Una Propuesta de Conjunción de Elementos Metodológicos en común dentro de los Enfoques ágiles para el Desarrollo de Software.

Rodolfo Meda (rodolfomeda@yahoo.com), Jorge Ierache (jierache@yahoo.com.ar).

Instituto de Sistemas Inteligentes y Enseñaza experimental de la Robótica. Universidad de Morón – Facultad de Informática Ciencias de la Comunicación y Técnicas Especiales. Cabildo 134 Morón (1708), Provincia de Buenos Aires, Argentina. Teléfono: 54-11-5627-2000 (int 189/268/310)

Palabras Claves: Ingeniería de Software, Metodologías Ágiles, Ciclo de Vida, Gestión de Proyectos.

Resumen

Variadas propuestas metodológicas han sido introducidas en el área de la Ingeniería de Software. Se conoce que tuvieron un quiebre en el enfoque tradicional que las caracterizaba. Éste surge debido a la necesidad de un nuevo enfoque para el desarrollo de software; un enfoque ágil, que enfatice procesos adaptables más abarcativos, conjunción de mejores prácticas, productos de trabajo necesarios y los roles más característicos.

En este contexto, donde el advenimiento de las metodologías ágiles es ya una realidad, dentro del mundo de propuestas metodológicas en la Ingeniería de Software, el presente artículo intentará, desde el ámbito académico, lograr un estudio analítico de las mismas. Partiendo del trabajo Agile Software Development Method. Review and Analysis [8], este estudio propone la realización de un análisis comparativo basado en ciertos elementos metodológicos dados, obteniendo conclusiones, elementos metodológicos comunes y resultantes.

1. Introducción

Las metodologías ágiles se caracterizan por estar, mayormente, basadas en principios y valores [1]. Estos son los que guían al equipo de proyecto de software, ya que, normalmente, no se cuenta con un conjunto de especificaciones concretas respecto de aspectos metodológicos clave cubiertos [5]. Sin embargo, del universo de metodologías ágiles existentes [4] se pueden destacar cuatro casos particulares: XP [2; 3], Scrum [10; 11], DSDM [6] y FDD [9], los cuales se distinguen por proveer una guía concreta, dentro de su especificación, respecto de los procesos, prácticas, roles y productos de trabajo involucrados. Es decir, proveen una guía concreta respecto de estos elementos metodológicos mencionados. Por lo cual, estos, pueden convertirse en puntos clave que posibilitarán descubrir, a través de su estudio, cuáles son los aspectos metodológicos de naturaleza similar, dentro de las metodologías de desarrollo de software de enfoque ágil. Dicho estudio se realizará a través de un análisis comparativo que tomará los puntos clave indicados como criterios de evaluación, dentro del marco del ciclo de vida general del software, el cual se plantea como un entorno adecuado para delimitar el presente estudio.

2. Análisis Comparativo, Conclusiones, Elementos en Común y Resultantes.

Los procesos, prácticas, roles y productos de trabajo propuestos por cada metodología de enfoque ágil son los puntos clave, es decir, los criterios de evaluación que posibilitarán descubrir los aspectos de naturaleza similar, o elementos metodológicos comunes, dentro de estas metodologías.

Las etapas genéricas del ciclo de vida del software, han estado presentes, de una u otra forma, en las distintas propuestas metodologicas, tanto en el enfoque tradicional como en el enfoque ágil. Sin embargo, se debe tener en cuenta que la óptica que toma el enfoque ágil respecto de éstas es diferente. Las mismas son realizadas recurrentemente, en lapsos cortos de tiempo o ciclos de desarrollo (iteraciones), diferente al enfoque tradicional, donde cada etapa se completa solo una vez. Además, en el enfoque tradicional todas las tareas involucradas para la completitud de todas las funcionalidades del producto de software deben estar completas en cada etapa, para recién en ese momento poder pasar a la siguiente etapa. En el enfoque ágil se toma de a subgrupos de funcionalidades, y se las desarrolla en el ciclo de tiempo asignado, volviendo a repetir esto hasta completar todas las funcionalidades.

No obstante, como se mencionó, las etapas genéricas han estado presentes, de una u otra forma, en las distintas propuestas metodológicas tradicionales y de enfoque ágil, por lo que se consideran adecuadas, como un entorno apropiado para enmarcar el estudio de los criterios de evaluación propuestos. Considerado lo expuesto, tanto los puntos clave indicados, como el ciclo de vida general del software, en conjunto, permitirán contar con un marco de trabajo que encauzará el análisis comparativo.

De esta manera, se logrará la difícil tarea de comparar sistemáticamente una metodología con otra. Como se explica en [8], a menudo, el

resultado de las comparaciones se basa en experiencias subjetivas e intuiciones por parte de los autores.

2.1 Primer criterio: El Proceso

A continuación se expone el análisis comparativo realizado y conclusiones obtenidas, de las mismas se desprenden los elementos en común propuestos que caracterizan los aspectos de naturaleza similar, y, finalmente, se indican los procesos resultantes de unificar los elementos en común.

Procesos - Etapa de Concepción del Proyecto: Se observa que las metodologías ágiles evaluadas (XP, FDD, Scrum y DSDM) apuntan a contar con una fase que les permita realizar un análisis de factibilidad ligero, con el nivel de detalle adecuado para considerar la posibilidad concreta de realización del proyecto en estudio. Esta fase se realiza una sola vez, por lo que no es iterada. Sin embargo, las consideraciones obtenidas de la misma, serán actualizadas por los resultados de las fases subsiguientes.

Procesos - Etapa de Especificación de requerimientos: Aunque con diferente énfasis, la obtención de requerimientos es de vital importancia. Por lo tanto, las metodologías ágiles expuestas apuntan a cubrir con una o más fases las actividades de gestión de requerimientos, a definir en un primer momento los requerimientos generales o de alto nivel en una fase no iterada, para luego en fases posteriores e iteradas, refinar los mismos, y pasar a especificar requerimientos de manera más detallada.

Procesos - Etapa de Diseño: Se han dejado a un lado los grandes documentos de diseño y arquitectónicos. Sin embargo, aunque XP se desentienda de una fase que abarque el diseño, se entiende, en otras metodologías, que contar con ésta, es necesario para la correcta continuación de las actividades subsiguientes. Esto puede verse expresado en las fases propuestas por FDD, Scrum y DSDM; donde se enfatiza la realización de un diseño ligero, pero conciso, que exprese los puntos críticos que deben ser comprendidos. Eventualmente, resultará conveniente la realización de revisiones de los diseños propuestos.

Procesos - Etapa de Codificación: Las metodologías ágiles se enfocan en las actividades de desarrollo o codificación. Las fases expuestas se pueden agrupar en dos grandes clasificaciones. Por un lado, XP y FDD poseen fases bien explícitas respecto de las actividades a ser realizadas, y cómo realizarlas. Por otro lado, Scrum y DSDM, más orientadas a la gestión del proyecto, enmarcan con sus fases a las actividades de codificación y desarrollo, pero sin entrar en detalle respecto de cómo realizar las mismas.

Procesos - Etapa de Testeo: No hay una fase explícita vinculada al testeo. Esto se debe mayormente a lo siguiente: para las metodologías ágiles las actividades de testeo deben encontrarse presentes durante todo el proceso de desarrollo de software. Sin embargo, repasando las fases más explícitas, donde se considera necesario la existencia indefectiblemente del testeo como actividad particular, se desprende que las metodologías ágiles expuestas consideran las siguientes variantes: los testeos unitarios, de integración, de sistema, funcionales y de aceptación.

Procesos - Etapa de Producción: Excepto en el caso de FDD, todas las metodologías ágiles (XP, Scrum y DSDM) proponen fases para encauzar las actividades correspondientes al pasaje del sistema de software al ambiente de producción del usuario. Pero, se debe considerar que las prácticas, roles y productos de trabajo relacionados con estas fases están pobremente descriptos, o directamente son obviados.

De las conclusiones obtenidas del análisis se desprenden los elementos en común propuestos que caracterizan los aspectos de naturaleza similar. A continuación, se indican los procesos resultantes de unificar los elementos en común.

Elementos en Común	Elementos Resultantes
Fases de exploración (XP) y estudio de factibilidad (DSDM).	Fase de Exploración y Factibilidad.
Fases Exploración (XP), Construir una Lista de Características (FDD), Iteración del Modelo Funcional (DSDM) y Pre-Juego (Scrum).	Fase Funcionalidades Generales del Sistema de Software
Fases de Planificación (XP), Planear por Característica (FDD), Planeamiento (Scrum), Iteración del Modelo Funcional (DSDM).	Fase Funcionalidades para la Iteración
Fases Diseño y Construcción (DSDM), Diseño por Característica (FDD) y Arquitectura/Diseño de Alto nivel (Scrum).	Diseñar por Funcionalidades.
Fases Iteraciones para el lanzamiento (XP), Construcción por Característica (FDD), Fase de Desarrollo o Juego (Scrum) e Iteración de Diseño y Construcción (DSDM).	Construir por Funcionalidades.
Fases Producción,	Implementar por

Mantenimiento y Muerte (XP), Post-Juego (Scrum) e Implementación (DSDM).	Funcionalidades.

Tabla 1. Elementos metodológicos

2.2 Segundo criterio: Las Prácticas

Segundo criterio de evaluación: las prácticas. Las mismas se implementan dentro del marco de un proceso o ciclo de vida. Las comparaciones se han enfocado, mayormente, en las técnicas o prácticas [12]. Pero, pocas veces, ahondando en el detalle; en esta instancia, entonces, se realizará el análisis comparativo basándose en cómo se han especificado las prácticas en cada propuesta, dentro de cada etapa del ciclo de vida general del software.

Teniendo en cuenta el segundo criterio, se exponen el análisis comparativo y conclusiones obtenidas, luego se derivan los elementos en común propuestos que caracterizan los aspectos de naturaleza similar, y, finalmente, se presentan las prácticas resultantes de aunar los elementos en común.

Prácticas - Etapa de Concepción del Proyecto: apuntan a contar con una visión general del sistema de software en cuestión, de manera que permita reconocer su alcance. Para la construcción de la visión general se deben trabajar los requerimientos de más alto nivel, tanto funcionales como técnicos. Siempre, a través del consenso, entre los miembros del negocio y los miembros de desarrollo del equipo. Lograda la visión general del sistema, su alcance y características tanto funcionales como técnicas, se podrá obtener un estudio de factibilidad ligero del proyecto de software.

Prácticas - Etapa de Especificación de Requerimientos: proponen contar con prácticas que le permitan gestionar los requerimientos de manera sencilla y práctica, sin entrar en aspectos ceremoniales innecesarios. Se sigue una misma línea en todas las prácticas propuestas, en primer lugar, lograr la especificación de requerimientos generales, para luego, refinarlos concurrentemente hasta llevarlos a especificaciones detalladas. Esto último se realiza por subgrupos de requerimientos generales.

Prácticas - Etapa de Diseño: se manifiesta la necesidad de conservar una práctica de diseño simple y ligero, con el objetivo de expresar y transmitir de manera clara y concisa cuáles son los puntos críticos para el desarrollo de la solución de cada requerimiento. Las prácticas de diseño utilizan como técnica de modelado el diseño orientado a objetos, y diagramas de clase, de secuencia, de colaboración, etc.

Prácticas - Etapa de Codificación: XP y FDD proponen prácticas puntuales para la actividad de desarrollo o codificación; aunque con enfoques dispares en algunos aspectos. El caso de la práctica conocida como Recodificación, que aunque de valor técnico reconocible para XP, no realiza ningún aporte al usuario, según FDD. La propiedad de código colectiva de XP, se opone a la propiedad de clases individual de FDD [7]. Incluso, XP no representa jerarquías en cuanto a los desarrolladores, mientras, FDD es bien explícita al respecto.

Por otro lado, Scrum y DSDM sólo brindan un marco para el desenvolvimiento de las actividades de ingeniería de software. Las prácticas puntuales a ser utilizadas, son seleccionadas en cada caso particular, según se considere necesario.

Prácticas - Etapa de Testeo: se orientan a realizar diferentes variantes de testeos, como son el testeo de componentes, de integración, de sistema y de aceptación, que permiten contar con un sistema de software testeado, tanto desde la óptica técnica del equipo de desarrollo como desde la óptica funcional del equipo del negocio. Teniendo en cuenta que el software testeado satisfactoriamente contará con los estándares de calidad acordados.

Prácticas - Etapa de Producción: las metodologías ágiles no hacen referencia a esta etapa del ciclo de vida, respecto de las prácticas que proponen. Manifiestan únicamente, un encuadre a través de las fases propuestas.

Las conclusiones obtenidas del análisis permiten derivar en elementos en común que caracterizan los aspectos de naturaleza similar. Finalmente, se presentan las prácticas resultantes de aunar los elementos en común.

Elementos en Común	Elementos Resultantes
Prácticas Juego de la Planificación y Metáfora (XP), Modelado del Objeto de Dominio (FDD) y Product Backlog (Scrum).	Modelado y Planificación del Sistema de Software.
Prácticas Planificación de la Iteración (XP), Desarrollar por Característica (FDD) y Sprint Backlog (Scrum).	Funcionalidades para la Iteración
Prácticas Desarrollar por Característica (FDD) y Sprint (Scrum).	Desarrollar por Funcionalidades para la Iteración
Prácticas Planificación de la Iteración (XP), Desarrollar por Característica (FDD) y Sprint (Scrum).	Iteraciones
Prácticas Programación en Pares (XP), Equipos por Características (FDD)	Equipos por Funcionalidad

y Equipos de Scrum (Scrum).	
Prácticas de Diseño Simple (XP) y Desarrollar por Característica (FDD).	Diseño Ligero por Funcionalidad
Prácticas Estándares de Codificación, Hacer Pruebas (XP) e Inspecciones (FDD).	Estándares, Pruebas e Inspecciones
Prácticas Construcciones Regulares (FDD) e Integración Continua (XP).	Construcciones e Integraciones Regulares por Funcionalidad
Prácticas Propiedad Colectiva (XP) y Propiedad de Clases Individual (FDD).	Propiedad del Código

Tabla 2. Elementos metodológicos

2.3 Tercer criterio: Los Productos de Trabajo

Tercer criterio de evaluación: productos de trabajo. Teniendo en cuenta las etapas, en el contexto del ciclo de vida general, se exponen y analizan qué productos de trabajo proponen las distintas metodologías.

Productos de trabajo - Etapa de Concepción del Proyecto: XP, FDD y Scrum apuntan a contar herramientas gráficas o escritas (o combinación de ambas) que le permitan especificar los requerimientos de alto nivel. Esto posibilitará contar con elementos que le proporcionen una evaluación del alcance del sistema de software. Lograda la visión general del sistema, su ámbito y características generales tanto funcionales como técnicas, se podrá obtener un estudio de factibilidad ligero del proyecto de software.

Productos de trabajo - Etapa de Especificación de Requerimientos: los productos de trabajo propuestos por las metodologías ágiles para cumplir con esta etapa tratan mayormente de Listas de Requerimientos, según las variantes propuestas por FDD, Scrum y DSDM, que permiten gestionar tanto los requerimientos generales como los más detallados, adicionando información complementaria, de ser necesario.

Productos de trabajo - Etapa de Diseño: las únicas dos metodologías que soportan esta etapa con productos de trabajo específicos son FDD y XP. Ambas se enfocan en contar con Modelos Ligeros o Paquetes de Diseño que le permitan mostrar, de manera simple y concisa, los diseños involucrados, para comunicar eficientemente los puntos clave en el desarrollo.

Productos de trabajo - Etapa de Codificación: Los productos de trabajo esenciales que proponen las metodologías ágiles son aquellos que permitan gestionar las actividades de desarrollo durante esta etapa del ciclo de vida. Es decir, que contengan prioridades, tareas asignadas, ordenamiento, estimaciones, etc. Los mismos no serán obtenidos como productos provenientes de contemplar esta etapa, pero si pueden llegar a ser actualizados como resultado de la retroalimentación de las actividades desarrolladas.

Productos de trabajo - Etapa de Testeo: Es indispensable contar con productos de trabajo que permitan conocer los resultados obtenidos en las distintas actividades de testeo. Se plantean distintos tipos de reportes, tanto gráficos como documentos escritos, que permitan registrar e informar, a través de porcentajes de completitud, los resultados de los testeos.

Productos de trabajo - Etapa de Producción: A excepción de DSDM, donde se utiliza un Reporte de Revisión, sobre el cual se registran las actividades y resultados de las actividades en los ambientes de producción, las demás metodologías ágiles no cuentan con productos de trabajo específicos para ser utilizados durante esta etapa.

Las conclusiones del análisis realizado permiten extraer los elementos en común propuestos que caracterizan los aspectos de naturaleza similar, y, finalmente, a continuación se presentan los productos de trabajo resultantes de aunar los elementos en común.

Elementos en Común	Elementos Resultantes
Productos de Trabajo Lista de Tareas (XP), Lista de Características Principales (FDD), Lista de Requerimientos (DSDM) y Product Backlog (Scrum).	Lista de Funcionalidades Generales
Productos de Trabajo Plan de Desarrollo (FDD), Lista de Tareas (XP) y Plan de Desarrollo (DSDM).	Plan General del Sistema de Software
Productos de Trabajo Modelo del Objeto de Dominio (FDD) y Metáfora (XP).	Modelo General del Sistema
Productos de Trabajo Lista de Características Detalladas (FDD) y el Sprint Backlog (Scrum).	Lista de Funcionalidades Detalladas
Productos de Trabajo Modelo de Diseño (XP) y el Paquete de Diseño (FDD).	Modelo de Diseño Ligero

Tabla 3. Elementos metodológicos

2.4 Cuarto criterio: Los Roles

Finalmente, contemplando el cuarto criterio, se exponen el análisis comparativo y conclusiones obtenidas, de estas últimas se infieren los elementos en común propuestos que caracterizan los aspectos de naturaleza similar, y, finalmente, se presentan

los productos de trabajo resultantes de aunar los elementos en común.

*Roles - Etapa de Concepción del Proyecto***:** los roles involucrados en esta etapa del ciclo de vida deben permitir contar con las personas responsables para decidir, mediante la obtención del alcance del sistema de software, si el proyecto es factible tanto a nivel de negocio, como a nivel técnico. Es probable que, como se menciona en XP, FDD y DSDM, exista la necesidad de contar con la presencia de especialistas en cuestiones tecnológicas y, en particular, especialistas en métodos de enfoque ágil, quienes realizan asesoramiento sobre si la aplicación de un enfoque de estas características es compatible con la naturaleza del proyecto a ser encarado.

Roles - Etapa de Especificación de Requerimientos: se propone que tanto los miembros del equipo del negocio como miembros del equipo de desarrollo, en sus distintas variantes, participen activamente. Dicha participación debe ser realizada estrechamente en conjunto, con una retroalimentación constante. Se debe recordar que los productos de trabajo no son muy detallados respecto de las necesidades o requerimientos, ya que justamente se pretende que se puedan refinar a través de una intensa interrelación entre el desarrollador y el usuario.

Roles - Etapa de Diseño: es llevada a cabo únicamente por miembros del equipo de desarrollo. Aunque no existe un rol particular que tome específicamente esta actividad, por lo general los miembros más experimentados (según, FDD y DSDM) son los indicados para obtener diseños más ligeros, prácticos y comunicativos, tanto como sea posible, también deberán mantener actualizados los documentos de diseño hasta el final del proceso.

Roles - Etapa de Codificación: hay un rol presente en todas las metodologías ágiles, el rol de desarrollador o programador. El énfasis está puesto en facilitarle todos los aspectos necesarios, de manera que le permita realizar su trabajo de codificación eficientemente, y sin obstrucciones.

Roles - Etapa de Testeo: el rol del programador es decisivo en esta etapa, de manera que continúe con los testeos de componentes e integración. Adicionalmente al de programador, se proponen el rol de testeador (FDD) o el de Encargado de Pruebas (XP). Estos roles comparten una responsabilidad de naturaleza similar, tendrán como objetivo principal ayudar a los usuarios finales a realizar los testeos de aceptación o funcionales.

Roles - Etapa de Producción: XP, Scrum y DSDM comprometen a los roles más representativos, tanto del equipo de desarrollo como del equipo de negocio, para que logren el consenso necesario respecto de la decisión sobre el mejor momento para utilizar el sistema de software en el ambiente de producción. Previo a esta instancia, en la medida que se fueron implementando las distintas funcionalidades en producción, los miembros del equipo de negocio realizaron los testeos funcionales y de aceptación.

Las conclusiones obtenidas, posibilitan inferir los elementos en común que caracterizan los aspectos de naturaleza similar. En la siguiente tabla se presentan los productos de trabajo resultantes de unificar los elementos en común.

Elementos en Común	Elementos Resultantes
Roles Cliente (XP), Cliente (Scrum), Expertos del Dominio (FDD), Usuario Embajador (DSDM).	Cliente/Usuario Final.
Roles de Programador (XP), Programadores Jefes, Propietarios de Clase (FDD), Equipo de Scrum (Scrum) y Desarrolladores (DSDM).	Desarrollador
Roles de Gran Jefe (XP), Gerente de Proyecto (FDD) y Scrum Master (Scrum).	Gerente de Proyecto
Roles testeador (FDD) y el de Encargado de Pruebas (XP).	Testeador o Encargado de Pruebas

Tabla 4 Elementos metodológicos.

3. Conclusiones

Se ha logrado en el presente trabajo obtener un Análisis Comparativo, Conclusiones, Elementos Metodológicos en Común y Resultantes, provenientes del estudio y desarrollo teórico de las metodologías ágiles tratadas..

4. Referencias

[1] Manifiesto Ágil, http://agilemanifesto.org/

[2] Beck, K. 2002, Una explicación de la programación extrema. Aceptar el Cambio. Pearson Educación S.A.

[3] Beck, K. 1999, Embracing Change with Extreme Programming. First Class Software. IEEE Computer Society.

[4] Highsmith, J. 2002, Agile Software Development Ecosystems, Addison-Wesley Professional.

[5] Largman, C. 2004, Agile and Iterative Development: A manager's guide. Addison-Wesley.

[6] Stapleton, J. 1997, Dynamic Systems Development Method. The Method in Practice. Addison-Wesley.

[7] Khramtchenko, S. 2004, Comparing eXtreme Programming and Feature Driven Development in Academic and regulated environments. Software Architecture and Engineering. Harvard University.

[8] Abrahamsson, P., Salo, O., Ronkainen, J. 2002, Agile Software Development Method. Review and Analysis. VTT Technical Research Centre of Finland.

[9] Palmer, S., Felsing, J. 2002, A Practical Guide to Feature-Driven Development. The Coad Series. Prentice Hall.

[10] Schwaber, K., Beedle M. 2002, Agile Software Development with Scrum. Prentice Hall.

[11] Schwaber, K. 2002, Scrum Development Process. Advanced Development Method.

[12] Tuffs, D., Stapleton, J., West, D., Eason, Z. 1999, Inter-operability of DSDM with the RUP. Rational and DSDM Consortium.