Métodos Ágiles como Alternativa al Proceso de Desarrollo Web

Jorge Bozo Parraguez Broderick Crawford Labrin

Escuela de Ingeniería Informática
Pontificia Universidad Católica de Valparaíso
Av. Brasil 2241
Valparaíso - Chile
Fono 56-32-273767 Fax 53-32-274097
{jbozo | broderick.crawford}@ucv.cl

RESUMEN

Los métodos "ágiles" para el desarrollo de software han sido un gran tema de discusión en la comunidad de la Ingeniería de Software este año. Este trabajo presenta un estudio de este enfoque indicando su origen, promesas, limitaciones, tipos de proyectos en los cuales se puede utilizar y una comparación con los métodos tradicionales. Su filosofía principal se relaciona con dar una rápida respuesta a los usuarios, con entrega de software trabajando, asumiendo que los requerimientos son inestables e involucrando fuertemente a los mismos usuarios en el proceso de desarrollo. La crítica que reciben los agilistas es que sus métodos son poco sistemáticos y documentados. Por otra parte, los agilistas argumentan contra los tradicionalistas, considerando que las respuestas dadas por estos últimos a algunos problemas son muy lentas, al punto que cuando entregan la solución el problema ya no existe o cambió. En este trabajo se plantea que los métodos ágiles y los tradicionales no son estrictamente competidores directos. Cada uno de ellos tiene su propio segmento de aplicación o terreno. Y pueden ser usados en proyectos con diferentes características.

Palabras claves: Ingeniería de Software, Métodos Ágiles, Lightweigh Processes, Ingeniería Web, Cambio de Requerimientos.

1. INTRODUCCION

Desde sus inicios la Ingeniería de Software ha intentado aplicar un enfoque sistemático, disciplinado y cuantificable para el desarrollo, la operación y mantención de sistemas, esto no incluye únicamente los aspectos técnicos de análisis, diseño y pruebas del software, también considera aspectos administrativos tales como: planificación del proyecto, aseguramiento de calidad e implantación [3].

Aunque la práctica promedio de esta disciplina aún presenta un estado deficitario, respecto al cumplimiento del presupuesto y plazo, los resultados de los proyectos de desarrollo de software se han mejorado sustancialmente desde la época de la "crisis del software" [8].

Lamentablemente para los desarrollos Internet no se ha aplicado este enfoque ingenieril. El creciente uso de la Web ha ido acompañado de la aparición de complejas aplicaciones, y este grado de complejidad no se ha visto respaldado por mecanismos que garanticen la calidad de sus aplicaciones. De la necesidad de adecuar los procesos de la Ingeniería de Software, a este entorno de rápido y constante cambio, surge la Ingeniería Web (Web Engineering). Planteándose dar respuesta a los importantes desafíos a los desarrolladores de aplicaciones Web, de características nuevas y especiales y que hacen que las herramientas de modelado empleadas hasta ahora para los desarrollos tradicionales sean poco apropiadas.

Debido a las actuales características del entorno y convergencia hacia Internet, la forma tradicional de abordar un proyecto de desarrollo de software ha cambiado, principalmente porque hoy se debe considerar la evolución de los requerimientos a través de todo el ciclo de vida. Los Procesos Ágiles de desarrollo de software tratan de resolver este problema, es decir, desarrollar software en "los tiempos de Internet" caracterizado por una velocidad de cambio nunca antes vista. Los enfoques ágiles utilizan procesos técnicos y de gestión cuyo objetivo es adaptarse continuamente a los cambios sugeridos por la experiencia ganada en conjunto con el usuario durante el proceso de desarrollo, los cambios de requerimientos y los cambios en el ambiente. Los seguidores de los métodos más formales no pueden entender cómo se podría construir algo que trabaje bien sin analizar los requerimientos. Pero los agilistas no se conforman con perder tiempo valioso y escaso en analizar requerimientos que cambiarán [5].

Este trabajo presenta un estudio del enfoque agilista indicando su origen, promesas, limitaciones, contraste con los métodos tradicionales y tipos de proyectos a los cuales se puede acomodar.

2. REQUERIMIENTOS QUE CAMBIAN

Hace años escuchamos respecto a la crisis del software y nuevamente nos encontramos con sus mismos síntomas: proyectos que no cumplen sus presupuestos y exceden sus plazos de entrega. En muchos casos la causa es la falta de acuciosidad en determinar los requerimientos y controlar sus cambios a lo largo del proceso de desarrollo.

Los requerimientos conforman los fundamentos de un sistema de software y dirigen su proceso de desarrollo. Los requerimientos funcionales indican lo que el sistema debe hacer, requerimientos de datos indican qué debe almacenar y requerimientos de calidad indican cómo se debe desempeñar.

El Análisis y Especificación de Requerimientos desde hace muchos años es identificado como un asunto clave en el proceso de desarrollo de software [11]. Tradicionalmente fue asociado a las primeras fases de este proceso. Pero debido a las actuales características del entorno: exigencia de menor tiempo a mercado, velocidad del cambio tecnológico y la omnipresencia de Internet. La manera de llevar a cabo esta actividad ha cambiado para un gran número de proyectos y debe ser entendida como una administración continua de la evolución de los requerimientos a través del ciclo de vida.

Actualmente se puede observar un amplio espectro de diferentes formas de abordar un proceso de desarrollo de software, en cuyos extremos destacan por un lado los conservadores y formalistas, quienes se apegan a un proceso de desarrollo tradicional bien planificado, haciendo énfasis en los detalles para asegurar la calidad de los productos de software, aspirando a niveles de control y repetición. Y en el otro extremo el "manifesto for agile software development" [1] que valora más las respuestas a los cambios, a través del uso de fuertes interacciones con usuarios, que el seguimiento estricto a una planificación. Argumentan que la rigidez de los planes o contratos muchas veces deriva en que cuando el sistema es entregado el problema que se pretendía resolver cambió o ya no existe.

El concepto de Requerimiento es relevante para definir la estrategia de desarrollo a seguir. Los seguidores de los métodos más formales no pueden entender cómo los "agilistas" podrían construir un sistema que trabaje bien sin analizar los requerimientos. Éstos a su vez, no se conforman en perder tiempo valioso en analizar requerimientos que con una alta probabilidad cambiarán. Su objetivo es construir software con pocos defectos pero que cuando se libere no sea el software incorrecto, y que la problemática que pretende abordar aún exista.

La existencia de estas dos visiones tiene mucho que ver con la transición de la economía industrial al nuevo paradigma tecno-económico actual, sociedad del conocimiento o como se le denomine al actual entorno competitivo. Donde la manera de producir y entregar bienes y servicios está cambiando [9].

Hasta hace muy poco se le pedía a los analistas que tuvieran Especificaciones de Requerimientos detalladas, hoy es momento de transar. Este trade off permite bajar la vara al momento de calificar o aceptar un "buen requerimiento". Lo que ha contribuido a definir nuevos escenarios por donde se están moviendo muchos proyectos de desarrollo principalmente para la Web.

La satisfacción de requerimientos es una condición necesaria pero no suficiente para producir un sistema de excelencia. Los detalles que hacen a un buen sistema que no son especificables por anticipado deben ser tratados durante todo el proceso de desarrollo y no necesariamente sólo en la fase temprana de análisis. Más aún, si se considera que muchos de los desarrollos para la Web están orientados a audiencias de usuarios muy amplias y variadas, la forma en que se ha realizado la captura de requerimientos tradicional, basado principalmente en las entrevistas con los usuarios, debe evolucionar. Hoy se debe considerar, además de los requerimientos de usuario, una multiplicidad de fuentes de requerimientos: estándares, leyes, certificaciones, integración de componentes comerciales COTS (Comercial off the shell). Y otras propiedades emergentes no funcionales del sistema y que valoran los usuarios del siglo XXI [4].

3. LAS METODOLOGIAS ACTUALES

Para las metodologías de desarrollo de software en general se han planteado tres puntos claves [20]:

- El propósito de las metodologías de desarrollo de software es mitigar los riesgos inherentes a un proyecto.
- El propósito de las metodologías de administración de requerimientos es mitigar los riesgos relativos a los requerimientos de un proyecto.
- Ninguna metodología se ajusta perfectamente a todos los proyectos.

Las metodologías entendidas como un conjunto de procesos y herramientas para el desarrollo de sistemas de Información han tenido una evolución creciente desde la aparición de los conceptos de Ingeniería de Software. En este contexto hoy en día hay una fuerte disputa entre los métodos más tradicionales y el emergente movimiento agilista.

3.1 Métodos Ágiles

Los métodos ágiles son también denominados livianos (lightweight), adaptativos e iterativos.

- Livianos puesto que ellos se consideran más fáciles de usar y no enfatizan la planificación y documentación detallada como sí lo hacen los métodos tradicionales más formales, que en contraste con las ágiles se denominan pesados (heavyweight).
- Adaptativos porque consideran los cambios como una realidad inevitable y no como excepciones. Los métodos ágiles permiten una rápida reacción frente a estos.
- Iterativos porque dividen el desarrollo del proyecto en ciclos muy cortos. Al final de cada ciclo una porción ejecutable del sistema es entregada al usuario para que éste la valide.

De las metodologías ágiles las más representativas corresponden a [15]:

- Programación Extrema XP (Extreme Programming),
- Open Source
- Crystal de Cockburn
- Desarrollo de Software Adaptable de Highsmith
- Scrum
- Desarrollo Guiado por Caraterísticas
- DSDM (Método de Desarrollo de Sistema Dinámico)

Para evitar confusión sobre el significado de un Proceso Ágil diecisiete investigadores de metodologías de desarrollo acordaron en el año 2001 a qué denominar "agilidad", el resultado de este acuerdo fue la formación de la Alianza Agilista y la publicación de su Manifiesto [1].

3.2 Principios del Manifiesto Agilista

El manifiesto de la Alianza Agilista es una definición resumida de los valores y objetivos del proceso de desarrollo ágil , este manifiesto detalla los principios comunes para todos los procesos denominados ágiles. Cuyos principios son los siguientes:

- Nuestra prioridad más alta es satisfacer al cliente con la entrega temprana y continua de software que pueda valorar.
- Los cambios en los requerimientos son bien aceptados, aún en fases tardías.
- Entregue software trabajando con frecuencia

- Usuarios y desarrolladores deben trabajar juntos diariamente durante el proyecto.
- Construya los proyectos alrededor de individuos motivados. Déles el ambiente y apóyelos en lo que necesiten. Y confíe en ellos.
- El método más eficiente y eficaz de compartir la información es la conversación cara a cara.
- El software trabajando (ejecutable) es la principal medida de avance.
- Los procesos ágiles promueven el desarrollo sostenible.
- Preocupación permanente por la excelencia técnica y el buen diseño refuerzan la agilidad.
- La simplicidad es esencial.
- Las mejores arquitecturas, requerimientos y diseños surgen de los equipos dentro de la organización.
- Los equipos de proyecto evalúan su efectividad a intervalos regulares y ajustan su comportamiento de acuerdo a esto.

3.3 Las metodologías Ágiles versus las Tradicionales

Las metodologías tradicionales imponen una disciplina de trabajo sobre el proceso de desarrollo de software, con el simple objetivo de asegurar que el software que se obtenga satisfaga los requerimientos del usuario y reúna estándares aceptables de calidad. El trabajo de planificación es riguroso, aún cuando en la práctica muchas veces estas planificaciones no se respetan. En contraposición a este tipo de metodología tradicional las metodologías ágiles aportan nuevos métodos de trabajo que apuestan por una cantidad apropiada de procesos. Es decir, no se desgastan con una excesiva cantidad de cuestiones administrativas (planificación, control, documentación) ni tampoco defienden la postura extremista de total falta de proceso. Ya que se tiene conciencia de que se producirán cambios lo que se pretende es reducir el costo de rehacer el trabajo.

Se identifican como principales diferencias entre ambos enfoques las siguientes [14]:

- Las metodologías ágiles son adaptativas mas que predictivas. Una metodología tradicional potencia la planificación detallada y de largo alcance de prácticamente todo el desarrollo de software (ejemplo modelo cascada). En contraste las metodologías ágiles proponen procesos que se adaptan y progresan con el cambio, llegando incluso hasta el punto de cambiar ellos mismos.
- Las metodologías ágiles están orientadas más a los desarrolladores que a los procesos. Intentan entonces trabajar con la naturaleza de las personas (desarrolladores y usuarios) asignadas a un proyecto, más que contra ellos, de tal forma que permiten que las actividades de desarrollo de software se conviertan en una actividad grata e interesante.

4. LIMITACIONES

Se identifican algunas limitaciones a la aplicación de los procesos ágiles en algunos tipos de proyectos, en los cuales su uso puede ser muy problemático. Como las organizaciones intentan ganar ventajas competitivas con la pronta entrega de soluciones a mercado, los desarrolladores están bajo una fuerte presión de liberar aplicaciones rápidamente. Los procesos ágiles fueron desarrollados para tratar de resolver este problema, es decir, desarrollar software en "los tiempos de Internet" caracterizados por una velocidad de cambio nunca antes vista.

Como se ha señalado, los enfoques ágiles utilizan procesos técnicos y de gestión cuyo objetivo es adaptarse continuamente a los cambios sugeridos por la experiencia ganada durante el proceso de desarrollo, los cambios de requerimientos y los cambios en el ambiente de desarrollo. Los procesos ágiles apoyan principalmente la temprana producción de código. De código que trabaje efectivamente. Esto se logra estructurando el proceso del desarrollo en iteraciones, donde una iteración se centra en la entrega del código y otros artefactos que proporcionan principalmente valor al cliente y en segundo plano al proyecto como tal [18].

Los agilistas y sus críticos focalizan la discusión en torno a la codificación. Por un lado se sostiene que el código es el único entregable que realmente importa, desplazando el rol del análisis y diseño en la creación de software. Los críticos del proceso precisan que el énfasis en el código puede conducir a la pérdida de la memoria corporativa o conocimiento organizacional, porque hay poca documentación y modelos para apoyar la creación y evolución de sistemas complejos [19]. Independiente de la posición que se adopte, es válido preguntarse respecto a cuales son las prácticas más adecuadas para el desarrollo de software en ambientes de cambio acelerado, y ponderar las reales posibilidades y limitaciones de los procesos ágiles.

Obviamente, existirán algunas situaciones en las cuales los procesos ágiles no resultan adecuados. Y es posible que algunos enfoques de procesos ágiles se puedan extender para tratar ciertas limitaciones. Pero en general, tales extensiones implicarán el incorporar principios y prácticas que se asocian al desarrollo planificado tradicional. ¿Qué tipo de proyectos de desarrollo de software son pertinentes al manifiesto agilista?

Con el propósito de orientar la respuesta a lo anterior, se presentan a continuación la limitaciones de los procesos ágiles [21].

4.1 Proporcionan una ayuda limitada en equipos de trabajo dispersos físicamente

El énfasis en la "co-localización" del equipo de trabajo propuesta por los agilistas no es adecuada cuando el equipo de desarrollo se encuentra físicamente disperso. Cuando los desarrolladores y clientes no pueden realizar la comunicación cara a cara planteada, aunque se podría establecer una comunicación cara a cara usando tecnologías tales como videoconferencia u otras, las alternativas en general no son tan eficaces como esperarían los agilistas.

La comunicación cara a cara es tan importante en ambientes de trabajo distribuido como no distribuido, pero en ambientes distribuidos probablemente ocurrirá con menos frecuencia y tiene que ser planeada por adelantado asegurándose que todos implicados participarán. Se pueden utilizar las reuniones cara a cara como acontecimientos importantes de la sincronización dentro de equipos de desarrollo geográficamente dispersos, donde se informa del avance realizado por otros y discuten sobre planes futuros de desarrollo del producto. En tales las reuniones, la documentación llega a ser una forma elemental de comunicación. Documentación de requerimientos y diseños, producidos y mantenidos de manera tal que aseguren en forma oportuna una visión común, a los miembros del equipo, respecto al producto a ser construido. Esto no se debe interpretar como requisito para documentar o modelar todos los aspectos del software.

La documentación y los modelos deben ser creados y ser mantenidos solamente si proporcionan valor al proyecto y/o a los stakeholders.

4.2 Proporcionan una ayuda limitada en equipos de trabajo grandes

Los proceso ágiles proporcionan ayuda a proyectos donde los mecanismos de coordinación, control, y comunicación son aplicables a equipos de desarrollo pequeños y medianos. Con equipos grandes, el número de interacciones "cara a cara" y de revisiones a ser consideradas puede reducir su eficacia.

Los equipos grandes requieren enfoques menos ágiles para abordar su gestión. En este tipo de proyectos los enfoques tradicionales, que enfatizan la documentación, definición de actividades, control de los cambios y el desarrollo centrado en la arquitectura diseñada, son más aplicables.

Lo anterior no debe ser excluyente, es decir no se debe suponer que las prácticas ágiles no son aplicables en tales ambientes. Puede haber oportunidades para utilizar estas prácticas, pero el grado de agilidad posible puede ser menor que la que se da para un equipo más pequeño de proyecto.

4.3 Consideran una ayuda limitada al tratamiento de subcontratos

La externalización del desarrollo del software hacia un subcontratista se basa en un contrato que estipula exactamente qué se requiere de él. Las tareas subcontratadas tienen que estar muy bien definidas, más aún en los casos que consideren licitaciones. Donde los oferentes deberán proponer un plan de trabajo que generalmente incluye un proceso, hitos y productos intermedios a entregar en cada una de las fases. Todo descrito con suficiente detalle para hacer consistente una estimación de costos.

El proceso puede ser uno iterativo e incremental, pero la oferta deberá "aterrizar" este proceso especificando el número de iteraciones y valorizando los productos intermedios de cada iteración para poder competir.

Es posible que un contrato pueda ser escrito de tal forma que permita a un subcontratista un cierto grado de flexibilidad dentro de restricciones generales de tiempo y costo. Esto es posible si el subcontratista tiene buena reputación y los grados de confianza son altos, pero en general cuando los proyectos se externalizan existe la tendencia de que "quede por escrito" en un contrato el tipo de trabajo solicitado y su costo.

4.4 No privilegian la reutilización de componentes

Los procesos ágiles se focalizan en el desarrollo de productos de software que resuelven un problema específico. La necesidad de salir prontamente a mercado y la inclusión de usuarios específicos en los equipos de trabajo, generalmente imposibilitan la obtención de soluciones generalizadas.

Aún cuando se reconocen las ventajas a largo plazo que implica la reutilización de software. La amplitud de las posibilidades de utilización de un artefacto reutilizable requiere que su proceso de desarrollo considere actividades de aseguramiento de calidad. La existencia de errores severos tendrá un impacto tan amplio como el número de aplicaciones que reutilizan el componente.

No está claro cómo los procesos ágiles pueden adaptarse al desarrollo de reusables, aunque se reconoce que su utilización puede redundar en bajar los tiempos de desarrollo, uno de sus principales objetivos.

4.5 Proporcionan una ayuda limitada para desarrollar software de seguridad crítica

El software de seguridad crítica es aquél donde una falla puede dar lugar a lesiones directas sobre humanos o cause daños económicos severos. Los mecanismos de control de calidad apoyados por los procesos ágiles actuales (revisiones informales, programación de a pares) no han demostrado ser adecuados para garantizar a los usuarios que el producto es seguro.

Existen dudas respecto a que si estas técnicas por sí mismas sean suficientes. Especificaciones formales, realización de pruebas rigurosas, y otra técnicas formales de análisis y evaluación proporcionan un mejor soporte para el desarrollo de este tipo de aplicaciones. Aunque resulten más costosos, principalmente en tiempo, el trade off entre la salida a mercado y cero defecto para software de seguridad crítica es diferente al de otros proyectos.

Sin embargo, los desarrollos ágil y formal no son incompatibles si consideramos la posibilidad de utilizarlos cada uno en su contexto: las técnicas formales pueden ser utilizadas, dentro de una estrategia agilista más amplia, específicamente para manejar los componentes de alta criticidad para aumentar calidad y confianza.

4.6 Proporcionan ayuda limitada para desarrollar software grande y complejo

El supuesto de los métodos ágiles, de que la refactorización de código soslaya la necesidad del diseño para administrar cambios, no resulta válido para sistemas complejos y de gran tamaño, donde puede haber aspectos de la arquitectura que son difíciles de cambiar. El costo de cambio puede ser muy alto y por lo tanto resulta conveniente realizar esfuerzos adicionales de diseño para visualizar alternativas tempranamente.

Puede también haber sistemas en los cuales la funcionalidad está muy acoplada y firmemente integrada, lo que puede imposibilitar el desarrollo de software incremental. En estos casos el acercamiento iterativo, en el cual el código se produce por iteración se podría utilizar, pero el código generado en cada iteración incluirá algunos componentes incompletos. Lo que no es consistente con la propuesta agilista.

4.7 Dificultad en la utilización de herramientas que apoyen el desarrollo

La mayor parte de las metodologías tradicionales cuentan con herramientas CASE de apoyo para asistir el proceso, donde los desarrolladores y gestores pueden establecer planes y controlar su ejecución, asignar personas a tareas, confeccionar documentos, y validar la consistencia de todas las herramientas de modelado utilizadas [15]. Dentro de los métodos ágiles este tipo de herramientas juegan un rol menor y generalmente las herramientas que se utilizarán serán sólo aquellas que faciliten la comunicación, coordinación y cooperación.

Pese a lo indicando anteriormente, existen esfuerzos tendientes a la adecuación de herramientas de apoyo para el tipo de planificación (períodos muy cortos y con visibilidad sólo sobre las iteraciones y no el proyecto completo), desarrollo y documentación asociados a metodologías ágiles [16].

5. LOS PROYECTOS

Habiendo contrastado las metodologías tradicionales y ágiles, y profundizado sobre las limitaciones de estas últimas se pueden identificar las características y / o condiciones de un proyecto bajo las cuales es aconsejable

utilizar una u otra opción. Las características a considerar en la elección son: relativas a la aplicación, al estilo de gestión, técnicas y sobre el personal involucrado. La Tabla 1 ilustra esto.

Los métodos ágiles y los tradicionales no son estrictamente competidores directos. Cada uno de ellos tiene su propio segmento de aplicación o terreno. Son usados en proyectos con diferentes características: los métodos tradicionales son más adecuados en grandes proyectos con requerimientos estables, aplicaciones críticas, grandes equipos de desarrollo y /o equipos distribuidos geográficamente. Los métodos ágiles en cambio se adecuan mejor en ambientes dinámicos, con equipos de trabajo pequeños y produciendo aplicaciones no críticas. También son una buena elección cuando se trabaja con requerimientos desconocidos o inestables, garantizando un menor riesgo ante la posibilidad de cambio en los requerimientos. Como muchas aplicaciones Web cumplen en gran parte con estas características, podría ser apropiado utilizar los métodos ágiles como estrategia amplia de desarrollo en la Ingeniería Web.

Característica	**Métodos Ágiles (lightweight)**	**Métodos Tradicionales (heavyweight)**
Aplicación		
Objetivos Principales	Capacidad de Respuesta a los cambios	Predecibles y estables
Tamaño	Pequeños proyectos y equipos de trabajo	Grandes proyectos y equipos de trabajo
Entorno	De alto cambio, turbulento y enfocados en el proyecto	De pocos cambios, estables y enfocados en la organización
Estilo de Gestión		
Relación con los usuarios	Dedicados y on-site, interacciones centradas en las prioridades de cada incremento	Interacciones según necesidad; guiadas por la planificación y contratos
Planificación y control	Planes internalizados; control cualitativo	Planes documentados, control cuantitativo
Comunicaciones	Conocimiento interpersonal tácito	Conocimiento documentado explícito
Técnicos		
Requerimientos	Historias informales priorizadas y casos de prueba; suposición de existencia de cambios no previstos	Proyecto formalizado, alcance, intefaces, calidad y evolución de requerimientos predecibles
Desarrollo Pruebas	Diseño simple; incrementos cortos, se supone refactorización de bajo costo Casos de prueba ejecutables definen requerimientos; pruebas	Diseño extenso; largos incrementos refactorización costosa Planes de prueba y procedimientos documentados
Personal involucrado		
Usuarios	Dedicados; en lugar físico de desarrollo, colaborativo, representativo, autorizado, comprometido y competente	Colaborativo, representativo, autorizado, comprometido y competente; no necesariamente en el mismo lugar físico
Desarrolladores	Solo personal con entrenamiento capaz de desempeñar métodos discrecionales; capaz de aprender un método adecuado para nuevas situaciones o reformular un método existente	Personal con entrenamiento capaz de desempeñar métodos discrecionales; capaz de aprender un método adecuado para nuevas situaciones o reformular un método existente. Y además personal con entrenamiento en métodos procedurales.
Cultura Organizacional	Confortable y empowerment con altos grados de libertad	Confortables y empowerment a través de políticas y procedimientos

Tabla 1: Características de los Métodos Ágiles y Tradicionales [17]

6. APLICACIONES WEB Y MÉTODOS ÁGILES

Sin lugar a dudas el desarrollo Web ha tenido en la última década un crecimiento explosivo, por lo cual se han elaborado, una gran cantidad de herramientas para este tipo de desarrollo. Desde una perspectiva general, al desarrollar un proyecto de este tipo se deben tener consideraciones diferentes a las de un proyecto tradicional:

La Web es un medio altamente dinámico, el tipo de usuario que hace uso de ésta es más variado y exigente en relación a requerimientos no funcionales.

6.1 La Ingeniería Web

Históricamente, la Ingeniería de Software ha intentado aplicar un enfoque planificado para el desarrollo, la operación y mantención de sistemas, esto no ha incluido únicamente los aspectos técnicos de análisis, diseño, construcción y pruebas del software, si no también aspectos administrativos complementarios (planificación del proyecto, aseguramiento de calidad e implantación de sistemas) [3]. Hoy, esta disciplina impone nuevos e importantes desafíos a los actuales desarrolladores de sistemas y especialmente a las aplicaciones desarrolladas para la Web, que tienen nuevas y especiales características que hacen que los mecanismos empleados hasta ahora se deban adaptar.

De la necesidad de adecuar los procesos de la Ingeniería de Software tradicionales a este entorno de rápido y constante cambio, surge la Ingeniería Web. En los últimos años se ha provisto de una serie de nuevas metodologías y herramientas para eficientar su quehacer. Como lo son las metodologías para el desarrollo de hipermedia HDM (Hypermedia Design Method), EORM (Enhanced Objetct Relationship Methodology), OOHDM (Object-Oriented Hypermedia Design), WSDM (Web Site Design Method) SODHM (Scenario-based Object-oriented Hypermedya Design Methodology), HFPM (Hypermedia Flexible Process Modeling) etc. [12].

La herramienta de modelado "Lenguaje de Modelado Unificado" [2] ha resultado un importante acontecimiento en la Ingeniería de Software, la mayoría de los desarrolladores la han adoptado como pieza fundamental para la especificación de todo tipo de aplicaciones. También existen extensiones de UML que han adaptado su notación para el desarrollo de aplicaciones Web [6]. Existen opiniones respecto a que estas adaptaciones no han resultado suficientes para representar satisfactoriamente aplicaciones Web [10], principalmente por la carencia de expresividad respecto a la arquitectura de navegación, la problemática de diseño gráfico y de presentación. Una herramienta que resuelve esta problemática es OOHDM, la cual separa diseño de datos y aspectos de uso y características de la interfaz, como navegabilidad, centrándose en la obtención de ventajas por el reuso de componentes.

Pese a esta gran cantidad de herramientas, el tema de la salida rápida al mercado sigue siendo reconocido como un factor crítico en el éxito del desarrollo Web. En un principio el desarrollo de las aplicaciones Web se centró en la construcción de soluciones simples, que básicamente consistían en poner documentos con hipertexto en la red. Para este tipo de aplicaciones no había mayores problemas debido a su misma naturaleza. Pero hoy podemos encontrar aplicaciones Web constituidas por software a gran escala y cubriendo un amplio espectro funcional: comercio electrónico, distribución de información, trabajos colaborativos, educación y numerosas otras actividades [7].

6.2 Desarrollo de Aplicaciones Web usando Métodos Ágiles

Como ya se ha indicado el desarrollo de aplicaciones Web tiene un tratamiento distinto al de una aplicación tradicional. Como los requerimientos muchas veces son desconocidos o bien variables los procesos ágiles pueden acomodarse a este tipo de desarrollo, ya que el supuesto de predictibilidad del conjunto de

requerimientos con que trabajan los enfoques tradicionales no se cumplen. De no existir un requisito estable y bien acotado, el diseño arrastrará la inestabilidad de estos y se tendrá así un diseño con las mismas características de inestabilidad que los requerimientos. En situaciones de este tipo un proceso que sea más adaptativo será más efectivo. Por otra parte los procesos de desarrollo adaptativos, también facilitan la generación rápida de prototipos. Un cliente, para el cual se está desarrollando una aplicación Web, no puede esperar llegar a una etapa de diseño para ver un prototipo de su aplicación, él requiere un diseño rápido para dar así su aprobación a la aplicación que se construye. Al usar un método ágil de desarrollo, por la flexibilidad que éste tiene, se debe convencer al cliente de que no hay una forma extremadamente planificada de hacer las cosas, lo cual puede ser para algunos algo chocante, la ventaja de esta menor planificación del esquema ágil sobre un método tradicional es la flexibilidad a los cambios de requerimientos [13].

Una aplicación Web, cumple en gran parte las características indicadas para el tipo de proyecto donde utilizar métodos ágiles puede resultar beneficioso. Las necesidades de contar con aplicaciones Web son por lo general contra el tiempo, y con requerimientos inestables.

Un cliente que contrata un desarrollo Web requiere que su producto esté disponible en la red lo más pronto posible. Si no se contempla esto, la aplicación no resultará un producto satisfactorio para el cliente. Como los procesos ágiles permiten obtener versiones de producto previas a la versión final, si se aplican adecuadamente estos procesos los clientes podrán disponer de forma rápida de alguna versión intermedia. Además el ciclo de desarrollo de la mayoría de los sitios y aplicaciones Web es extremadamente corto. Esto implica que generalmente no se aplique ningún tipo de proceso.

Los desarrollos Web son percibidos por el cliente como desarrollos sencillos, sometiendo al equipo de desarrollo a entregas prontas de software trabajando. Otro aspecto relevante a tener en cuenta es que las aplicaciones Web se desarrollan sin conocer los perfiles de los usuarios finales de las mismas, lo que hace a los requerimientos riesgosos y poco estables. Por lo tanto se puede concluir que los procesos ágiles son apropiados al desarrollo de aplicaciones Web.

7. CONCLUSIONES

Muchos autores a la fecha vienen indicando que las metodologías tradicionales no son totalmente adecuadas para todos los desarrollos software. Las razones son diversas, las principales son la falta de flexibilidad de los procesos de desarrollo frente a cambios y su excesiva documentación. Como se ha planteado en este trabajo los métodos ágiles y los tradicionales no son estrictamente competidores directos. Cada uno de ellos tiene su propio segmento de aplicación o terreno. Y pueden ser usados en proyectos con diferentes características: los métodos tradicionales son más adecuados en grandes proyectos con requerimientos estables y-o en aplicaciones críticas. Los métodos ágiles en cambio se adecuan mejor en ambientes dinámicos, con equipos de trabajo pequeños y produciendo aplicaciones no críticas. También son una buena elección cuando se trabaja con requerimientos desconocidos o inestables, garantizando un menor riesgo ante la posibilidad de cambio en los requerimientos.

Se presento también una lista de limitaciones de los procesos ágiles para clarificar su dominio de aplicación, ciertos dominios pueden ser más adecuados a este tipo de procesos. Entre los cuales se destaca el desarrollo

de aplicaciones Web, las que cumplen en gran parte con las características apropiadas para la utilización de métodos ágiles, los que permiten la incorporación de nuevos requerimientos sin exigir una excesiva generación de documentación.

En general algunos aspectos del desarrollo de software se beneficiarán del enfoque agilista mientras otros obtendrán beneficios de un enfoque tradicional-predictivo menos ágil. Desde esta perspectiva los procesos de desarrollo de software podrán ser clasificados dentro de un amplio espectro dependiendo de su "grado de agilidad". Lo importante es saber ubicarse debidamente dentro de él y optar por el tipo de proceso y herramientas que mejor sirvan a cada proyecto.

8. REFERENCIAS

[1] K. Beck et al, *Manifesto for Agile Software Development*, http://agilemanifesto.org/

[2] G. Booch et al, *El Lenguaje Unificado de Modelado*, Addison Wesley Iberoamericana, Madrid, 1999.

[3] R.G. Matheieu, *Top-Down Approach to Computing*, IEEE Computer, Vol. 35, Nº1 January 2002.

[4] S.I. Melnick & J.M. Barraza, *e-business, sí o sí,* Anticipa S.A, Santiago, 2002.

[5] B. Meyer, *Software Enginnering on Internet Time*, IEEE Computer, Vol. 34, Nº35, May 2001.

[6] S. Murugesan & Y. Deshpande, *Web Engineering : Managing Diversity and Complexity of Web Application Development*, Springer Verlag, 2001.

[7] J. Offutt, *Quality Attributes of Web Software Applications*, IEEE Software, Vol. 19, Nº2 Mach/April 2002.

[8] R.Pressman, *Ingeniería de Software. Un Enfoque Práctico*, Quinta Edición, McGraw-Hill / Iberoamericana, España, 2002.

[9] J. Ridderstrale & K. Nordstrom, *Funky Business: Talent Makes Capital Dance,* Financial Times / Prentice Hall, EEUU, 2000.

[10] D. Schwabe et al. *Engineering Web Applications for Reuse*, IEEE Multimedia, Special Issue on Web Engineering, Enero-Marzo 2001.

[11] I. Sommerville, *Ingeniería de Software, Sexta Edición*, Pearson Educación, México, 2002.

[12] N.P. de Koch. *Software Engineering for Adaptative Hypermedia System.* Tesis de Doctorado.

[13] P. Cáceres & E. Marcos, *Procesos Ágiles para el Desarrollo de Aplicaciones Web,* VI Jornadas de Ingeniería del Software y Bases de Datos, Almagro - España, 2001.

[14] M. Fowler, *The New Methodology*, http://www.programacionextrema.org/articulos/newMethodology.html

[15] U. Kelter et al, *Do we Need 'Agile' Software Development Tools?,* http://pi.informatik.uni-siegen.de/

[16] JUnit.org, http://junit.org/index.htm

[17] B. Boehm, *Using Risk to Balance Agile and Plan-Driven Methods*, IEEE Computer, Junio - 2003.

[18] K. Beck, *Embracing Change whith Extreme Programning*, IEEE Computer, Octubre – 1999

[19] I. Nonaka & H. Takeuchi, *Knowledge-Creating Company.* Oxford University Press, 1995

[20] D. Leffingwell & D. Widrig, *Managing Software Requirements,* Addison Wesley, Boston, 2003

[21] D. Turk et al, *Limitations of Agile Software Processes*, http://www.agilealliance.com/articles/articles/LimitationsofAgile.pdf