglamentarios
- La investigación original
- Información estratégica

¿Ayudarán las metodologías de desarrollo ágil a tener más éxito? Es posible. El desarrollo ágil se centra en el logro de éxitos personales, técnicos *y* organizativos.

Éxito Organizacional

Las metodologías de desarrollo ágil logran éxitos de organización, centrándose en la entrega de valor y la disminución de los costes. Esto se traduce directamente en una mayor rentabilidad de la inversión. Las metodologías de desarrollo ágiles también establecen expectativas a principios del proyecto, así que si su proyecto no será un éxito a nivel de organización, dispondrá de tiempo suficiente para llevar a cabo su cancelación con suficiente antelación antes de que su organización haya gastado mucho dinero.

En concreto, los equipos de desarrollo ágil aumentan el valor mediante la inclusión de expertos en negocios y centrando los esfuerzos de desarrollo en el valor central que el proyecto proporcionará a la organización. Los proyectos ágiles liberan sus características más valiosas primero y liberan nuevas versiones con frecuencia, lo que aumenta drásticamente el valor aportado. Cuando las necesidades del negocio cambian o cuando se descubre nueva informa-

ción, los equipos ágiles cambian de dirección para responder a los nuevos requerimientos. De hecho, un equipo ágil y experimentado buscará por sí mismo oportunidades no previstas para mejorar sus planes.

Los equipos ágiles también disminuyen los costes. Hacen esto en parte por la excelencia técnica; los mejores proyectos ágiles generan sólo algunos errores al mes. También eliminan los residuos mediante la cancelación de proyectos malos al inicio y reemplazando las prácticas de desarrollo más costosas con otras más sencillas. Los equipos ágiles se comunican de forma rápida y precisa, y hacen progresos, incluso cuando las personas clave no están disponibles. Revisan regularmente sus procesos y mejoran continuamente su código, lo que hace el software sea más fácil de mantener y mejorar a lo largo del tiempo.

Éxito técnico

La metodología de desarrollo ágil que expondremos es Extreme Programming (XP). Esta metodología es particularmente recomendable para alcanzar éxitos técnicos.

Programadores XP trabajan juntos, lo que les ayuda a realizar un seguimiento de los detalles más minuciosos, necesarios para un gran trabajo y asegura que al menos dos personas revisan cada pieza de código. Los programadores integran continuamente su código, lo que permite al equipo liberar el software cada vez que tiene sentido a nivel de negocio. Todo el equipo se concentra en terminar cada función por completo antes de comenzar la siguiente, lo que evita retrasos inesperados antes de la liberación y permite al equipo cambiar de dirección a voluntad.

Además de la estructura de desarrollo, Extreme Programming incluye prácticas técnicas avanzadas que llevan a la excelencia técnica. La práctica más conocida es el desarrollo basado en pruebas, que ayuda a los programadores a escribir código que hace exactamente lo que ellos creen que hará. Equipos XP también crean diseños simples y en constante evolución que son fáciles de modificar cuando los planes cambian.

Éxito Personal

El éxito personal es, bueno, personal. El desarrollo ágil puede no satisfacer todos los requerimientos para el éxito personal. Sin embargo, una vez que se acostumbra a él, probablemente encontrará lo mucho que le gusta la metodología, con independencia del rol que se asuma:

Los ejecutivos y altos directivos Apreciarán el foco del equipo en proporcionar un sólido retorno sobre la inversión y la longevidad del software.
Usuarios, grupos de interés, los expertos del dominio, y gerentes de producto Apreciarán su capacidad de influir en la dirección de desarrollo de software, el enfoque del equipo en la entrega de software útil y valioso, y el aumento de la frecuencia de entrega.
Gerentes de proyecto y de producto Apreciarán la capacidad de cambiar de dirección al mismo tiempo que cambian las necesidades empresariales, la capacidad del equipo para realizar y cumplir con los compromisos, y la mejora de la satisfacción de las partes interesadas (*stakeholders*).
Desarrolladores Apreciarán la mejora de su calidad de vida como resultado del aumento de la calidad técnica, la mayor influencia sobre las estimaciones y los horarios, y la autonomía del equipo.
Probadores Apreciarán su integración como miembros de primera clase del equipo, su capacidad de influir en la calidad en todas las etapas del proyecto, y más trabajo desafiante e interesante, y menos repetitivo.

¿Qué significa “ser ágil”?

La respuesta es más complicada de lo que parece. El desarrollo ágil no es un proceso específico que se pueda seguir. Ningún equipo practica el método ágil. No existe tal cosa.

El desarrollo ágil es una filosofía. Es una manera de pensar sobre el desarrollo de software. La descripción canónica de esta forma de pensar es el Manifiesto Ágil, una colección de 4 valores y 12 principios. Para «ser ágiles», hay que poner estos valores y principios en práctica.

Manifiesto por el Desarrollo Ágil de Software

Estamos descubriendo formas mejores de desarrollar software tanto por nuestra propia experiencia como ayudando a terceros. A través de este trabajo hemos aprendido a valorar:

– Individuos e interacciones sobre procesos y herramientas.

– Software funcionando sobre documentación extensiva.

– Colaboración con el cliente sobre negociación contractual.

– Respuesta ante el cambio sobre seguir un plan.

Esto es, aunque valoramos los elementos de la derecha,valoramos más los de la izquierda.

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekum	Andrew Hunt	Ken Schwaber
Alistair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kern	Dave Thomas
Martin Fowler	Brian Marick	

Principios del Manifiesto Ágil

– Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.

– Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.

– Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.

– Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.

– Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.

– El método más eficiente y efectivo de comunicar información al equipo de

desarrollo y entre sus miembros es la conversación cara a cara.

– El software funcionando es la medida principal de progreso.

– Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.

– La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.

– La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.

– Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.

– A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

Métodos Ágiles

Un *método* o *proceso,* es una forma de trabajar. Cada vez que hacemos algo, estamos siguiendo un proceso. Algunos procesos se han escrito, por ejemplo, para montar un mueble; otros son ad-hoc e informales, por ejemplo, cuando limpio mi casa.

Los métodos ágiles son procesos que apoyan la filosofía ágil; son elementos individuales llamados *prácticas.* Las prácticas incluyen el uso de control de versiones, el establecimiento de estándares de codificación, y demostraciones semanales a los grupos de interés. La mayoría de estas prácticas han existido desde hace años. Los métodos ágiles las combinan de una forma única, acentuando aquellas partes que apoyan la filosofía ágil, desechando el resto, y mezclándolas en algunas nuevas ideas. El resultado es una metodología ligera, poderosa, y auto-reforzada.

Extreme programming (xp)

Hay varias metodologías dentro del paradigma de desarrollo ágil (Agile Development), entre ellas las más aceptadas globalmente son Scrum y Extreme Programming (XP).